Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления (ИУ)

КАФЕДРА Система обработки информации и управления

ДИСЦИПЛИНА Методы машинного обучения в автоматизированных системах обработки информации и управления

ОТЧЕТ ПО ДОМАШНЕМУ ЗАДАНИЮ

Группа ИУ5-22М

Студент _____ _____ Хижняков В. М.
дата выполнения работы *подпись* *фамилия, и.о.*

Преподаватель _____ Гапанюк Ю. Е.
подпись *фамилия, и.о.*

Москва, 2024г.

Hate Speech Detection

Ввод [1]:
```
import pandas as panda
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem.porter import *
import string
import nltk
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix
import seaborn
from textstat.textstat import *
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn.feature_selection import SelectFromModel
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.svm import LinearSVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
import numpy as np
from nltk.sentiment.vader import SentimentIntensityAnalyzer as VS
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
%matplotlib inline
```

Ввод [2]:
```
dataset = panda.read_csv("../data/HateSpeechData.csv")
dataset
```

Out[2]:

	Unnamed: 0	count	hate_speech	offensive_language	neither	class	tweet
0	0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't...
1	1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	4	6	0	6	0	1	!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...
...	...	...	...	...	...	...	...
24778	25291	3	0	2	1	1	you's a muthaf***in lie “@LifeAsKing: @2...
24779	25292	3	0	1	2	2	you've gone and broke the wrong heart baby, an...
24780	25294	3	0	3	0	1	young buck wanna eat!!.. dat nigguh like I ain...
24781	25295	6	0	6	0	1	youu got wild bitches tellin you lies
24782	25296	3	0	0	3	2	~~Ruffled | Ntac Eileen Dahlia - Beautiful col...

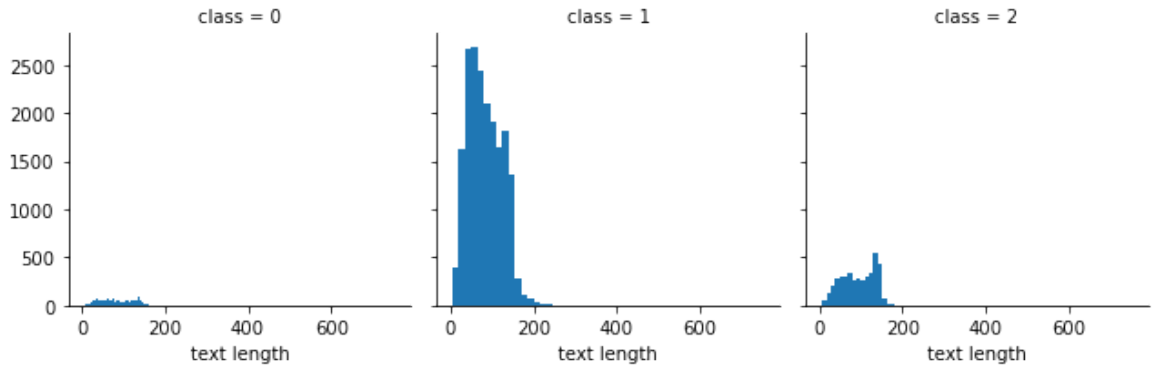
24783 rows × 7 columns

Ввод [3]:
```
# Добавляем поле с длиной твита
dataset['text length'] = dataset['tweet'].apply(len)
print(dataset.head())
```

```
   Unnamed: 0  count  hate_speech  offensive_language  neither  clas
s  \
0           0      3            0                   0        3
2
1           1      3            0                   3        0
1
2           2      3            0                   3        0
1
3           3      3            0                   2        1
1
4           4      6            0                   6        0
1

                                                tweet  text length
0  !!! RT @mayasolovely: As a woman you shouldn't...           140
1  !!!!! RT @mleew17: boy dats cold...tyga dwn ba...            85
2  !!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...           120
3  !!!!!!!!! RT @C_G_Anderson: @viva_based she lo...            62
4  !!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...           137
```

Ввод [4]:
```
import seaborn as sns
import matplotlib.pyplot as plt
graph = sns.FacetGrid(data=dataset, col='class')
graph.map(plt.hist, 'text length', bins=50)
```

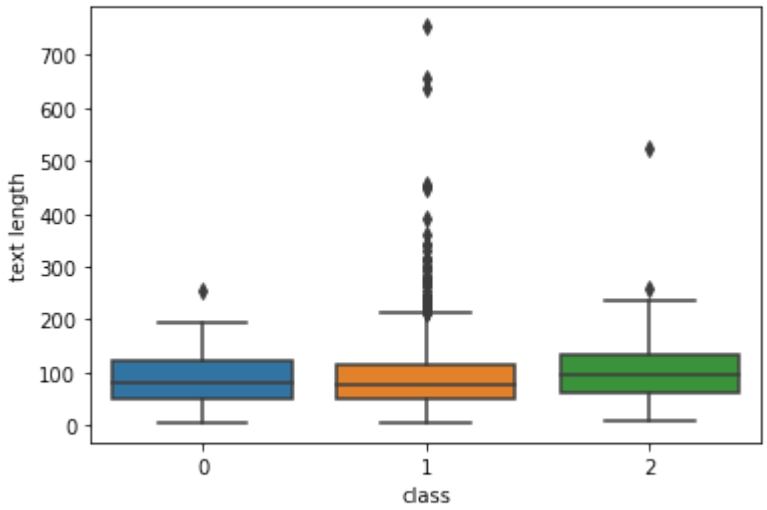
Out[4]:
```
<seaborn.axisgrid.FacetGrid at 0x7fd7b84cf9a0>
```



1. Распределение длины текста почти одинаково во всех трех классах
2. Количество твитов класса 1 намного выше

Ввод [5]:
```
sns.boxplot(x='class', y='text length', data=dataset)
```

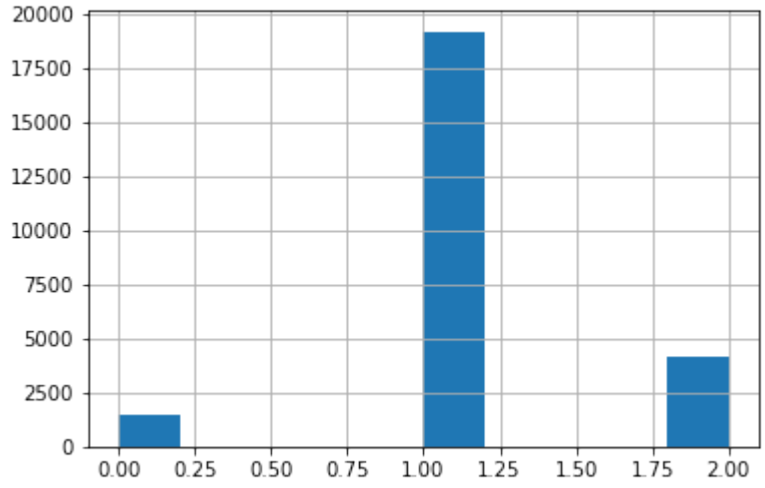
Out[5]:
```
<AxesSubplot:xlabel='class', ylabel='text length'>
```



Судя по диаграмме, твиты класса 1 содержат гораздо более длинный текст. Также присутствуют выбросы, поэтому длину текста не стоит учитывать

Ввод [6]:
```
dataset['class'].hist()
```

Out[6]:
```
<AxesSubplot:>
```



Большинство твитов оскорбительные

Ввод [7]:
```
tweet=dataset.tweet
```

Предобработка

Ввод [8]:

```
## 1. Удаление знаков препинания и заглавных букв
## 2. Токенизация
## 3. Удаление стоп-слов
## 4. Стемминг

stopwords = nltk.corpus.stopwords.words("english")

#extending the stopwords to include other words used in twitter such a
other_exclusions = ["#ff", "ff", "rt"]
stopwords.extend(other_exclusions)
stemmer = PorterStemmer()

def preprocess(tweet):

    # removal of extra spaces
    regex_pat = re.compile(r'\s+')
    tweet_space = tweet.str.replace(regex_pat, ' ')

    # removal of @name[mention]
    regex_pat = re.compile(r'@[\w\-]+')
    tweet_name = tweet_space.str.replace(regex_pat, '')

    # removal of links[https://abc.com]
    giant_url_regex =  re.compile('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.
            '[!*\(\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+')
    tweets = tweet_name.str.replace(giant_url_regex, '')

    # removal of punctuations and numbers
    punc_remove = tweets.str.replace("[^a-zA-Z]", " ")
    # remove whitespace with a single space
    newtweet=punc_remove.str.replace(r'\s+', ' ')
    # remove leading and trailing whitespace
    newtweet=newtweet.str.replace(r'^\s+|\s+?$','')
    # replace normal numbers with numbr
    newtweet=newtweet.str.replace(r'\d+(\.\d+)?','numbr')
    # removal of capitalization
    tweet_lower = newtweet.str.lower()

    # tokenizing
    tokenized_tweet = tweet_lower.apply(lambda x: x.split())

    # removal of stopwords
    tokenized_tweet=  tokenized_tweet.apply(lambda x: [item for item i

    # stemming of the tweets
    tokenized_tweet = tokenized_tweet.apply(lambda x: [stemmer.stem(i)

    for i in range(len(tokenized_tweet)):
        tokenized_tweet[i] = ' '.join(tokenized_tweet[i])
        tweets_p= tokenized_tweet

    return tweets_p

processed_tweets = preprocess(tweet)

dataset['processed_tweets'] = processed_tweets
print(dataset[["tweet","processed_tweets"]].head(10))
```

```
                                                   tweet  \
0  !!! RT @mayasolovely: As a woman you shouldn't...
1  !!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2  !!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3  !!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4  !!!!!!!!!!!! RT @ShenikaRoberts: The shit you...
5  !!!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just...
6  !!!!!!"@__BrighterDays: I can not just sit up ...
7  !!!!&#8220;@selfiequeenbri: cause I'm tired of...
8  " &amp;amp; you might not get ya bitch back &amp;amp; ...
9  " @rhythmixx_ :hobbies include: fighting Maria...

                                     processed_tweets
0  woman complain clean hous amp man alway take t...
1  boy dat cold tyga dwn bad cuffin dat hoe st place
2         dawg ever fuck bitch start cri confus shit
3                                     look like tranni
4     shit hear might true might faker bitch told ya
5      shit blow claim faith somebodi still fuck hoe
6              sit hate anoth bitch got much shit go
7            caus tire big bitch come us skinni girl
8               amp might get ya bitch back amp that
9                    hobbi includ fight mariam bitch
```

Построение доп признаков

Ввод [9]:
```
#TF-IDF Features-F1
# https://scikit-learn.org/stable/modules/generated/sklearn.feature_ex
tfidf_vectorizer = TfidfVectorizer(ngram_range=(1, 2),max_df=0.75, mir

# TF-IDF feature matrix
tfidf = tfidf_vectorizer.fit_transform(dataset['processed_tweets'] )
tfidf
```

Out[9]:
```
<24783x6441 sparse matrix of type '<class 'numpy.float64'>'
        with 189618 stored elements in Compressed Sparse Row format>
```

Запуск различных моделей. Использование TFIDF без дополнительных признаков.

Ввод [10]:
```
# If you don't specify the random_state in the code,
# then every time you run(execute) your code a new random value is ger
# and the train and test datasets would have different values each tim
X = tfidf
y = dataset['class'].astype(int)
X_train_tfidf, X_test_tfidf, y_train, y_test = train_test_split(X, y,
model = LogisticRegression().fit(X_train_tfidf,y_train)
y_preds = model.predict(X_test_tfidf)
report = classification_report( y_test, y_preds )
print(report)
acc=accuracy_score(y_test,y_preds)
print("Logistic Regression, Accuracy Score:" , acc)
```

```
              precision    recall  f1-score   support

           0       0.56      0.18      0.27       290
           1       0.92      0.96      0.94      3832
           2       0.85      0.84      0.85       835

    accuracy                           0.90      4957
   macro avg       0.77      0.66      0.68      4957
weighted avg       0.88      0.90      0.88      4957

Logistic Regression, Accuracy Score: 0.8975186604801291

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_logistic.py:763: ConvergenceWarning: lbfgs failed to conver
ge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as sh
own in:
    https://scikit-learn.org/stable/modules/preprocessing.html (http
s://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver option
s:
    https://scikit-learn.org/stable/modules/linear_model.html#logist
ic-regression (https://scikit-learn.org/stable/modules/linear_model.
html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Ввод [11]:
```
X_train_tfidf, X_test_tfidf, y_train, y_test = train_test_split(X, y,
rf=RandomForestClassifier()
rf.fit(X_train_tfidf,y_train)
y_preds = rf.predict(X_test_tfidf)
acc1=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Random Forest, Accuracy Score:",acc1)
```

```
              precision    recall  f1-score   support

           0       0.53      0.17      0.26       290
           1       0.93      0.96      0.94      3832
           2       0.83      0.91      0.87       835

    accuracy                           0.90      4957
   macro avg       0.76      0.68      0.69      4957
weighted avg       0.89      0.90      0.89      4957

Random Forest, Accuracy Score: 0.9035707080895703
```

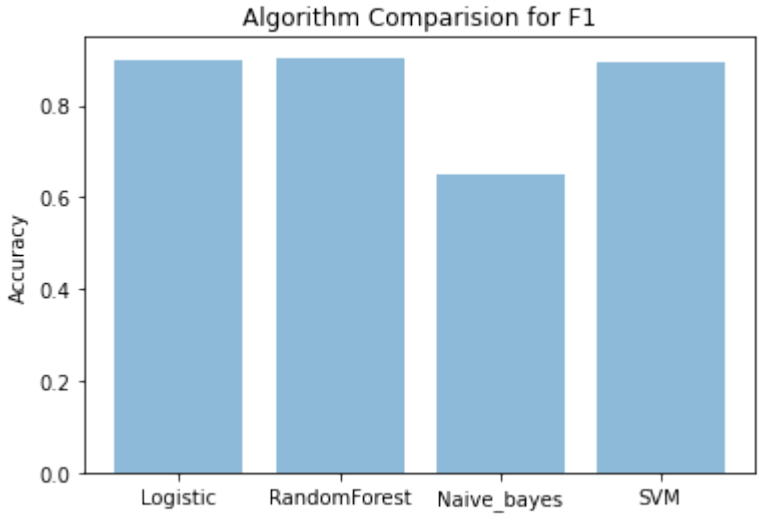
Ввод [12]:
```
X_train_tfidf, X_test_tfidf, y_train, y_test = train_test_split(X.toar
nb=GaussianNB()
nb.fit(X_train_tfidf,y_train)
y_preds = nb.predict(X_test_tfidf)
acc2=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Naive Bayes, Accuracy Score:",acc2)
```

```
              precision    recall  f1-score   support

           0       0.10      0.39      0.16       290
           1       0.89      0.68      0.77      3832
           2       0.54      0.58      0.56       835

    accuracy                           0.65      4957
   macro avg       0.51      0.55      0.50      4957
weighted avg       0.79      0.65      0.70      4957

Naive Bayes, Accuracy Score: 0.6491829735727255
```

Ввод [13]:
```
support = LinearSVC(random_state=20)
support.fit(X_train_tfidf,y_train)
y_preds = support.predict(X_test_tfidf)
acc3=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("SVM, Accuracy Score:" , acc3)
```

```
              precision    recall  f1-score   support

           0       0.46      0.26      0.33       290
           1       0.92      0.95      0.94      3832
           2       0.83      0.85      0.84       835

    accuracy                           0.89      4957
   macro avg       0.74      0.69      0.70      4957
weighted avg       0.88      0.89      0.89      4957

SVM, Accuracy Score: 0.8932822271535202
```

Ввод [14]:
```
objects = ('Logistic', 'RandomForest', 'Naive_bayes', 'SVM')
y_pos = np.arange(len(objects))
performance = [acc,acc1,acc2,acc3]
plt.bar(y_pos, performance, align='center', alpha=0.5)
plt.xticks(y_pos, objects)
plt.ylabel('Accuracy')
plt.title('Algorithm Comparision for F1')
plt.show()
```



Анализ тональности с использованием оценок полярности в качестве признаков

Ввод [15]:

```
sentiment_analyzer = VS()
def count_tags(tweet_c):

    space_pattern = '\s+'
    giant_url_regex = ('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|'
        '[!*\(\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+')
    mention_regex = '@[\w\-]+'
    hashtag_regex = '#[\w\-]+'
    parsed_text = re.sub(space_pattern, ' ', tweet_c)
    parsed_text = re.sub(giant_url_regex, 'URLHERE', parsed_text)
    parsed_text = re.sub(mention_regex, 'MENTIONHERE', parsed_text)
    parsed_text = re.sub(hashtag_regex, 'HASHTAGHERE', parsed_text)
    return(parsed_text.count('URLHERE'),parsed_text.count('MENTIONHERE

def sentiment_analysis(tweet):
    sentiment = sentiment_analyzer.polarity_scores(tweet)
    twitter_objs = count_tags(tweet)
    features = [sentiment['neg'], sentiment['pos'], sentiment['neu'],
                twitter_objs[2]]
    #features = pandas.DataFrame(features)
    return features

def sentiment_analysis_array(tweets):
    features=[]
    for t in tweets:
        features.append(sentiment_analysis(t))
    return np.array(features)

final_features = sentiment_analysis_array(tweet)
#final_features

new_features = panda.DataFrame({'Neg':final_features[:,0],'Pos':final_
                                'url_tag':final_features[:,4],'mention_tag
new_features
```

Out[15]:

	Neg	Pos	Neu	Compound	url_tag	mention_tag	hash_tag
0	0.000	0.120	0.880	0.4563	0.0	1.0	0.0
1	0.237	0.000	0.763	-0.6876	0.0	1.0	0.0
2	0.538	0.000	0.462	-0.9550	0.0	2.0	0.0
3	0.000	0.344	0.656	0.5673	0.0	2.0	0.0
4	0.249	0.081	0.669	-0.7762	0.0	1.0	1.0
...	...	...	...	...	...	...	...
24778	0.000	0.000	1.000	0.0000	0.0	3.0	3.0
24779	0.454	0.000	0.546	-0.8074	0.0	0.0	0.0
24780	0.000	0.219	0.781	0.4738	0.0	0.0	0.0
24781	0.573	0.000	0.427	-0.7717	0.0	0.0	0.0
24782	0.000	0.218	0.782	0.5994	1.0	0.0	0.0

24783 rows × 7 columns

Ввод [16]:
```
# F2-Conctaenation of tf-idf scores and sentiment scores
tfidf_a = tfidf.toarray()
modelling_features = np.concatenate([tfidf_a,final_features],axis=1)
modelling_features.shape
```

Out[16]: (24783, 6448)

Запуск различных моделей. Использование TFIDF и дополнительных признаков.

Ввод [17]:
```
# Running the model Using TFIDF with some features from sentiment anal

X = panda.DataFrame(modelling_features)
y = dataset['class'].astype(int)
X_train_bow, X_test_bow, y_train, y_test = train_test_split(X, y, ranc

model = LogisticRegression().fit(X_train_bow,y_train)
y_preds = model.predict(X_test_bow)
report = classification_report( y_test, y_preds )
print(report)
acc=accuracy_score(y_test,y_preds)
print("Logistic Regression,Accuracy Score:" , acc)
```

```
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_logistic.py:763: ConvergenceWarning: lbfgs failed to conver
ge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as sh
own in:
    https://scikit-learn.org/stable/modules/preprocessing.html (http
s://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver option
s:
    https://scikit-learn.org/stable/modules/linear_model.html#logist
ic-regression (https://scikit-learn.org/stable/modules/linear_model.
html#logistic-regression)
  n_iter_i = _check_optimize_result(

              precision    recall  f1-score   support

           0       0.60      0.19      0.29       290
           1       0.92      0.96      0.94      3832
           2       0.85      0.84      0.85       835

    accuracy                           0.90      4957
   macro avg       0.79      0.67      0.69      4957
weighted avg       0.89      0.90      0.89      4957

Logistic Regression,Accuracy Score: 0.898930804922332
```

Ввод [18]:
```
X = panda.DataFrame(modelling_features)
y = dataset['class'].astype(int)
X_train_bow, X_test_bow, y_train, y_test = train_test_split(X, y, rand
rf=RandomForestClassifier()
rf.fit(X_train_bow,y_train)
y_preds = rf.predict(X_test_bow)
acc1=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Random Forest, Accuracy Score:",acc1)
```

```
              precision    recall  f1-score   support

           0       0.49      0.13      0.21       290
           1       0.91      0.97      0.94      3832
           2       0.85      0.83      0.84       835

    accuracy                           0.89      4957
   macro avg       0.75      0.64      0.66      4957
weighted avg       0.88      0.89      0.88      4957

Random Forest, Accuracy Score: 0.8944926366754085
```

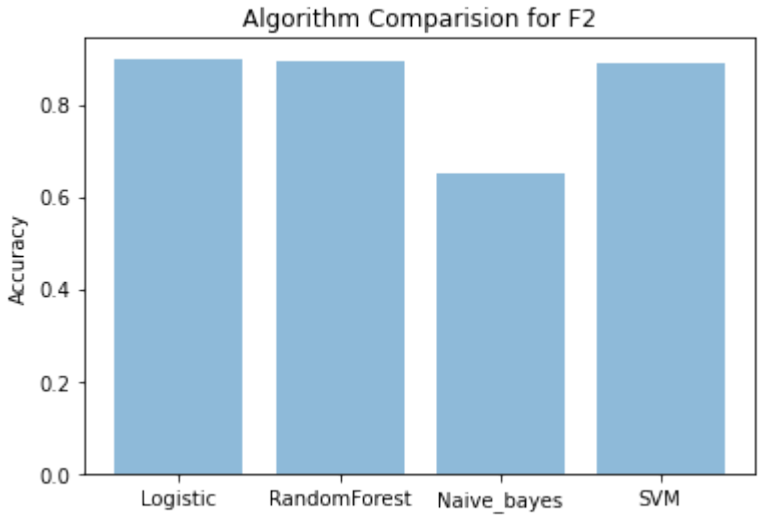
Ввод [19]:
```
X = panda.DataFrame(modelling_features)
y = dataset['class'].astype(int)
X_train_bow, X_test_bow, y_train, y_test = train_test_split(X, y, rand
nb=GaussianNB()
nb.fit(X_train_bow,y_train)
y_preds = nb.predict(X_test_bow)
acc2=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Naive Bayes, Accuracy Score:",acc2)
```

```
              precision    recall  f1-score   support

           0       0.10      0.39      0.16       290
           1       0.89      0.68      0.77      3832
           2       0.54      0.59      0.56       835

    accuracy                           0.65      4957
   macro avg       0.51      0.55      0.50      4957
weighted avg       0.79      0.65      0.70      4957

Naive Bayes, Accuracy Score: 0.650191648174299
```

Ввод [20]:
```
X = panda.DataFrame(modelling_features)
y = dataset['class'].astype(int)
X_train_bow, X_test_bow, y_train, y_test = train_test_split(X, y, rand
support =LinearSVC(random_state=20)
support.fit(X_train_bow,y_train)
y_preds = support.predict(X_test_bow)
acc3=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("SVM, Accuracy Score:" , acc3)
```

```
              precision    recall  f1-score   support

           0       0.46      0.26      0.33       290
           1       0.92      0.95      0.94      3832
           2       0.83      0.85      0.84       835

    accuracy                           0.89      4957
   macro avg       0.73      0.69      0.70      4957
weighted avg       0.88      0.89      0.88      4957

SVM, Accuracy Score: 0.8912648779503732

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_
base.py:985: ConvergenceWarning: Liblinear failed to converge, incre
ase the number of iterations.
  warnings.warn("Liblinear failed to converge, increase "
```

Ввод [21]:
```
objects = ('Logistic', 'RandomForest', 'Naive_bayes', 'SVM')
y_pos = np.arange(len(objects))
performance = [acc,acc1,acc2,acc3]
plt.bar(y_pos, performance, align='center', alpha=0.5)
plt.xticks(y_pos, objects)
plt.ylabel('Accuracy')
plt.title('Algorithm Comparision for F2')
plt.show()
```



Ввод [22]:
```
# create doc2vec vector columns
# Initialize and train the model
from gensim.test.utils import common_texts
from gensim.models.doc2vec import Doc2Vec, TaggedDocument
#The input for a Doc2Vec model should be a list of TaggedDocument(['li
#A good practice is using the indexes of sentences as the tags.
documents = [TaggedDocument(doc, [i]) for i, doc in enumerate(dataset[

# train a Doc2Vec model with our text data
# window- The maximum distance between the current and predicted word
# mincount-Ignores all words with total frequency lower than this.
# workers -Use these many worker threads to train the model
#  Training Model - distributed bag of words (PV-DBOW) is employed.
model = Doc2Vec(documents, vector_size=5, window=2, min_count=1, worke

#infer_vector - Infer a vector for given post-bulk training document.
# Syntax- infer_vector(doc_words, alpha=None, min_alpha=None, epochs=N
# doc_words-A document for which the vector representation will be inf

# transform each document into a vector data
doc2vec_df = dataset["processed_tweets"].apply(lambda x: model.infer_v
doc2vec_df.columns = ["doc2vec_vector_" + str(x) for x in doc2vec_df.c
doc2vec_df
```

Out[22]:

	doc2vec_vector_0	doc2vec_vector_1	doc2vec_vector_2	doc2vec_vector_3	doc2vec_vec
0	0.018026	0.017211	0.097070	0.068574	0.01
1	0.047766	0.085429	0.266810	0.073316	-0.07
2	-0.134311	0.126408	-0.005137	0.101088	0.02
3	0.096752	0.136317	-0.002050	-0.077735	0.05
4	0.082045	0.012922	0.092758	-0.154263	-0.05
...	...	...	...	...	
24778	0.215684	-0.075122	0.254222	-0.295271	-0.04
24779	-0.080566	0.119454	0.124048	-0.013470	-0.08
24780	-0.122564	0.177422	0.145220	0.028246	-0.12
24781	-0.073633	0.005331	0.035100	-0.165321	-0.09
24782	0.569885	-0.233530	0.410167	-0.355862	0.04

24783 rows × 5 columns

Ввод [23]:
```
# conctaenation of tf-idf scores, sentiment scores and doc2vec columns
modelling_features = np.concatenate([tfidf_a,final_features,doc2vec_df
modelling_features.shape
```

Out[23]: (24783, 6453)

Запуск моделей с использованием TFIDF с дополнительными признаками анализа настроений и doc2vec

Ввод [24]:
```
X = panda.DataFrame(modelling_features)
y = dataset['class'].astype(int)
X_train_bow, X_test_bow, y_train, y_test = train_test_split(X, y, rand

model = LogisticRegression().fit(X_train_bow,y_train)
y_preds = model.predict(X_test_bow)
report = classification_report( y_test, y_preds )
print(report)
acc=accuracy_score(y_test,y_preds)
print("Logistic Regression, Accuracy Score:" , acc)
```

```
              precision    recall  f1-score   support

           0       0.57      0.19      0.28       290
           1       0.92      0.96      0.94      3832
           2       0.84      0.84      0.84       835

    accuracy                           0.90      4957
   macro avg       0.78      0.66      0.69      4957
weighted avg       0.88      0.90      0.88      4957

Logistic Regression, Accuracy Score: 0.8975186604801291

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_logistic.py:763: ConvergenceWarning: lbfgs failed to conver
ge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as sh
own in:
    https://scikit-learn.org/stable/modules/preprocessing.html (http
s://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver option
s:
    https://scikit-learn.org/stable/modules/linear_model.html#logist
ic-regression (https://scikit-learn.org/stable/modules/linear_model.
html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Ввод [25]:
```
X = panda.DataFrame(modelling_features)
y = dataset['class'].astype(int)
X_train_bow, X_test_bow, y_train, y_test = train_test_split(X, y, ranc
rf=RandomForestClassifier()
rf.fit(X_train_bow,y_train)
y_preds = rf.predict(X_test_bow)
acc1=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Random Forest, Accuracy Score:",acc1)
```

```
              precision    recall  f1-score   support

           0       0.45      0.07      0.12       290
           1       0.90      0.97      0.93      3832
           2       0.86      0.78      0.82       835

    accuracy                           0.89      4957
   macro avg       0.74      0.61      0.63      4957
weighted avg       0.87      0.89      0.87      4957

Random Forest, Accuracy Score: 0.8884405890659673
```

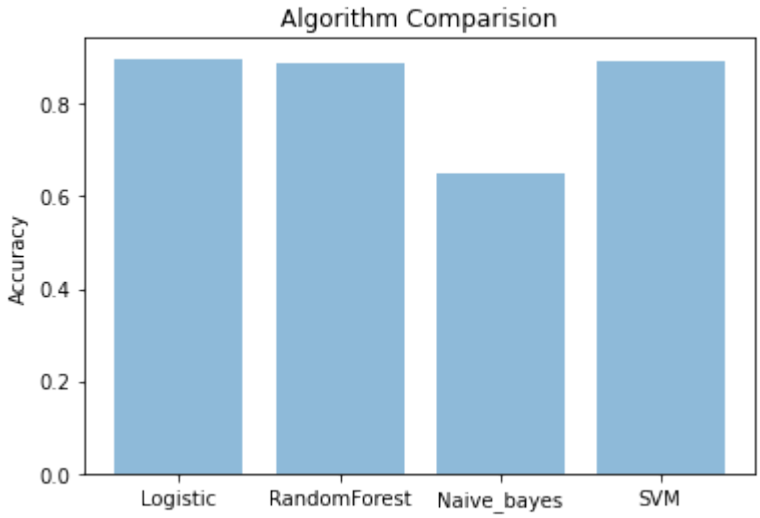
Ввод [26]:
```
X = panda.DataFrame(modelling_features)
y = dataset['class'].astype(int)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
nb=GaussianNB()
nb.fit(X_train,y_train)
y_preds = nb.predict(X_test)
acc2=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Naive Bayes, Accuracy Score:",acc2)
```

```
              precision    recall  f1-score   support

           0       0.10      0.39      0.16       290
           1       0.89      0.68      0.77      3832
           2       0.54      0.59      0.56       835

    accuracy                           0.65      4957
   macro avg       0.51      0.55      0.50      4957
weighted avg       0.79      0.65      0.70      4957

Naive Bayes, Accuracy Score: 0.650191648174299
```

Ввод [27]:
```
X = panda.DataFrame(modelling_features)
y = dataset['class'].astype(int)
X_train_bow, X_test_bow, y_train, y_test = train_test_split(X, y, rand
support =LinearSVC(random_state=20)
support.fit(X_train_bow,y_train)
y_preds = support.predict(X_test_bow)
acc3=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("SVM, Accuracy Score:" , acc3)
```

```
              precision    recall  f1-score   support

           0       0.46      0.27      0.34       290
           1       0.92      0.95      0.94      3832
           2       0.83      0.85      0.84       835

    accuracy                           0.89      4957
   macro avg       0.74      0.69      0.71      4957
weighted avg       0.88      0.89      0.89      4957

SVM, Accuracy Score: 0.8924752874722615

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_
base.py:985: ConvergenceWarning: Liblinear failed to converge, incre
ase the number of iterations.
  warnings.warn("Liblinear failed to converge, increase "
```

Ввод [28]:
```
objects = ('Logistic', 'RandomForest', 'Naive_bayes', 'SVM')
y_pos = np.arange(len(objects))
performance = [acc,acc1,acc2,acc3]
plt.bar(y_pos, performance, align='center', alpha=0.5)
plt.xticks(y_pos, objects)
plt.ylabel('Accuracy')
plt.title('Algorithm Comparision')
plt.show()
```



Ввод [29]:
```
#Using TFIDF with sentiment scores,doc2vec and enhanced features
def additional_features(tweet):

    syllables = textstat.syllable_count(tweet)
    num_chars = sum(len(w) for w in tweet)
    num_chars_total = len(tweet)
    num_words = len(tweet.split())
    # avg_syl = total syllables/ total words
    avg_syl = round(float((syllables+0.001))/float(num_words+0.001),4)
    num_unique_terms = len(set(tweet.split()))

    #  Flesch-Kincaid readability tests are readability tests
    #  designed to indicate how difficult a passage in English is to u
    # There are two tests, the Flesch Reading Ease, and the Flesch-Kir
    # A text with a comparatively high score on FRE test should have a
    # Reference - https://en.wikipedia.org/wiki/Flesch%E2%80%93Kincaid

    ###Modified FK grade, where avg words per sentence is : just num w
    FKRA = round(float(0.39 * float(num_words)/1.0) + float(11.8 * avg
    ##Modified FRE score, where sentence fixed to 1
    FRE = round(206.835 - 1.015*(float(num_words)/1.0) - (84.6*float(a

    add_features=[FKRA, FRE,syllables, avg_syl, num_chars, num_chars_t
                num_unique_terms]
    return add_features

def get_additonal_feature_array(tweets):
    features=[]
    for t in tweets:
        features.append(additional_features(t))
    return np.array(features)

fFeatures = get_additonal_feature_array(processed_tweets)
```

Ввод [30]:
```
tfidf_a = tfidf.toarray()
modelling_features_enhanced = np.concatenate([tfidf_a,final_features,c
modelling_features_enhanced.shape
```

Out[30]:
```
(24783, 6461)
```

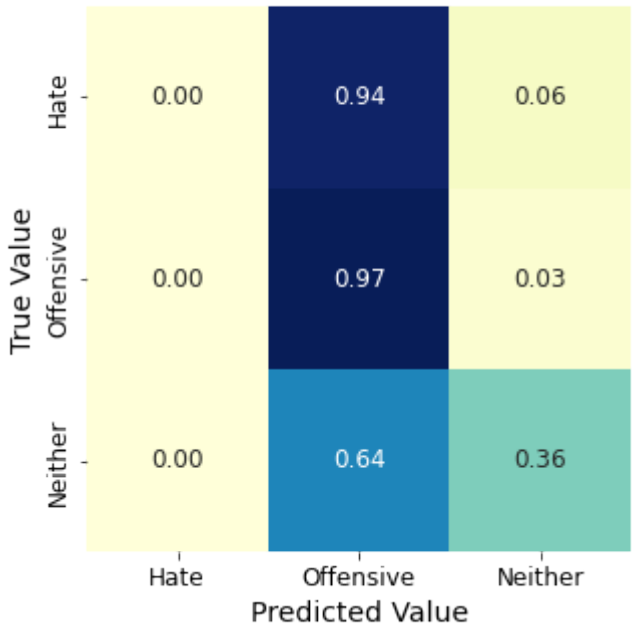
Запуск моделей. Использование TFIDF с оценками настроений, doc2vec и расширенными признаками.

Ввод [31]:

```
# Running the model Using TFIDF with enhanced features

X = panda.DataFrame(modelling_features_enhanced)
y = dataset['class'].astype(int)
X_train_features, X_test_features, y_train, y_test = train_test_split(

model = LogisticRegression().fit(X_train_features,y_train)
y_preds = model.predict(X_test_features)
report = classification_report( y_test, y_preds )
print(report)
acc=accuracy_score(y_test,y_preds)
print("Logistic Regression, Accuracy Score:" , acc)
```

```
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_logistic.py:763: ConvergenceWarning: lbfgs failed to conver
ge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as sh
own in:
    https://scikit-learn.org/stable/modules/preprocessing.html (http
s://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver option
s:
    https://scikit-learn.org/stable/modules/linear_model.html#logist
ic-regression (https://scikit-learn.org/stable/modules/linear_model.
html#logistic-regression)
  n_iter_i = _check_optimize_result(

              precision    recall  f1-score   support

           0       0.00      0.00      0.00       279
           1       0.83      0.97      0.89      3852
           2       0.70      0.36      0.48       826

    accuracy                           0.82      4957
   macro avg       0.51      0.45      0.46      4957
weighted avg       0.76      0.82      0.77      4957

Logistic Regression, Accuracy Score: 0.8150090780714142

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/metri
cs/_classification.py:1248: UndefinedMetricWarning: Precision and F-
score are ill-defined and being set to 0.0 in labels with no predict
ed samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/metri
cs/_classification.py:1248: UndefinedMetricWarning: Precision and F-
score are ill-defined and being set to 0.0 in labels with no predict
ed samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/metri
cs/_classification.py:1248: UndefinedMetricWarning: Precision and F-
score are ill-defined and being set to 0.0 in labels with no predict
ed samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

Ввод [32]:
```
#Confusion Matrix for TFIDF with additional features
from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y_test,y_preds)
matrix_proportions = np.zeros((3,3))
for i in range(0,3):
    matrix_proportions[i,:] = confusion_matrix[i,:]/float(confusion_ma
names=['Hate','Offensive','Neither']
confusion_df = panda.DataFrame(matrix_proportions, index=names,columns
plt.figure(figsize=(5,5))
seaborn.heatmap(confusion_df,annot=True,annot_kws={"size": 12},cmap='Y
plt.ylabel(r'True Value',fontsize=14)
plt.xlabel(r'Predicted Value',fontsize=14)
plt.tick_params(labelsize=12)
```



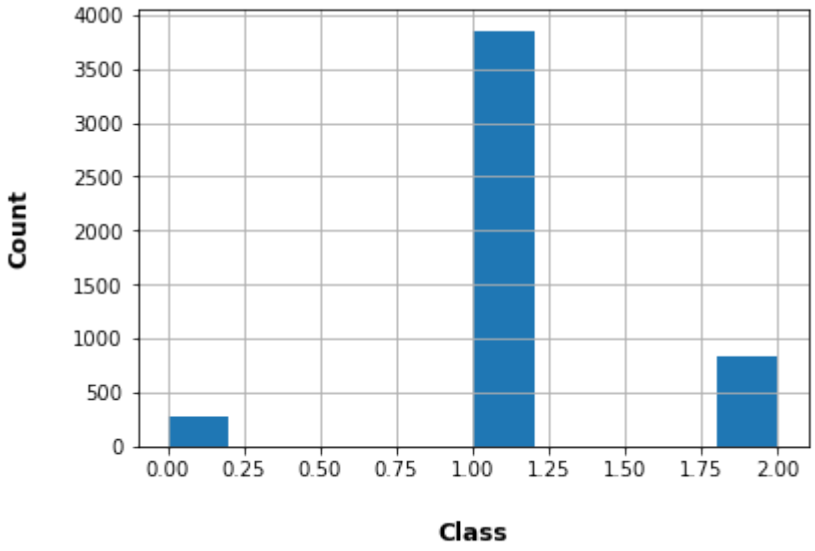
Ввод [33]:
```
# From the confusion matrix its clear that the model misclassifies 78%
# bar on the histogram for the predicted class and increase of bar for
```

Ввод [34]:
```
testing_index=list(X_test_features.index[0:10])
#print(testing_index)
print("Predicted Class:",y_preds[0:10])
print("Actual Class:",y_test.tolist()[0:10])
```

```
Predicted Class: [1 1 1 1 1 1 1 1 1 1]
Actual Class: [2, 1, 1, 0, 2, 1, 1, 1, 2, 2]
```

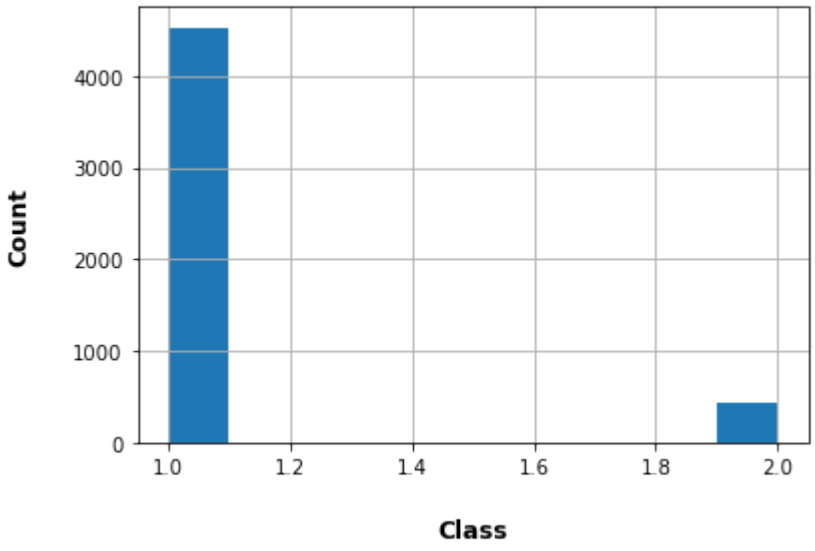
Ввод [35]:
```
# Histogram presenting the count of different classes- Actual
ax=y_test.hist()
ax.set_xlabel("Class", labelpad=20, weight='bold', size=12)
ax.set_ylabel("Count", labelpad=20, weight='bold', size=12)
```

Out[35]: Text(0, 0.5, 'Count')



Ввод [36]:
```
# Histogram presenting the count of different classes- Predicted
ax=panda.Series(y_preds).hist()
ax.set_xlabel("Class", labelpad=20, weight='bold', size=12)
ax.set_ylabel("Count", labelpad=20, weight='bold', size=12)
```

Out[36]: Text(0, 0.5, 'Count')



Ввод [37]:
```
X = panda.DataFrame(modelling_features_enhanced)
y = dataset['class'].astype(int)
X_train_features, X_test_features, y_train, y_test = train_test_split(
rf=RandomForestClassifier()
rf.fit(X_train_features,y_train)
y_preds = rf.predict(X_test_features)
acc1=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Random Forest, Accuracy Score:",acc1)
```

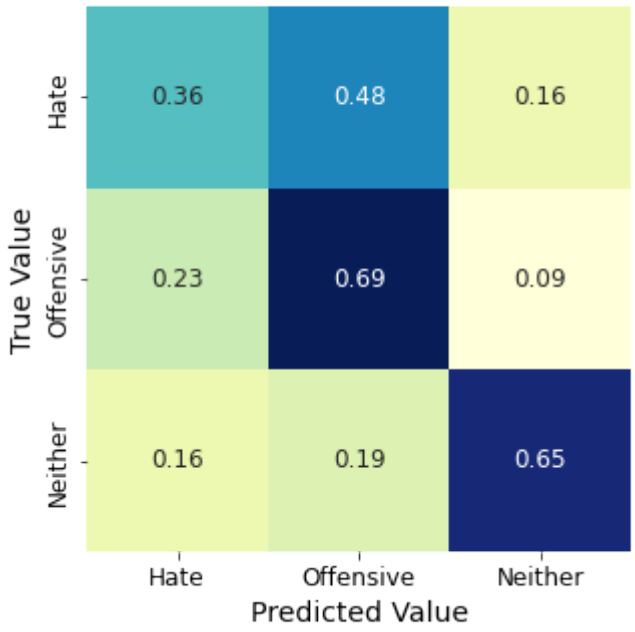
```
              precision    recall  f1-score   support

           0       0.48      0.05      0.09       279
           1       0.89      0.97      0.93      3852
           2       0.85      0.73      0.79       826

    accuracy                           0.88      4957
   macro avg       0.74      0.59      0.60      4957
weighted avg       0.86      0.88      0.86      4957

Random Forest, Accuracy Score: 0.8827920112971556
```

Ввод [38]:
```
X = panda.DataFrame(modelling_features_enhanced)
y = dataset['class'].astype(int)
X_train_features, X_test_features, y_train, y_test = train_test_split(
nb=GaussianNB()
nb.fit(X_train_features,y_train)
y_preds = nb.predict(X_test_features)
acc2=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Naive Bayes, Accuracy Score:",acc2)
```

```
              precision    recall  f1-score   support

           0       0.09      0.36      0.15       279
           1       0.90      0.69      0.78      3852
           2       0.59      0.65      0.62       826

    accuracy                           0.66      4957
   macro avg       0.53      0.57      0.51      4957
weighted avg       0.80      0.66      0.72      4957

Naive Bayes, Accuracy Score: 0.662497478313496
```

Ввод [39]:

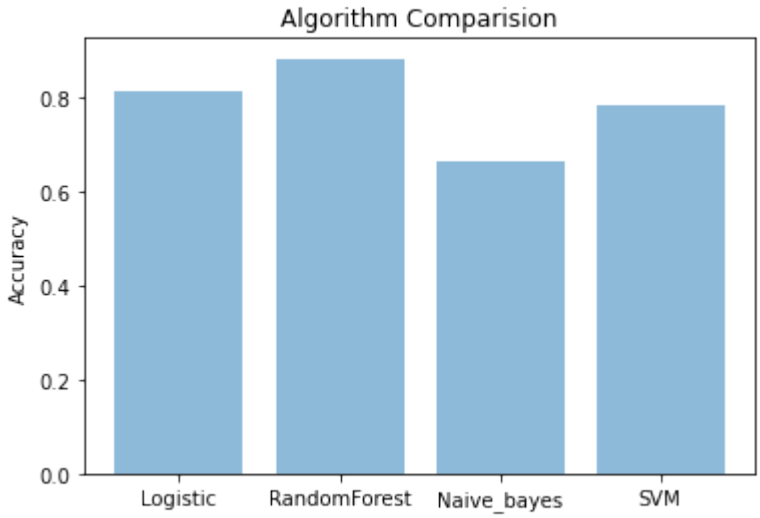
```
#Confusion Matrix for TFIDF with additional features
from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y_test,y_preds)
matrix_proportions = np.zeros((3,3))
for i in range(0,3):
    matrix_proportions[i,:] = confusion_matrix[i,:]/float(confusion_ma
names=['Hate','Offensive','Neither']
confusion_df = panda.DataFrame(matrix_proportions, index=names,columns
plt.figure(figsize=(5,5))
seaborn.heatmap(confusion_df,annot=True,annot_kws={"size": 12},cmap='Y
plt.ylabel(r'True Value',fontsize=14)
plt.xlabel(r'Predicted Value',fontsize=14)
plt.tick_params(labelsize=12)
```



Ввод [40]:
```
X = panda.DataFrame(modelling_features_enhanced)
y = dataset['class'].astype(int)
X_train_features, X_test_features, y_train, y_test_helo = train_test_s
support =LinearSVC(random_state=20)
support.fit(X_train_features,y_train)
y_preds = support.predict(X_test_features)
acc3=accuracy_score(y_test_helo,y_preds)
report = classification_report( y_test_helo, y_preds )
print(report)
print("SVM, Accuracy Score:" ,acc3 )
```

```
              precision    recall  f1-score   support

           0       0.17      0.59      0.26       279
           1       0.94      0.89      0.92      3852
           2       0.89      0.33      0.48       826

    accuracy                           0.78      4957
   macro avg       0.67      0.61      0.55      4957
weighted avg       0.89      0.78      0.81      4957

SVM, Accuracy Score: 0.7825297559007465

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_
base.py:985: ConvergenceWarning: Liblinear failed to converge, incre
ase the number of iterations.
  warnings.warn("Liblinear failed to converge, increase "
```

Ввод [41]:
```
objects = ('Logistic', 'RandomForest', 'Naive_bayes', 'SVM')
y_pos = np.arange(len(objects))
performance = [acc,acc1,acc2,acc3]
plt.bar(y_pos, performance, align='center', alpha=0.5)
plt.xticks(y_pos, objects)
plt.ylabel('Accuracy')
plt.title('Algorithm Comparision')
plt.show()
```



Сочетание различных признаков

Ввод [42]:
```
#f1,f3 and f4 combined

tfidf_a = tfidf.toarray()
modelling_features_one = np.concatenate([tfidf_a,doc2vec_df,fFeatures]
modelling_features_one.shape
```

Out[42]:
```
(24783, 6454)
```

Ввод [43]:
```
X = panda.DataFrame(modelling_features_one)
y = dataset['class'].astype(int)
X_train_features, X_test_features, y_train, y_test = train_test_split(
support =LinearSVC(random_state=20)
support.fit(X_train_features,y_train)
y_preds = support.predict(X_test_features)
acc3=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("SVM, Accuracy Score:" ,acc3 )
```

```
              precision    recall  f1-score   support

           0       0.52      0.04      0.07       279
           1       0.83      0.99      0.90      3852
           2       0.90      0.38      0.53       826

    accuracy                           0.83      4957
   macro avg       0.75      0.47      0.50      4957
weighted avg       0.83      0.83      0.80      4957

SVM, Accuracy Score: 0.8349808351825702

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_
base.py:985: ConvergenceWarning: Liblinear failed to converge, incre
ase the number of iterations.
  warnings.warn("Liblinear failed to converge, increase "
```

Ввод [44]:
```
#f1,f2 and f4 combined
tfidf_a = tfidf.toarray()
modelling_features_two = np.concatenate([tfidf_a,final_features,fFeatu
modelling_features_two.shape
```

Out[44]:
```
(24783, 6456)
```

Ввод [45]:
```
X = panda.DataFrame(modelling_features_two)
y = dataset['class'].astype(int)
X_train_features, X_test_features, y_train, y_test = train_test_split(
support =LinearSVC(random_state=20)
support.fit(X_train_features,y_train)
y_preds = support.predict(X_test_features)
acc3=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("SVM, Accuracy Score:" ,acc3 )
```

```
              precision    recall  f1-score   support

           0       0.59      0.09      0.15       279
           1       0.89      0.98      0.93      3852
           2       0.86      0.70      0.77       826

    accuracy                           0.88      4957
   macro avg       0.78      0.59      0.62      4957
weighted avg       0.86      0.88      0.86      4957

SVM, Accuracy Score: 0.8791607827314908

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_
base.py:985: ConvergenceWarning: Liblinear failed to converge, incre
ase the number of iterations.
  warnings.warn("Liblinear failed to converge, increase "
```

Ввод [46]:
```
#f2,f3 and f4 combined
modelling_features_three = np.concatenate([final_features,fFeatures],a
modelling_features_three.shape
```

Out[46]:
```
(24783, 15)
```

Ввод [47]:
```
X = panda.DataFrame(modelling_features_three)
y = dataset['class'].astype(int)
X_train_features, X_test_features, y_train, y_test = train_test_split(
support =LinearSVC(random_state=20)
support.fit(X_train_features,y_train)
y_preds = support.predict(X_test_features)
acc3=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("SVM, Accuracy Score:" ,acc3 )
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       279
           1       0.78      1.00      0.88      3852
           2       0.62      0.03      0.06       826

    accuracy                           0.78      4957
   macro avg       0.47      0.34      0.31      4957
weighted avg       0.71      0.78      0.69      4957

SVM, Accuracy Score: 0.7795037320960259

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_
base.py:985: ConvergenceWarning: Liblinear failed to converge, incre
ase the number of iterations.
  warnings.warn("Liblinear failed to converge, increase "
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/metri
cs/_classification.py:1248: UndefinedMetricWarning: Precision and F-
score are ill-defined and being set to 0.0 in labels with no predict
ed samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/metri
cs/_classification.py:1248: UndefinedMetricWarning: Precision and F-
score are ill-defined and being set to 0.0 in labels with no predict
ed samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/metri
cs/_classification.py:1248: UndefinedMetricWarning: Precision and F-
score are ill-defined and being set to 0.0 in labels with no predict
ed samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

Ввод [48]:
```
# the most important feature we found to be was the tf-idf scores whic
# Doc2vec columns are not found to be very significant in classificati
#its removed form the feature set. SVM's and RF's performance is hugel
```

Ввод [49]:
```
X = panda.DataFrame(modelling_features_two)
y = dataset['class'].astype(int)
X_train_features, X_test_features, y_train, y_test = train_test_split(
nb=GaussianNB()
nb.fit(X_train_features,y_train)
y_preds = nb.predict(X_test_features)
acc2=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Naive Bayes, Accuracy Score:",acc2)
```

```
              precision    recall  f1-score   support

           0       0.09      0.36      0.15       279
           1       0.90      0.69      0.78      3852
           2       0.59      0.65      0.62       826

    accuracy                           0.66      4957
   macro avg       0.53      0.57      0.51      4957
weighted avg       0.80      0.66      0.72      4957

Naive Bayes, Accuracy Score: 0.662497478313496
```

Ввод [50]:
```
# Naive Baiyes Classifier performs significantly better with feature s
#actually performs poor for Logistic Regression especially in predicti
```

Ввод [51]:
```
X = panda.DataFrame(modelling_features_two)
y = dataset['class'].astype(int)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state

model = LogisticRegression().fit(X_train,y_train)
y_preds = model.predict(X_test)
report = classification_report( y_test, y_preds )
print(report)
acc=accuracy_score(y_test,y_preds)
print("Logistic Regression, Accuracy Score:" , acc)
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       279
           1       0.81      0.96      0.88      3852
           2       0.59      0.29      0.39       826

    accuracy                           0.80      4957
   macro avg       0.47      0.42      0.42      4957
weighted avg       0.73      0.80      0.75      4957

Logistic Regression, Accuracy Score: 0.795239055880573

/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_logistic.py:763: ConvergenceWarning: lbfgs failed to conver
ge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as sh
own in:
    https://scikit-learn.org/stable/modules/preprocessing.html (http
s://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver option
s:
    https://scikit-learn.org/stable/modules/linear_model.html#logist
ic-regression (https://scikit-learn.org/stable/modules/linear_model.
html#logistic-regression)
  n_iter_i = _check_optimize_result(
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/metri
cs/_classification.py:1248: UndefinedMetricWarning: Precision and F-
score are ill-defined and being set to 0.0 in labels with no predict
ed samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/metri
cs/_classification.py:1248: UndefinedMetricWarning: Precision and F-
score are ill-defined and being set to 0.0 in labels with no predict
ed samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/Users/vadim/opt/anaconda3/lib/python3.9/site-packages/sklearn/metri
cs/_classification.py:1248: UndefinedMetricWarning: Precision and F-
score are ill-defined and being set to 0.0 in labels with no predict
ed samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

Ввод [52]:
```
X = panda.DataFrame(modelling_features_three)
y = dataset['class'].astype(int)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
rf=RandomForestClassifier()
rf.fit(X_train,y_train)
y_preds = rf.predict(X_test)
acc1=accuracy_score(y_test,y_preds)
report = classification_report( y_test, y_preds )
print(report)
print("Random Forest, Accuracy Score:",acc1)
```

```
              precision    recall  f1-score   support

           0       0.12      0.02      0.04       279
           1       0.82      0.93      0.87      3852
           2       0.51      0.32      0.39       826

    accuracy                           0.78      4957
   macro avg       0.48      0.43      0.43      4957
weighted avg       0.73      0.78      0.75      4957

Random Forest, Accuracy Score: 0.7797054670163406
```

Ввод [53]:
```
#Confusion Matrix for TFIDF with additional features
from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y_test,y_preds)
matrix_proportions = np.zeros((3,3))
for i in range(0,3):
    matrix_proportions[i,:] = confusion_matrix[i,:]/float(confusion_ma
names=['Hate','Offensive','Neither']
confusion_df = panda.DataFrame(matrix_proportions, index=names,columns
plt.figure(figsize=(5,5))
seaborn.heatmap(confusion_df,annot=True,annot_kws={"size": 12},cmap='Y
plt.ylabel(r'True Value',fontsize=14)
plt.xlabel(r'Predicted Value',fontsize=14)
plt.tick_params(labelsize=12)
```