

**Липецкий государственный технический университет**

**Факультет автоматизации и информатики**

**Кафедра автоматизированных систем управления**

**ЛАБОРАТОРНАЯ РАБОТА №3**

**по дисциплине «Прикладные интеллектуальные системы и экспертные  
системы»**

**Предварительная обработка текстовых данных**

Студент

Лобов М.Ю.

Группа М-ИАП-22

Руководитель

Кургасов В.В.

Липецк 2022 г.

## Задание кафедры

### Вариант 10

- 1) В среде Jupiter Notebook создать новый ноутбук (Notebook);
- 2) Импортировать необходимые для работы библиотеки и модули;
- 3) Загрузить обучающую и экзаменационную выборку в соответствие с вариантом;
- 4) Вывести на экран по одному-два документа каждого класса;
- 5) Применить стемминг, записав обработанные выборки (тестовую и обучающую) в новые переменные;
- 6) Провести векторизацию выборки:
  - a. Векторизовать обучающую и тестовую выборки простым подсчетом слов (CountVectorizer) и значением `max_features = 10000`
  - b. Вывести и проанализировать первые 20 наиболее частотных слов всей выборки и каждого класса по-отдельности.
  - c. Применить процедуру отсечения стоп-слов и повторить пункт b.
  - d. Провести пункты a – c для обучающей и тестовой выборки, для которой проведена процедура стемминга.
  - e. Векторизовать выборки с помощью TfidfTransformer (с использованием TF и TF-IDF взвешиваний) и повторить пункты b-d.
- 7) По результатам пункта 6 заполнить таблицы наиболее частотными терминами обучающей выборки и каждого класса по отдельности.

Всего должно получиться по 4 таблицы для выборки, к которой применялась операция стемминга и 4 таблицы для выборки, к которой операция стемминга не применялась
- 8) Используя конвейер (Pipeline) реализовать модель Наивного Байесовского классификатора и выявить на основе показателей качества (значения полноты, точности, f1-меры и аккуратности), какая предварительная обработка данных обеспечит наилучшие результаты классификации. Должны быть исследованы следующие характеристики:
  - Наличие - отсутствие стемминга

- Отсечение – не отсечение стоп-слов
- Количество информативных терминов (max\_features)
- Взвешивание: Count, TF, TF-IDF

9) По каждому пункту работы занести в отчет программный код и результат вывода.

10) По результатам классификации занести в отчет выводы о наиболее подходящей предварительной обработке данных (наличие стемминга, взвешивание терминов, стоп-слова, количество информативных терминов).

## Ход работы

Импортируем необходимые для работы библиотеки и модули.

- pandas — программная библиотека на языке Python для обработки и анализа данных;

- numPy (сокращенно от Numerical Python)— библиотека с открытым исходным кодом для языка программирования Python. Возможности: поддержка многомерных массивов (включая матрицы); поддержка высокоуровневых математических функций, предназначенных для работы с многомерными массивами;

- matplotlib — библиотека на языке программирования Python для визуализации данных двумерной и трёхмерной графикой;

- библиотека NLTK — пакет библиотек и программ для символьной и статистической обработки естественного языка, написанных на языке программирования Python. Содержит графические представления и примеры данных;

- itertools стандартизирует основной набор быстрых эффективных по памяти инструментов, которые полезны сами по себе или в связке с другими инструментами;

scikit-learn – это библиотека Python, которая является одной из самых полезных библиотек Python для машинного обучения. Она включает все алгоритмы и инструменты, которые нужны для задач классификации, регрессии и кластеризации. Она также включает все методы оценки производительности модели машинного обучения.

1) Импортировать необходимые для работы библиотеки и модули;

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.pipeline import Pipeline
from sklearn.naive_bayes import MultinomialNB
from nltk.stem import *
from nltk import word_tokenize
import itertools
import nltk
```

Рисунок 1 – Импорт библиотек

2) Загрузить обучающую и экзаменационную выборку в соответствии с вариантом;

```
categories = ['comp.os.ms-windows.misc', 'comp.sys.mac.hardware', 'sci.space']
remove = ['headers', 'footers', 'quotes']
twenty_train_full = fetch_20newsgroups(subset='train', shuffle=True, random_state=42, categories=categories, remove=remove)
twenty_test_full = fetch_20newsgroups(subset='test', shuffle=True, random_state=42, categories=categories, remove=remove)

twenty_train_full = twenty_train_full.data
twenty_test_full = twenty_test_full.data
```

Рисунок 2 – Загрузка выборки

3) Вывести на экран по одному-два документа каждого класса;

```
'\nIf I rember correctly, Lotus Notes gives u this possiblity, among other things...'
```

Рисунок 3 – Документ для класса comp.os.ms-windows.misc

```
"I read in a recent Tidbits(171-2?) about the possibility of putting\na 68030 in a PB100. I am interested in doing so, but woul
d like\nto know more about it. Does it involve just replacing the 68000 that\nis on the daughterboard, or does it involve getti
ng a new daughter-\nboard. Also, would the 68030 be able to run QT with the PB100's\nscreen(not pretty I know, but possible?) A
nd of course, what would\nthe damage be ($). Any info would be appreciated.\nThanks in advance. Jay Fogel\n\n"
```

Рисунок 4 – Документ для класса comp.sys.mac.hardware

```
"\nI'd guess this was a garbled report of the NERVA effort to develop a\nsolid-core fission rocket (the most mundane type of nu
clear rocket).\nThat was the only advanced-propulsion project that was done on a large\nenough scale to be likely to attract ne
ws attention. It *could* be any\nnumber of things -- the description given is awfully vague -- but I'd\nput a small bet on NER
VA."
```

Рисунок 5 – Документ для класса sci.space

4) Применить стемминг, записав обработанные выборки (тестовую и обучающую) в новые переменные;

## Стемминг

```
def stemming(data):
    porter_stemmer = PorterStemmer()
    stem = []
    for text in data:
        nltk_tokens = word_tokenize(text)
        line = ''
        for word in nltk_tokens:
            line += ' ' + porter_stemmer.stem(word)
        stem.append(line)
    return stem

stem_train = dict()
stem_test = dict()
for category in categories:
    stem_train[category] = stemming(twenty_train[category])
    stem_test[category] = stemming(twenty_test[category])

stem_train['full'] = stemming(twenty_train['full'])
stem_test['full'] = stemming(twenty_test['full'])
```

Рисунок 6 – Процедура стемминга

5) Провести векторизацию выборки:

- а. Векторизовать обучающую и тестовую выборки простым подсчетом слов (CountVectorizer) и значением max\_features = 10000
- б. Вывести и проанализировать первые 20 наиболее частотных слов всей выборки и каждого класса по-отдельности.
- с. Применить процедуру отсечения стоп-слов и повторить пункт б.
- д. Провести пункты а – с для обучающей и тестовой выборки, для которой проведена процедура стемминга.
- е. Векторизовать выборки с помощью TfidfTransformer (с использованием TF и TF-IDF взвешиваний) и повторить пункты б-д.

	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами
0	('ax', 62375)	('ax', 62375)	('window', 66.59871872142205)	('the', 155.2840457611457)	('window', 27.7743990392879)	('the', 62.8731213090935)
1	('max', 4474)	('max', 4474)	('use', 47.1996176649944)	('to', 89.529730762218)	('file', 23.014545190736936)	('to', 39.75087544648268)
2	('g9v', 1166)	('the', 2706)	('thi', 42.2362695472442)	('it', 66.8881447320785)	('use', 20.948949127256274)	('it', 31.791441260216537)
3	('b8f', 1111)	('to', 1492)	('file', 39.66721443662778)	('and', 60.008986802890746)	('thi', 19.333668978500036)	('and', 28.345851678567815)
4	('a86', 916)	('g9v', 1166)	('driver', 25.014820100221286)	('is', 58.098861808314446)	('driver', 16.658334679831132)	('is', 27.66351857674044)
5	('pl', 822)	('b8f', 1111)	('ani', 24.570130510182207)	('of', 48.59804818078258)	('problem', 14.398781397188685)	('of', 24.59180191137564)
6	('145', 760)	('and', 1095)	('problem', 23.584679326612637)	('for', 46.772694833615006)	('ani', 13.913565351656242)	('for', 24.341326441352177)
7	('window', 678)	('it', 1036)	('thank', 20.879233560232695)	('window', 44.57705362012204)	('thank', 12.810152458161534)	('window', 24.14123661660397)
8	('1d9', 672)	('is', 1016)	('know', 20.49223495470095)	('in', 43.53441963726538)	('know', 12.591684755501632)	('that', 23.0145871901136)
9	('34u', 549)	('of', 922)	('program', 18.872863422521135)	('that', 40.53934251036627)	('program', 12.440190267580325)	('you', 22.95243261156568)
10	('1t', 510)	('a86', 916)	('doe', 18.549998342055865)	('have', 38.175464095502086)	('doe', 11.649924933715221)	('in', 22.034349755151275)
11	('0t', 505)	('pl', 822)	('wa', 18.34662847615492)	('you', 37.76894193852423)	('wa', 11.135207366569679)	('have', 20.779283873958605)
12	('use', 497)	('145', 760)	('like', 17.754394251250936)	('with', 31.91691929013105)	('card', 10.960310169199825)	('file', 19.90156507366835)
13	('bhj', 456)	('for', 734)	('anyon', 16.543898441484213)	('use', 31.16930257432249)	('anyon', 10.928283494060256)	('do', 18.678417177036266)
14	('75u', 447)	('in', 706)	('run', 16.482385941440164)	('do', 30.67100685109011)	('run', 10.839808257779888)	('with', 18.49135204961512)
15	('thi', 445)	('window', 678)	('tri', 15.788658748857234)	('on', 28.781299598201535)	('like', 10.702662894952912)	('use', 18.31677978035442)
16	('3t', 441)	('1d9', 672)	('ha', 15.684886526753514)	('thi', 27.712366117794982)	('font', 10.242377382092688)	('on', 17.294877702936663)
17	('giz', 433)	('that', 664)	('just', 15.416413654153528)	('file', 26.39273195853486)	('tri', 10.170342301448702)	('thi', 16.844067049093002)
18	('file', 431)	('have', 621)	('card', 15.208869340153491)	('but', 23.320379373207913)	('ax', 9.99821180806796)	('or', 15.084964497390342)
19	('2di', 415)	('you', 608)	('work', 13.854614071828784)	('be', 23.183464093544489)	('ha', 9.669851771821174)	('but', 15.030811048629891)

Рисунок 7 – Со стеммингом для comp.os.ms-windows.misc

	Count		TF		TF-IDF	
	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами
0	{ax', 62376}	{ax', 62376}	{thf', 138.10306478897867}	{the', 548.1743253810621}	{thf', 56.73681753578598}	{the', 204.6005535655787}
1	{max', 4503}	{the', 12312}	{use', 98.23332155901028}	{to', 276.96931753021213}	{use', 45.184815902966744}	{to', 114.41787381417121}
2	{thi', 1664}	{to', 5895}	{wa', 75.630124337131607}	{and', 198.72739393240043}	{window', 42.16659729511384}	{and', 88.8650442811465}
3	{use', 1200}	{and', 4947}	{window', 68.6937664827718}	{of', 196.24874956700597}	{wa', 38.875997564701656}	{of', 87.95626190081434}
4	{g9v', 1166}	{of', 4862}	{anl', 61.36597025832105}	{anl', 184.02041748529458}	{anl', 33.750066281894185}	{ir', 84.70038233452496}
5	{h8f', 1111}	{mac', 4503}	{thf', 55.4789152554171}	{s', 179.95921788235471}	{problem', 33.28271430256718}	{s', 81.1495043459019}
6	{space', 579}	{s', 3551}	{know', 54.7815601833294}	{in', 143.13965307492573}	{file', 32.515721740952706}	{thaf', 68.29320552340506}
7	{wa', 932}	{in', 3282}	{ha', 54.464562877002756}	{that', 136.50582646570444}	{know', 30.98397722305867}	{in', 67.24770380198822}
8	{a86', 916}	{f', 3227}	{problem', 54.213052823388205}	{for', 133.4656393544668}	{like', 30.11472167892794}	{for', 65.97393304649562}
9	{p', 823}	{to', 2749}	{toe', 51.50648489891999}	{toe', 106.65519885198540}	{thank', 29.9020929870311}	{you', 62.47148571745875}
10	{145', 765}	{thaf', 2508}	{ust', 50.14437066112852}	{have', 100.31115988366414}	{doe', 29.804680032835154}	{have', 54.18662105427451}
11	{window', 728}	{on', 2040}	{thank', 48.072721221708754}	{on', 95.32577590429649}	{ha', 28.795530268882734}	{on', 51.84809182982481}
12	{ha', 702}	{you', 1861}	{lie', 44.973189861840094}	{thf', 87.30645298083387}	{just', 27.94434638767364}	{just', 50.00600431598561}
13	{1d9', 672}	{have', 1805}	{space', 43.66483240292382}	{de', 86.93506711481417}	{space', 27.9297088480323}	{be', 49.5275480784525}
14	{lie', 577}	{be', 1763}	{worl', 43.5406289749797}	{with', 86.55643026316217}	{anyon', 26.75527873289231}	{with', 48.22477195588755}
15	{problem', 565}	{with', 1688}	{anyon', 41.38927802867616}	{do', 70.80833416817352}	{work', 25.66012235171234}	{of', 43.29225113645906}
16	{ani', 562}	{thf', 1664}	{think', 36.39719640620317}	{thf', 68.36640110584909}	{mac', 24.797224560247642}	{if', 41.679620693090804}
17	{34u', 549}	{are', 1446}	{onl', 35.27262292363869}	{but', 65.8281173081393}	{drive', 24.46236365808067}	{or', 40.93056320170661}
18	{lie', 512}	{or', 1273}	{mac', 35.04023157432011}	{or', 65.113391913482}	{card', 24.08998624544712}	{use', 40.308329901206154}
19	{t', 510}	{do', 1255}	{need', 34.03909816075486}	{not', 64.5468684035843}	{driver', 23.39644132075989}	{are', 40.17813806642236}

Рисунок 8 – Со стеммингом для всех категорий

	Count		TF		TF-IDF	
	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами
0	{thi', 482}	{the', 3289}	{thi', 47.14200127760944}	{the', 183.8348453544744}	{thi', 20.687077395881932}	{the', 69.34081073969716}
1	{mac', 386}	{to', 1544}	{mac', 30.958336115718655}	{to', 86.39888860773068}	{mac', 16.07410546704644}	{to', 36.44340890749818}
2	{use', 355}	{and', 1248}	{use', 30.78627924724098}	{and', 62.5288229051547}	{drive', 15.631605135842705}	{and', 29.15557723850467}
3	{drive', 274}	{is', 987}	{drive', 24.44747992042716}	{is', 61.773139041774776}	{use', 15.137258101434772}	{is', 28.59994889521901}
4	{appl', 254}	{problem', 23.03129106310386}	{appl', 58.89743933981437}	{problem', 13.88689226614134}	{problem', 13.88689226614134}	{appl', 27.780422615064477}
5	{problem', 235}	{it', 945}	{anl', 22.93196182260546}	{s', 53.70680505431753}	{appl', 12.635867145444662}	{of', 24.991042300024652}
6	{ha', 226}	{in', 748}	{doe', 21.992010394173704}	{for', 45.93271981929874}	{anl', 12.56865218035183}	{for', 27.498163723870398}
7	{doe', 201}	{for', 731}	{appl', 21.34228625042482}	{that', 44.323863068096344}	{doe', 12.261278890379222}	{that', 22.35799477130168}
8	{onl', 190}	{that', 706}	{in', 21.329996529015936}	{in', 40.5543546493516746}	{ha', 11.459184649435167}	{you', 21.09228139179046}
9	{worl', 180}	{with', 622}	{know', 19.858882535660577}	{with', 36.3446216023387}	{work', 11.402835023958062}	{in', 20.482021711471553}
10	{card', 177}	{have', 576}	{work', 19.414003791400862}	{have', 35.21629329083215}	{know', 11.352623929083215}	{have', 19.149582662905853}
11	{know', 176}	{on', 533}	{thank', 18.429670677863538}	{you', 34.757032433558514}	{card', 11.313024540308914}	{with', 18.917034096273834}
12	{like', 165}	{you', 531}	{anyon', 16.3425581826923}	{on', 31.5045795866288173}	{thank', 10.77203066730954}	{thi', 18.076988757875874}
13	{bit', 160}	{thi', 482}	{ust', 16.308823631445807}	{thi', 30.28152296523649}	{simn', 10.528421217184949}	{on', 17.53793962536255}
14	{wa', 158}	{like', 15.278148807319253}	{like', 26.26882620694144}	{anyon', 10.293843878915254}	{be', 15.86849268848105}	{be', 15.86849268848105}
15	{onl', 154}	{if', 402}	{is', 20.3051677131465}	{if', 25.844097572654466}	{just', 9.946928928253758}	{if', 15.52215216273105}
16	{just', 148}	{mac', 386}	{wa', 13.73358215910845}	{or', 22.244141381577755}	{monitor', 9.82231757504746}	{my', 14.242859859232457}
17	{monitor', 143}	{not', 373}	{simn', 13.165741723623679}	{but', 22.195929084431974}	{like', 9.203305905175593}	{mac', 14.226716618680284}
18	{scsi', 142}	{but', 361}	{onl', 12.965228185523037}	{not', 22.168038765484994}	{wa', 8.640107070917702}	{or', 14.00451848257456}
19	{thank', 133}	{or', 358}	{monitor', 12.781200525134151}	{my', 21.711330254341522}	{need', 8.296488616621357}	{can', 13.916095496829769}

Рисунок 9 – Со стеммингом для comp.sys.mac.hardware

	Count		TF		TF-IDF	
	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами
0	{space', 921}	{the', 6317}	{thi', 46.24383462365558}	{the', 205.6331861996026}	{thi', 19.250018399958233}	{the', 80.46560023187664}
1	{thi', 737}	{to', 2968}	{wa', 42.35033617115941}	{to', 99.024618219301596}	{wa', 19.018261819301596}	{to', 42.98303173084809}
2	{wa', 560}	{to', 2859}	{space', 38.314795216401656}	{of', 92.1045609345099}	{space', 18.96312393356916}	{of', 40.213615471466156}
3	{orbit', 402}	{and', 2604}	{ike', 21.32602740594018}	{and', 74.29963562705534}	{wie', 11.417412865436105}	{and', 34.50656454861889}
4	{launch', 399}	{in', 1828}	{use', 19.583788957467437}	{is', 58.48098101402759}	{orbit', 10.752095080469597}	{is', 28.72931150815752}
5	{nas', 376}	{is', 1548}	{orbit', 17.642704636781843}	{in', 57.7904656442908}	{nas', 10.202157828935743}	{it', 27.635476935422368}
6	{use', 348}	{for', 1284}	{ust', 17.42980030032279}	{ir', 57.5086078079843}	{use', 10.14336058141241}	{in', 27.23025284705928}
7	{ha', 293}	{it', 1246}	{ha', 16.387781266087696}	{that', 50.618482373960745}	{launch', 9.75989657838095}	{that', 25.618184917105793}
8	{satellit', 285}	{that', 1138}	{nasa', 15.79386265970003}	{for', 39.58187287917161}	{just', 9.66647453583248}	{for', 21.859186576099145}
9	{year', 277}	{on', 1027}	{thnk', 15.294592433574094}	{be', 36.73589280178944}	{thnk', 9.387974053438247}	{you', 20.92307537233606}
10	{like', 241}	{be', 853}	{launch', 14.702313959720486}	{34', 24.06916497226166}	{ha', 8.88244810632814}	{be', 20.183909189179456}
11	{time', 238}	{space', 921}	{year', 14.462628176540425}	{you', 33.51454216624072}	{year', 8.701465839766646}	{on', 18.85732299357959}
12	{earth', 224}	{are', 761}	{know', 13.482651859311023}	{thi', 28.57095695385257}	{know', 8.30813034998602}	{thi', 16.837737987817718}
13	{mission', 224}	{time', 737}	{time', 13.086437659457207}	{wa', 25.964875205330394}	{anl', 8.238859337919525}	{wa', 16.613913442873088}
14	{data', 216}	{you', 722}	{anl', 12.932964648791371}	{are', 25.940516282901832}	{time', 7.834224602664401}	{space', 16.547805994909627}
15	{program', 203}	{as', 640}	{thing', 12.176512511639096}	{have', 24.0476148836235273}	{thing', 7.688495118839462}	{space', 15.970837198521432}
16	{new', 199}	{have', 608}	{onl', 10.70497043029426}	{is', 45.9496122625269}	{mo', 5.559231568767245}	{is', 15.38718886893571}
17	{just', 189}	{with', 561}	{peopl', 10.63262300585063}	{space', 23.288235419639395}	{peopl', 7.26246264765531}	{ax', 14.966559191391287}
18	{shuttl', 187}	{wa', 560}	{moon', 10.3475449918885}	{not', 20.467904547787573}	{cost', 7.0273161010948225}	{they', 13.917557893157962}
19	{lunar', 182}	{at', 556}	{new', 10.06195436363668}	{if', 20.22853699531048}	{post', 6.864566246993111}	{would', 13.767253653948917}

Рисунок 10 – Со стеммингом для sci.space

	Count		TF		TF-IDF	
	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами
0	{ax', 62375}	{ax', 62375}	{windows', 65.12677424528061}	{the', 158.2816792256827}	{windows', 25.668345694833473}	{the', 61.92055657553077}
1	{max', 4474}	{mac', 4474}	{file', 25.619899459434357}	{to', 91.34923514171841}	{file', 15.233040295492764}	{to', 39.096486925244726}
2	{g9v', 1166}	{the', 2707}	{use', 23.011239326113046}	{f', 64.92223978616043}	{use', 12.424463518980232}	{f', 29.9474152417449004}
3	{h8f', 1111}	{to', 1492}	{thanks', 20.167493242987913}	{and', 61.22729491975415}	{dos', 12.01078552067887}	{and', 27.98124908599975}
4	{a86', 916}	{g9v', 1166}	{know', 19.760975448960327}	{s', 58.52377963455573}	{thanks', 11.987323570257036}	{s', 26.961961459344305}
5	{p', 822}	{h8f', 1111}	{dos', 19.18943788824051}	{of', 49.58087175547401}	{file', 11.622886461440886}	{of', 24.360508095827914}
6	{145', 755}	{and', 1095}	{problem', 17.840564351149794}	{for', 47.66897254736519}	{know', 11.61233524322242}	{for', 23.9028683550302}
7	{1d9', 672}	{is', 1026}	{like', 17.478649020274673}	{in', 44.24575940818995}	{problem', 11.177799913186696}	{you', 22.663539237769164}
8	{windows', 645}	{it', 984}	{files', 17.415217264036215}	{windows', 41.62754715045388}	{does', 10.363359783996652}	{that', 22.0823599949338}
9	{34u', 549}	{of', 922}	{using', 16.885272152069508}	{that', 40.688971504168364}	{driver', 10.014071170777417}	{windows', 22.077262316465564}
10	{t', 510}	{a86', 916}	{ust', 16.629428347832466}	{you', 38.47517797655345}	{ax', 9.99846213598706}	{in', 21.64364078984249}
11	{t', 505}	{pl', 822}	{does', 16.345822078145297}	{have', 35.5582233806141}	{like', 9.9716639590625}	{have', 19.168864580883335}
12	{h1f', 456}	{145', 755}	{program', 14.521332207098672}	{with', 32.5022257647117}	{program', 9.876911147890741}	{with', 18.226059282668427}
13	{75u', 447}	{for', 734}	{driver', 14.096406109052158}	{on', 29.15731167062002}	{drivers', 9.805246839149405}	{on', 17.02247693112356}
14	{3t', 441}	{in', 706}	{card', 13.844337412647256}	{can', 28.202740235914614}	{using', 9.626617222998261}	{this', 16.573368631746774}
15	{giz', 433}	{1d9', 672}	{drivers', 13.339255256465048}	{thaf', 24.50921133933111}	{can', 9.565966991788846}	{can', 15.403848131393976}
16	{2di', 414}	{that', 661}	{edu', 12.621093542464694}	{but', 23.788334133205116}	{just', 9.484966219641585}	{or', 14.880119385615574}
17	{cx', 373}	{windows', 645}	{mail', 12.015744903545047}	{or', 23.20499734282228}	{ftp', 8.668976012536401}	{be', 14.123832440169965}
18	{wm', 358}	{you', 608}	{ftp', 11.905531253377935}	{be', 22.646360238172576}	{mail', 8.53328135106901}	{be', 14.123832440169965}
19	{2tn', 353}	{as', 569}	{version', 11.896496816767188}	{if', 22.159888509423602}	{edu', 8.422393328583944}	{if', 14.041383269392984}

Рисунок 11 – Без стемминга для comp.os.ms-windows.misc

	Count		TF		TF-IDF	
	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами
0	('ax', 62376)	('ax', 62376)	('windows', 66.47067130148628)	('the', 560.2990851493455)	('windows', 38.684336218976384)	('the', 204.23336925665132)
1	('max', 4503)	('the', 12314)	('like', 56.66864253751202)	('to', 282.93647957843035)	('space', 29.863491700078487)	('to', 113.93823671923242)
2	('g9v', 1166)	('to', 5892)	('just', 54.427395911887096)	('and', 203.00011396142446)	('know', 29.245701391288925)	('and', 88.66984133734529)
3	('b8f', 1111)	('and', 4947)	('know', 54.18196579138407)	('of', 200.4134781034294)	('like', 29.085553845705963)	('of', 88.06562729215649)
4	('space', 1041)	('of', 4862)	('space', 50.979997110835185)	('is', 180.27189043373522)	('thanks', 28.255571244979407)	('it', 80.628763714549)
5	('a86', 916)	('max', 4503)	('use', 50.69620968080782)	('it', 177.05107499708862)	('just', 28.101746163289064)	('is', 79.62738404837968)
6	('p', 823)	('is', 3510)	('thanks', 47.22337876929786)	('in', 146.04043237253202)	('use', 67.111754631035474)	('that', 67.5292300653039)
7	('145', 760)	('in', 3282)	('does', 45.21380407203235)	('that', 138.70687990773726)	('does', 26.22344185370232)	('in', 67.13791727139187)
8	('windows', 679)	('it', 2940)	('problem', 40.78195736125016)	('for', 136.23867026187293)	('problem', 25.066612038294668)	('for', 65.61593301519157)
9	('1d9', 672)	('for', 2749)	('don', 40.11764797670633)	('you', 109.11164501986845)	('mac', 22.690856205841037)	('you', 62.19659439802179)
10	('34u', 549)	('that', 2505)	('think', 34.84536767051748)	('on', 97.00314572919189)	('don', 22.593218488438428)	('on', 51.59643618520398)
11	('like', 542)	('on', 2034)	('mac', 33.396625802027636)	('have', 94.41614392542543)	('file', 21.059332563344146)	('have', 50.726630803070385)
12	('use', 533)	('you', 1861)	('new', 30.718554523806983)	('this', 89.13254839163734)	('think', 20.824850341263133)	('this', 49.768937848909516)
13	('1t', 510)	('with', 1688)	('using', 28.40691828236921)	('with', 88.46031232354682)	('drive', 19.302621889781687)	('with', 48.04299309663674)
14	('0t', 505)	('this', 1664)	('file', 28.378512196960422)	('be', 82.81178596037823)	('card', 19.246885693506634)	('card', 46.69799226437908)
15	('just', 485)	('have', 1657)	('need', 27.511769065852262)	('if', 69.76254374284767)	('new', 18.849785302153744)	('if', 41.41047459654504)
16	('bhj', 456)	('be', 1633)	('use', 26.537173489532382)	('if', 67.23135433673392)	('apple', 18.108901324719188)	('or', 40.69804014579347)
17	('75u', 447)	('as', 1449)	('work', 26.200993546303142)	('or', 67.01097248210164)	('need', 17.927499523291296)	('can', 39.72428352322031)
18	('3t', 441)	('are', 1409)	('time', 26.141505849082485)	('can', 64.15300523041691)	('mail', 17.56008683056373)	('but', 39.48005437945485)
19	('giz', 433)	('or', 1269)	('drive', 26.067445078218135)	('not', 63.63091187140009)	('edu', 17.42335656606333)	('are', 39.364547995046685)

Рисунок 12 – Без стемминга для всех категорий

	Count		TF		TF-IDF	
	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами
0	('mac', 327)	('the', 3290)	('mac', 28.632597423926)	('the', 186.28530396848485)	('mac', 14.463676969657609)	('the', 67.7099572758338)
1	('apple', 266)	('to', 1544)	('apple', 23.31436692334048)	('to', 87.63175781280674)	('apple', 12.49625883726875)	('to', 35.52533678550535)
2	('drive', 211)	('and', 1248)	('drive', 20.25252957820821)	('and', 63.43610028855522)	('drive', 12.797076286363978)	('and', 28.46481041561887)
3	('use', 173)	('of', 972)	('know', 19.615439671551663)	('is', 61.38979015265487)	('know', 10.612354654633103)	('is', 26.72585563991222)
4	('problem', 171)	('is', 966)	('does', 18.80073364276716)	('is', 56.13666065956749)	('problem', 10.45705597493345)	('it', 26.677147131798005)
5	('like', 163)	('it', 873)	('thanks', 18.28629277422955)	('of', 54.40634306325833)	('does', 10.398313064226242)	('of', 24.41548674235528)
6	('know', 162)	('in', 748)	('problem', 17.503374993042566)	('for', 46.44910058872344)	('thanks', 10.17190482390627)	('for', 21.8712481281960405)
7	('does', 160)	('for', 731)	('use', 17.495211940199702)	('that', 44.97671965636135)	('use', 9.912672464149352)	('that', 21.8188163408988)
8	('bit', 150)	('that', 706)	('just', 17.27135715291802)	('in', 41.11990207631689)	('just', 9.69655879350543)	('you', 20.60544336325283)
9	('just', 148)	('with', 622)	('like', 15.7415460902871)	('with', 36.88184698596226)	('like', 8.780429835603503)	('in', 19.9831715733427)
10	('sci', 142)	('have', 534)	('don', 13.5636832379864)	('you', 35.263026862745465)	('new', 8.039273898146508)	('with', 18.45458904473647)
11	('don', 123)	('on', 532)	('new', 12.781414899338923)	('have', 33.248682081703265)	('card', 8.023194833144826)	('have', 17.817241001001778)
12	('thanks', 120)	('you', 531)	('work', 11.508486184129177)	('on', 31.905838605043645)	('don', 8.00259482260771)	('this', 17.56025292954881)
13	('card', 115)	('this', 582)	('card', 10.685646062178995)	('this', 30.66893774785233)	('monitor', 7.936521734883877)	('on', 17.08324914164528)
14	('32', 113)	('be', 410)	('monitor', 10.682428890754101)	('if', 26.193376000421924)	('simms', 7.488571293435438)	('if', 15.139262719417871)
15	('memory', 112)	('if', 402)	('ve', 10.364887792214013)	('be', 25.347946863179516)	('work', 7.2459737657071726)	('work', 14.906892373432164)
16	('new', 110)	('but', 361)	('need', 10.33466781921155)	('can', 22.688018912060382)	('need', 7.0132100850139265)	('this', 13.8881858267192)
17	('monitor', 106)	('or', 358)	('want', 9.974427058946128)	('want', 22.54083394319228)	('want', 6.720023195038986)	('my', 13.836463613476601)
18	('disk', 105)	('can', 357)	('simms', 9.451730884950774)	('but', 22.0050399457239)	('quadra', 6.678830075390477)	('or', 13.653163495302813)
19	('ram', 103)	('not', 347)	('sci', 9.112740317134982)	('my', 22.002007970462575)	('ve', 6.646919590248976)	('but', 13.32264381200065)

Рисунок 13 – Без стемминга comp.sys.mac.hardware

	Count		TF		TF-IDF	
	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами	Без стоп-слов	С стоп-словами
0	('space', 989)	('the', 6317)	('space', 45.310295270374944)	('the', 209.30732651695874)	('space', 20.136904785718123)	('the', 80.52942869455931)
1	('nasa', 374)	('of', 2968)	('like', 21.568656187462523)	('to', 100.85293110629775)	('like', 11.005688172976395)	('to', 42.89403622241328)
2	('launch', 267)	('to', 2856)	('just', 18.992547008148122)	('of', 93.87444120270081)	('nasa', 10.213775809877331)	('of', 40.31581854318306)
3	('earth', 222)	('and', 2604)	('nasa', 16.874407220071564)	('and', 75.70807365227847)	('just', 9.881160350206702)	('and', 34.65318262723836)
4	('like', 222)	('in', 1828)	('think', 14.756642307825098)	('in', 58.849367107761545)	('think', 8.620891576610706)	('is', 28.159840973318197)
5	('data', 216)	('is', 1518)	('don', 14.187483014334049)	('is', 58.11974974845199)	('don', 8.137859822946584)	('in', 27.25791211331705)
6	('orbit', 201)	('for', 1284)	('know', 13.325869774977626)	('it', 54.05386188615627)	('know', 7.830931680873458)	('it', 26.2749439416272)
7	('time', 197)	('that', 1138)	('time', 11.791180258291359)	('that', 51.59644057284977)	('moon', 7.490293275540706)	('that', 25.59082055896577)
8	('shuttle', 192)	('it', 1083)	('people', 11.600146184347478)	('for', 40.34599120676248)	('launch', 7.465895369489682)	('for', 21.82334206068477)
9	('just', 189)	('on', 1024)	('orbit', 11.431327588348053)	('on', 34.67928531925788)	('people', 7.31137650874348)	('you', 20.862996836332314)
10	('satellite', 187)	('space', 989)	('launch', 11.201773189672254)	('you', 34.16614014530568)	('orbit', 7.290604940645562)	('on', 18.94392399331072)
11	('lunar', 182)	('be', 848)	('moon', 10.89312079155181)	('be', 33.686476383451664)	('time', 6.835065067505623)	('be', 18.721820630891614)
12	('moon', 168)	('are', 740)	('earth', 10.588084874620773)	('this', 29.12864114395294)	('earth', 6.5695480796588726)	('space', 17.395953408310811)
13	('new', 158)	('this', 737)	('shuttle', 8.83128529703086)	('space', 26.2378465444882)	('shuttle', 6.3405250787438385)	('this', 16.780435570689672)
14	('program', 156)	('you', 722)	('use', 8.829843418967132)	('was', 25.320954227410255)	('lunar', 5.857580010896132)	('was', 16.11812034662486)
15	('don', 151)	('as', 640)	('does', 8.781453538849481)	('have', 24.438003301462338)	('does', 5.818594588061753)	('as', 14.935202563524841)
16	('people', 146)	('have', 562)	('new', 8.674026683950443)	('as', 23.943475013407177)	('data', 5.774500803184407)	('have', 14.925750843540607)
17	('people', 142)	('with', 561)	('long', 8.55107484500191)	('are', 23.691502511438475)	('thanks', 5.654236282280587)	('are', 14.860936986269719)
18	('mission', 141)	('at', 556)	('data', 8.439022631202338)	('if', 20.603316946545803)	('new', 5.524701152622701)	('they', 13.989381156801263)
19	('use', 134)	('from', 547)	('make', 8.019852306535931)	('not', 20.470092229846834)	('things', 5.41076571682147)	('would', 13.386451891244565)

Рисунок 14 – Без стемминга для sci.space

6) Используя конвейер (Pipeline) реализовать модель Наивного Байесовского классификатора и выявить на основе показателей качества (значения полноты, точности, f1-меры и аккуратности), какая предварительная обработка данных обеспечит наилучшие результаты классификации. Должны быть исследованы следующие характеристики:

- Отсечение – не отсечение стоп-слов
- Количество информативных терминов (max\_features)
- Взвешивание: Count, TF, TF-IDF



	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,002538	0,903896	0,926396	0,608696	0,610943371	0,911106001
<b>recall</b>	0,5	0,486713	0,800439	0,608696	0,595717294	0,608695652
<b>f1-score</b>	0,005051	0,632727	0,858824	0,608696	0,498867102	0,719551262
<b>support</b>	2	715	456	0,608696	1173	1173

Рисунок 15 – Пример работы программы со следующими параметрами  
(max\_features = 1000, со стоп словами, без TF, TF-IDF)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,002538	0,903896	0,926396	0,608696	0,610943371	0,911106001
<b>recall</b>	0,5	0,486713	0,800439	0,608696	0,595717294	0,608695652
<b>f1-score</b>	0,005051	0,632727	0,858824	0,608696	0,498867102	0,719551262
<b>support</b>	2	715	456	0,608696	1173	1173

Рисунок 16 – Пример работы программы со следующими параметрами  
(max\_features=1000, со стоп словами без tf, с idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,78934	0,805195	0,941624	0,845695	0,845386424	0,852654101
<b>recall</b>	0,861496	0,856354	0,824444	0,845695	0,847431293	0,8456948
<b>f1-score</b>	0,823841	0,829987	0,879147	0,845695	0,844324864	0,846954723
<b>support</b>	361	362	450	0,845695	1173	1173

Рисунок 17 – Пример работы программы со следующими параметрами  
(max\_features=1000, со стоп словами с tf, без idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,799492	0,797403	0,944162	0,8474	0,84701914	0,854365291
<b>recall</b>	0,849057	0,872159	0,826667	0,8474	0,84929412	0,847399829
<b>f1-score</b>	0,823529	0,833107	0,881517	0,8474	0,846051064	0,848649282
<b>support</b>	371	352	450	0,8474	1173	1173

Рисунок 18 – Пример работы программы со следующими параметрами  
(max\_features=1000, со стоп словами, с tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,002538	0,898701	0,918782	0,604433	0,606673699	0,905030872
<b>recall</b>	0,5	0,485955	0,788671	0,604433	0,591542027	0,604433078
<b>f1-score</b>	0,005051	0,630811	0,848769	0,604433	0,494876953	0,715032177
<b>support</b>	2	712	459	0,604433	1173	1173

Рисунок 19 – Пример работы программы со следующими параметрами  
(max\_features=1000, без стоп слов без tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,002538	0,898701	0,918782	0,604433	0,606673699	0,905030872
<b>recall</b>	0,5	0,485955	0,788671	0,604433	0,591542027	0,604433078
<b>f1-score</b>	0,005051	0,630811	0,848769	0,604433	0,494876953	0,715032177
<b>support</b>	2	712	459	0,604433	1173	1173

Рисунок 20 – Пример работы программы со следующими параметрами  
(max\_features=1000, без стоп слов, без tf, с idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,72335	0,766234	0,959391	0,816709	0,816324961	0,83624778
<b>recall</b>	0,868902	0,850144	0,759036	0,816709	0,826027559	0,81670929
<b>f1-score</b>	0,789474	0,806011	0,847534	0,816709	0,814339415	0,81901527
<b>support</b>	328	347	498	0,816709	1173	1173

Рисунок 21 – Пример работы программы со следующими параметрами  
(max\_features=1000, без стоп слов, с tf, без idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,753807	0,768831	0,939086	0,820972	0,820574857	0,831934639
<b>recall</b>	0,838983	0,838527	0,793991	0,820972	0,823833793	0,820971867
<b>f1-score</b>	0,794118	0,802168	0,860465	0,820972	0,818916928	0,822898297
<b>support</b>	354	353	466	0,820972	1173	1173

Рисунок 22 – Пример работы программы со следующими параметрами  
(max\_features=1000, без стоп слов, с tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,007614	0,919481	0,967005	0,629156	0,631366603	0,933372292
<b>recall</b>	0,5	0,499295	0,831878	0,629156	0,610390837	0,62915601
<b>f1-score</b>	0,015	0,647166	0,894366	0,629156	0,518844186	0,740452403
<b>support</b>	6	709	458	0,629156	1173	1173

Рисунок 23 – Пример работы программы со следующими параметрами  
(max\_features=5000, со стоп словами без tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,007614	0,919481	0,967005	0,629156	0,631366603	0,93337229
<b>recall</b>	0,5	0,499295	0,831878	0,629156	0,610390837	0,62915601
<b>f1-score</b>	0,015	0,647166	0,894366	0,629156	0,518844186	0,7404524
<b>support</b>	6	709	458	0,629156	1173	1173

Рисунок 24 – Пример работы программы со следующими параметрами  
(max\_features=5000, со стоп словами, без tf, с idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,799492	0,820779	0,939086	0,853367	0,8531193	0,859440927
<b>recall</b>	0,863014	0,880223	0,824053	0,853367	0,855763331	0,853367434
<b>f1-score</b>	0,83004	0,849462	0,877817	0,853367	0,852439737	0,854272287
<b>support</b>	365	359	449	0,853367	1173	1173

Рисунок 25 – Пример работы программы со следующими параметрами

(max\_features=5000, со стоп словами с tf, без idf=False)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,819797	0,818182	0,944162	0,86104	0,860713736	0,866277901
<b>recall</b>	0,859043	0,889831	0,839729	0,86104	0,862867394	0,861040068
<b>f1-score</b>	0,838961	0,852503	0,888889	0,86104	0,86011777	0,861903944
<b>support</b>	376	354	443	0,86104	1173	1173

Рисунок 26 – Пример работы программы со следующими параметрами

(max\_features=5000, со стоп словами с tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,005076	0,916883	0,959391	0,624893	0,627116707	0,929303791
<b>recall</b>	0,4	0,491643	0,84	0,624893	0,577214485	0,624893436
<b>f1-score</b>	0,010025	0,640073	0,895735	0,624893	0,515277396	0,735466982
<b>support</b>	5	718	450	0,624893	1173	1173

Рисунок 27 – Пример работы программы со следующими параметрами

(max\_features=5000, без стоп слов, без tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,005076	0,916883	0,959391	0,624893	0,627116707	0,929303791
<b>recall</b>	0,4	0,491643	0,84	0,624893	0,577214485	0,624893436
<b>f1-score</b>	0,010025	0,640073	0,895735	0,624893	0,515277396	0,735466982
<b>support</b>	5	718	450	0,624893	1173	1173

Рисунок 28 – Пример работы программы со следующими параметрами

(max\_features=5000, без стоп слов без tf, с idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,751269	0,815584	0,951777	0,839727	0,839543367	0,852715051
<b>recall</b>	0,883582	0,872222	0,784519	0,839727	0,84677438	0,839727195
<b>f1-score</b>	0,812071	0,842953	0,860092	0,839727	0,838372031	0,841117507
<b>support</b>	335	360	478	0,839727	1173	1173

Рисунок 29 – Пример работы программы со следующими параметрами

(max\_features=5000, без стоп слов, с tf, без idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,794416	0,833766	0,951777	0,860188	0,859986376	0,867867533
<b>recall</b>	0,876751	0,896648	0,818777	0,860188	0,864058679	0,860187553
<b>f1-score</b>	0,833555	0,864065	0,880282	0,860188	0,859300518	0,861111142
<b>support</b>	357	358	458	0,860188	1173	1173

Рисунок 30 – Пример работы программы со следующими параметрами  
(max\_features=5000, без стоп слов, с tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,01269	0,922078	0,964467	0,630861	0,633078427	0,93309354
<b>recall</b>	0,714286	0,499297	0,835165	0,630861	0,682915772	0,63086104
<b>f1-score</b>	0,024938	0,64781	0,895171	0,630861	0,522639555	0,740042914
<b>support</b>	7	711	455	0,630861	1173	1173

Рисунок 31 – Пример работы программы со следующими параметрами  
(max\_features=10000, со стоп словами, без tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,01269	0,922078	0,964467	0,630861	0,633078427	0,93309354
<b>recall</b>	0,714286	0,499297	0,835165	0,630861	0,682915772	0,63086104
<b>f1-score</b>	0,024938	0,64781	0,895171	0,630861	0,522639555	0,740042914
<b>support</b>	7	711	455	0,630861	1173	1173

Рисунок 32 – Пример работы программы со следующими параметрами  
(max\_features=10000, со стоп словами, без tf, с idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,804569	0,820779	0,941624	0,855925	0,855657371	0,861524612
<b>recall</b>	0,859079	0,880223	0,833708	0,855925	0,857669766	0,855924979
<b>f1-score</b>	0,830931	0,849462	0,884386	0,855925	0,854926359	0,856881675
<b>support</b>	369	359	445	0,855925	1173	1173

Рисунок 33 – Пример работы программы со следующими параметрами  
(max\_features=10000, со стоп словами, с tf, без idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,822335	0,823377	0,946701	0,86445	0,864137385	0,869301454
<b>recall</b>	0,859416	0,890449	0,847727	0,86445	0,865864386	0,864450128
<b>f1-score</b>	0,840467	0,855601	0,894484	0,86445	0,863517293	0,865322221
<b>support</b>	377	356	440	0,86445	1173	1173

Рисунок 34 – Пример работы программы со следующими параметрами  
(max\_features=10000, со стоп словами, с tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,002538	0,922078	0,951777	0,623188	0,625464214	0,930209017
<b>recall</b>	0,25	0,490331	0,842697	0,623188	0,52767604	0,623188406
<b>f1-score</b>	0,005025	0,640216	0,893921	0,623188	0,513054291	0,734298189
<b>support</b>	4	724	445	0,623188	1173	1173

Рисунок 35 – Пример работы программы со следующими параметрами  
(max\_features=10000, без стоп слов, без tf и idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,002538	0,922078	0,951777	0,623188	0,625464214	0,930209017
<b>recall</b>	0,25	0,490331	0,842697	0,623188	0,52767604	0,623188406
<b>f1-score</b>	0,005025	0,640216	0,893921	0,623188	0,513054291	0,734298189
<b>support</b>	4	724	445	0,623188	1173	1173

Рисунок 36 – Пример работы программы со следующими параметрами  
(max\_features=10000, без стоп слов, без tf, с idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,756345	0,81039	0,956853	0,841432	0,84119586	0,85487535
<b>recall</b>	0,881657	0,881356	0,783784	0,841432	0,848932174	0,841432225
<b>f1-score</b>	0,814208	0,844384	0,861714	0,841432	0,84010208	0,842795226
<b>support</b>	338	354	481	0,841432	1173	1173

Рисунок 37 – Пример работы программы со следующими параметрами  
(max\_features=10000, без стоп слов с tf, без idf)

	0	1	2	accuracy	macro avg	weighted avg
<b>precision</b>	0,791878	0,825974	0,956853	0,858483	0,858234997	0,866839524
<b>recall</b>	0,876404	0,888268	0,821351	0,858483	0,862007804	0,858482523
<b>f1-score</b>	0,832	0,855989	0,883939	0,858483	0,857309424	0,859645494
<b>support</b>	356	358	459	0,858483	1173	1173

Рисунок 38 – Пример работы программы со следующими параметрами  
(max\_features=10000, без стоп слов, с tf и idf)

По результатам классификации наиболее подходящая предварительная обработка данных является со следующими параметрами:

- с tf и tf-idf;
- max\_features = 10000;
- со стоп словами.

	precision	recall	f1-score	support
0	0.82	0.86	0.84	377
1	0.82	0.89	0.86	356
2	0.95	0.85	0.89	440
accuracy			0.86	1173
macro avg	0.86	0.87	0.86	1173
weighted avg	0.87	0.86	0.87	1173

Рисунок 39 – Результат работы программы

Код программы

```
#!/usr/bin/env python
# coding: utf-8

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.pipeline import Pipeline
from sklearn.naive_bayes import MultinomialNB
from nltk.stem import *
from nltk import word_tokenize
import itertools
import nltk

# ## Загрузка выборки

categories = ['comp.os.ms-windows.misc', 'comp.sys.mac.hardware',
'sci.space']

remove = ['headers', 'footers', 'quotes']

twenty_train_full = fetch_20newsgroups(subset='train', shuffle=True,
random_state=42, categories=categories, remove=remove)

twenty_test_full = fetch_20newsgroups(subset='test', shuffle=True,
random_state=42, categories=categories, remove=remove)

twenty_train_full = twenty_train_full.data
```

```

twenty_test_full = twenty_test_full.data

twenty_train = dict()
twenty_test = dict()
for category in categories:
    twenty_train[category] = fetch_20newsgroups(subset='train',
shuffle=True, random_state=42, categories=[category], remove=remove)
    twenty_test[category] = fetch_20newsgroups(subset='test', shuffle=True,
random_state=42, categories=[category], remove=remove)

    twenty_train[category] = twenty_train[category].data
    twenty_test[category] = twenty_test[category].data

twenty_train['full'] = twenty_train_full
twenty_test['full'] = twenty_test_full

# ## СТЕММИНГ

def stemming(data):
    porter_stemmer = PorterStemmer()
    stem = []
    for text in data:
        nltk_tokens = word_tokenize(text)
        line = ""
        for word in nltk_tokens:
            line += ' ' + porter_stemmer.stem(word)
        stem.append(line)
    return stem

```



```

stem_train = dict()
stem_test = dict()
for category in categories:
    stem_train[category] = stemming(twenty_train[category])
    stem_test[category] = stemming(twenty_test[category])

stem_train['full'] = stemming(twenty_train['full'])
stem_test['full'] = stemming(twenty_test['full'])

# ## Векторизация

def SortbyTF(inputStr):
    return inputStr[1]
def top_list(vect, data, count):
    x = list(zip(vect.get_feature_names(),np.ravel(data.sum(axis=0))))
    x.sort(key=SortbyTF, reverse = True)
    return x[:count]

# ## Итоговая таблица

def process(train, categories):
    cats = categories[:]
    cats.append('full')
    mux = pd.MultiIndex.from_product([[ 'Count','TF','TF-IDF'], [ 'Без стоп-
слов','С стоп-словами']])
    summary = dict()
    for category in cats:
        summary[category] = pd.DataFrame(columns=mux)

stop_words = [None, 'english']

```

```

idf = [False, True]

indx_stop = {
    'english': 'Без стоп-слов',
    None: 'С стоп-словами'
}

indx_tf = {
    False: 'TF',
    True: 'TF-IDF'
}

for category in cats:
    for stop in stop_words:
        vect = CountVectorizer(max_features=10000, stop_words=stop)
        vect.fit(train[category])
        train_data = vect.transform(train[category])
        summary[category]['Count',    indx_stop[stop]]    =    top_list(vect,
train_data, 20)

        for tf in idf:
            tfidf = TfidfTransformer(use_idf = tf).fit(train_data)
            train_fidf = tfidf.transform(train_data)
            summary[category][indx_tf[tf], indx_stop[stop]] = top_list(vect,
train_fidf, 20)

    return summary

summ_without_stem = process(twenty_train, categories)
summ_with_stem = process(stem_train, categories)

```

```

for cat in ['full'] + categories:
    summ_without_stem[cat].to_excel('without_stem_' + cat + '.xlsx')
    summ_with_stem[cat].to_excel('with_stem_' + cat + '.xlsx')

# ## Pipelines
import os

def print_classification_score(clf, data):
    print(classification_report(gs_clf.predict(data.data), data.target))

categories = ['comp.os.ms-windows.misc', 'comp.sys.mac.hardware',
'sci.space']

remove = ['headers', 'footers', 'quotes']

twenty_train_full = fetch_20newsgroups(subset='train', shuffle=True,
random_state=42, categories=categories, remove=remove)

twenty_test_full = fetch_20newsgroups(subset='test', shuffle=True,
random_state=42, categories=categories, remove=remove)

def presprocess(data, max_features, stop_words, use_tf, use_idf):
    tf = None
    cv = CountVectorizer(max_features=max_features,
stop_words=stop_words).fit(data)
    if use_tf:
        tf = TfidfTransformer(use_idf=use_idf).fit(cv.transform(data))
    return cv, tf

def models_grid_search(data_train, data_test):
    max_features = [1000,5000,10000]
    stop_words = ['english', None]
    use_tf = [True, False]
    use_idf = [True, False]

```

```

res = dict()
for param in itertools.product(max_features, stop_words, use_tf, use_idf):
    cv, tf = prespocess(data_train.data, param[0], param[1], param[2],
param[3])
    if tf:
        clf = MultinomialNB().fit(tf.transform(cv.transform(data_train.data)),
data_train.target)
        prep_test = tf.transform(cv.transform(data_test.data))
    else:
        clf = MultinomialNB().fit(cv.transform(data_train.data),
data_train.target)
        prep_test = cv.transform(data_test.data)

    name =
f'max_features={param[0]}_stop_words={param[1]}_use_tf={param[2]}_use_idf
={param[3]}'
    res[name] = pd.DataFrame(classification_report(clf.predict(prep_test),
data_test.target, output_dict=True))

    return res

scores = models_grid_search(twenty_train_full, twenty_test_full)

if not os.path.exists('scores'):
    os.makedirs('scores')

for name, score in scores.items():
    score.to_excel('scores/' + name + '.xlsx')

from sklearn.model_selection import GridSearchCV
parameters = {
    'vect__max_features': (1000,5000,10000),

```

```
'vect__stop_words': ('english', None),  
'tfidf__use_idf': (True, False),  
}
```

```
text_clf = Pipeline([  
    ('vect', CountVectorizer()),  
    ('tfidf', TfidfTransformer()),  
    ('clf', MultinomialNB())  
])
```

```
gs_clf = GridSearchCV(text_clf, parameters, n_jobs=-1, cv=3)  
gs_clf.fit(X = twenty_train_full.data, y = twenty_train_full.target)  
print_classification_score(gs_clf, twenty_test_full)
```

```
gs_clf.best_params_
```

## Вывод

В ходе выполнения данной лабораторной работы мы получили базовые навыки работы с языком python и набором функций для анализа и обработки данных.

## Контрольные вопросы

### 1) Особенности задачи классификации текстовых данных.

Анализе текстовых данных в машинном обучении используются методы регрессии, классификации и кластеризации. Данные методы были описаны в этой работе ранее. Но стоит отметить что есть главное отличие в анализе текстовых данных, так как сама обработка текста является очень сложной задачей в машинном обучении. Главное отличие – это интеллектуальный анализ текстовых данных. Так как текстовый документ для человека – это набор слов, который несет смысл, для машины – это просто битовые данные. И задача интеллектуального анализа текстовых данных состоит в том, чтобы машина смогла понимать смысл текстового документа. Перед тем как использовать алгоритмы машинного обучения, нужно также применить методы обработки текстовых данных.

Классификация текстовых документов, так же как и в случае классификации объектов, заключается в отнесении документа к одному из заранее известных классов. Часто классификацию применительно к текстовым документам называют категоризацией или рубрикацией. Очевидно, что данные названия происходят от задачи систематизации документов по каталогам, категориям и рубрикам. При этом структура каталогов может быть как одноуровневой, так и многоуровневой (иерархической).

### 2) Этапы предварительной обработки данных.

Этап подготовки и фильтрации данных может занять много времени.

Предварительная подготовка данных включает в себя:

- очистку;
- отбор экземпляров;
- нормализацию;

- преобразование данных;
- выделение признаков;
- отбор признаков;
- прочие манипуляции с данными.

### 3) Алгоритм и особенности Наивного Байесовского метода.

Алгоритм применения;

1. Для каждого класса вычисляется апостериорная вероятность;
2. Выбирается тот класс, для которого значение максимально.

Особенности:

- алгоритм легко и быстро предсказывает класс тестового набора данных. Он также хорошо справляется с многоклассовым прогнозированием;
- производительность наивного байесовского классификатора лучше, чем у других простых алгоритмов, таких как логистическая регрессия. Более того, вам требуется меньше обучающих данных;
- он хорошо работает с категориальными признаками(по сравнению с числовыми). Для числовых признаков предполагается нормальное распределение, что может быть серьезным допущением в точности нашего алгоритма.

#### 4) Как влияет размер словаря терминов на точность классификации?

При увеличении размера словаря точность оценок увеличивается.

#### 5) Как влияет способ взвешивания терминов на точность классификации?

Способ взвешивания терминов влияет прямо пропорционально на точность классификации.