In [1]:
```
import numpy as np
import pandas as pd
from typing import Dict, Tuple
from scipy import stats
from IPython.display import Image
from sklearn.datasets import load_iris, load_boston
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.metrics import precision_score, recall_score, f1_score, classification_
from sklearn.naive_bayes import ComplementNB
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_absolute_error, mean_squared_error, mean_squared_lo
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.svm import SVC, NuSVC, LinearSVC, OneClassSVM, SVR, NuSVR, LinearSVR
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

In [2]:
```
def accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray) -> Dict[int, float]:
    """
    Вычисление метрики accuracy для каждого класса
    y_true - истинные значения классов
    y_pred - предсказанные значения классов
    Возвращает словарь: ключ - метка класса,
    значение - Accuracy для данного класса
    """
    # Для удобства фильтрации сформируем Pandas DataFrame
    d = {'t': y_true, 'p': y_pred}
    df = pd.DataFrame(data=d)
    # Метки классов
    classes = np.unique(y_true)
    # Результирующий словарь
    res = dict()
    # Перебор меток классов
    for c in classes:
        # отфильтруем данные, которые соответствуют
        # текущей метке класса в истинных значениях
        temp_data_flt = df[df['t']==c]
        # расчет accuracy для заданной метки класса
        temp_acc = accuracy_score(
            temp_data_flt['t'].values,
            temp_data_flt['p'].values)
        # сохранение результата в словарь
        res[c] = temp_acc
    return res

def print_accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray):
    """
    Вывод метрики accuracy для каждого класса
    """
    accs = accuracy_score_for_classes(y_true, y_pred)
```

```
    if len(accs)>0:
        print('Метка \t Accuracy')
    for i in accs:
        print('{} \t {}'.format(i, accs[i]))
```

In [3]:
```
# Загрузка данных
df = pd.read_csv('D:\\Ботва\\Магистратура\\2сем\\ММО\\РК2\\Лосева\\imdb_sup.csv')
text_df=df.head(500).append(df.tail(500))
text_df.drop('Rating', axis=1, inplace=True)
text_df.head(15)
```

Out[3]:

	Review	Sentiment
0	Kurt Russell's chameleon-like performance, cou...	1
1	It was extremely low budget(it some scenes it ...	1
2	James Cagney is best known for his tough chara...	1
3	Following the brilliant "Goyôkiba" (aka. "Hanz...	1
4	One of the last classics of the French New Wav...	1
5	Having just watched this film again from a 199...	1
6	The Straight Story is a truly beautiful movie ...	1
7	Four teenage girlfriends drive to Fort Laurdal...	1
8	I haven't seen all of Jess Franco's movies, I ...	1
9	What's in a name? If the name is Jerry Bruckhe...	1
10	Batman: Mystery of the Batwoman is the latest ...	1
11	I have to say that Higher Learning is one of t...	1
12	In all honesty, this series is as much a class...	1
13	This movie is actually FUNNY! If you'd like to...	1
14	What does the Marquis de Sade have to do with ...	1

Изначально датасет содержит 50000 строк, что для выполнения нашей задачи слишком много. Так что сделаем из него датасет по-меньше

In [4]:
```
text_df.shape
```

Out[4]:
```
(1000, 2)
```

In [5]:
```
text_df['Sentiment'].unique()
```

Out[5]:
```
array([1, 0], dtype=int64)
```

Наш целевой признак - столбец Sentiment, который имеет всего 2 значения: 1 - если комментарий о фильме был положительным и 0 - если комментарий о фильме был отрицательным

In [6]:
```
# Сформируем общий словарь для обучения моделей из обучающей и тестовой выборки
vocab_list = text_df['Review'].tolist()
vocab_list[1:10]
```

Out[6]: ["It was extremely low budget(it some scenes it looks like they recorded with a home video recorder). However it does have a good plot line, and its easy to follow. 8 years after shooting her sexually abusive step father Amanda is released from the psychiatric ward, with the help of her doctor who she is secretly having an affair with. The doctor ends up renting her a house and buying her a car. But within the first 20 minutes of the movie Amanda kills him and buries him in her backyard. Then she see's her neighbor Richard sets eyes on him and stops at nothing until she has him. She acts innocent but after another neighbor Buzz finds out that Amanda killed that doctor and attempted to kill Richards wife Laurie (this is after Amanda and him get it on in the hot tub). Then she stops acting so Innocent and kills Buzz and later on attempts to kill Richard whom she supposedly loves and cares for. And you'll have to rent the movie to find out if Amanda dies or not. Overall good movie, reminds me a lot of my life you know the whole falling for the neighbor and stopping at nothing until you have him part.",
 'James Cagney is best known for his tough characters- and gangster roles but he has also played quite a lot \'soft\' characters in his career. This musical is one of them and it was the first but not the last musical movie Cagney would star in.

Cagney is even doing a bit of singing in this one and also quite an amount of dancing. And it needs to be said that he was not bad at it. He plays the role with a lot of confidence. He apparently had some dancing jobs in his early life before his acting career started to take off big time, so it actually isn\'t a weird thing that he also took on some musical acting roles in his career. He obviously also feels at ease in this totally different genre than most people are accustomed to seeing him in.

The movie is directed by Lloyd Bacon, who was perhaps among the best and most successful director within the genre. His earliest \'30\'s musicals pretty much defined the musical genre and he also was responsible for genre movies such as "42nd Street". His musicals were always light and fun to watch and more comedy like than anything else really. \'30\'s musicals never were really about its singing, this was something that more featured in \'40\'s and later made musicals, mainly from the MGM studios.

As usual it has a light and simple story, set in the musical world, that of course is also predictable and progresses in a formulaic way. It nevertheless is a fun and simple story that also simply makes this an entertaining movies to watch. So do the characters and actors that are portraying them. Sort of weird though that that the total plot line of the movie gets sort of abandoned toward the end of the movie, when the movie only starts to consists out of musical number routines.

The musical moments toward the ending of the movie are also amusing and well done, even though I\'m not a too big fan of the genre itself. Once again the musical numbers also feature a young Billy Barty. he often played little boys/babies/mice and whatever more early on in his career, including the movie musical "Gold Diggers of 1933", of one year earlier.

A recommendable early genre movie.

8/10',
 'Following the brilliant "Goyôkiba" (aka. "Hanzo The Razor - Sword Of Justice", 1972) and its excellent (and even sleazier) sequel "Goyôkiba: Kamisori Hanzô jigoku zeme" (aka. "Razor 2: The Snare", 1973), this "Goyôkiba: Oni no Hanzô yawahada koban" aka. "Razor 3: Who\'s Got The Gold" is the third, and sadly final installment to the awesome saga about the incorruptible Samurai-constable Hanzo \'The Razor\' Ittami (brilliantly played by the great Shintarô Katsu), who fights corruption with his fighting expertise as well as his enormous sexual powers. As a big fan of 70s exploitation cinema made in Nippon, "Sword Of Justice" became an instant favorite of mine, and I was therefore more than eager to find the sequels, and full of anticipation when I finally stumbled over them recently. While this third "Hanzo" film is just not quite as brilliant as its predecessors it is definitely another great piece of cult-cinema that no lover of Japanese exploitation cinema can afford to miss. "Who\'s Got The Gold" is a bit tamer than the two foregoing Hanzo films, but it is just as brilliantly comical and crudely humorous, and immediately starts out fabulously odd: The film begins, when Hanzo\'s two assistants see a female ghost when fishing. Having always wanted to sleep with a ghost, Hanzo insists that his assistants lead him to the site of the occurrence... If that is not a promising beginning for an awesome film experience, I don\'t know what is. Shintaro Katsu, one of my personal favorite actors, is once again brilliant in the role of Hanzo, a role that seems to have been written specifically for him. Katsu IS Hanzo, the obstinate and fearless constable, who hates corruption and deliberately insults his superiors, and whose unique interrogation techniques include raping female suspects. The interrogated women than immediately fall for him, due to his sexual powers and enormous penis, which he trains in a rather grotesque routine ritual. I will not give away more about the plot in "Who\'s Got The Gold", but I can assure that it is as cool as it sounds. The supporting performances are also very good, and, as in the predecessors, there are plenty of hilariously eccentric characters. This is sadly the last film in the awesomely sleazy

\'Hanzo\' series. If they had made 20 sequels more, I would have happily watched the
m all! The entire Hanzo series is brilliant, and while this third part is a bit infe
rior compared to its predecessors, it is definitely a must-see for all lovers of cul
t-cinema! Oh how I wish they had made more sequels!',
'One of the last classics of the French New Wave. For direction, cineaste Jean Eust
ache drew from the simplicity of early-century cinema; for story, Eustache drew on t
he torments of his own complicated love life. So many things can be said of this fil
m - observationally brilliant; self indulgently overlong; occasionally hilarious; em
otionally draining...etc. etc. In my mind, whatever complaints that can be leveled a
gainst this film are easily overshadowed by its numerous strengths. Every film stude
nt, writer, or simply anyone willing to handle a 3 hour film with no abrupt cuts, no
music video overstyling, no soap opera-like plot twists, and no banal dialogue shoul
d make it a point to see this movie. Everything is to be admired: the writing (conci
se, clever, surprisingly funny), acting (everyone, quite simply, is perfect in their
respective roles), and, simple direction (the viewer feels like a casual observer wi
thin the film) make this film unforgettable. This is undoubtedly a film that stays w
ith you.',
'Having just watched this film again from a 1998 showing off VH-1, I just had to co
mment.

The first time I saw this film on TV, it was about 1981, and I rem
ember taping it off of my mother\'s betamax. It wound up taping in black and white f
or some reason, which gave it a period look that I grew to like.

I rememb
er very distinctively the film beginning with the song, "My Bonnie", as the camera p
anned over a scene of Liverpool. I also remember the opening scene where Paul gestur
es to some girls and says, "Look, talent!" So it was with great irritation that I po
pped in my 1998 taped version and "remembered" that the film opens with "She Loves Y
ou", instead of "My Bonnie". When you see how slowly the camera pans vs. the speed o
f the music, you can see that "She Loves You" just doesn\'t fit. Also, in this "late
r" version when Paul sees the girls, he says, "Look, GIRLS!"..and somehow having rem
embered the earlier version, THAT word just didn\'t seem to fit, either. Why they fe
lt they had to Americanize this film for American audiences is beyond me. Personall
y, if I\'m going to watch a film about a British band, I want all of the British col
loquialisms and such that would be a part of their speech, mannerisms, etc.

Another irritation was how "choppy" the editing was for television. Just after Stu
gets beaten, for example, the film cuts to a commercial break-LOTS of \'em. Yeah, I
know it depends on the network, but it really ruins the effect of a film to have it
sliced apart, as we all know. What some people might find as insignificant in terms
of dialogue (and thereby okay to edit), may actually go the way of explaining a par
ticular action or scene that follows.

My point is, the "best" version of
this film was probably the earlier version I taped from 1981, which just so happene
d to include the "Shake, Rattle & Roll" scene that my 1998 version didn\'t. I starte
d to surmise that there had to have been two different versions made for television,
and a look at the "alternate versions" link regarding this film proved me right. Tha
t the American version had some shorter/cut/different scenes and/or dialogue is a hu
ge disappointment to me and something worth mentioning if one cares about such thing
s. Imo, ones best bet is to try and get a hold of the European version of this film,
if possible, and (probably even less possible), an unedited version. Sadly, I had to
discard my old betamax European version because I didn\'t know how to convert it.

All that aside, I found this film to be, perhaps, one of the best films rega
rding the story behind the "birth of the Beatles". Being well aware that artistic an
d creative license is often used in movies and TV when portraying events in history,
I didn\'t let any discrepancies mar my enjoyment of the film. Sure, you see the Beat
les perform songs at the Cavern that made me wonder, "Did they even write that back
then?? I don\'t think so", but, nevertheless, I thought it was a great film and the
performances, wonderful.

The real stand-out for me, in fact, was the acto
r who played John, Stephen MacKenna. I just about fell in love with him. His look, m
annerisms, personality and speaking voice seemed to be spot-on. He looked enough lik
e a young John for me to do a double-take towards the end of the film when you see t
he Beatles performing on Ed Sullivan for the first time. I actually found myself que
stioning whether or not it was actual Beatle footage, until I saw the other actors i
n the scene.

If you\'re looking for a dead accurate history of The Beatle
s\' life and beginnings, you can\'t get any better than, "The Beatles\' Anthology",
as it was "written" by the boys\', themselves. However, if you\'re looking for a fu
n snapshot of their pre-Beatlemania days leading up to their arrival in America and
you leave your anal critical assessments at the door, you can\'t go wrong with the
"Birth of the Beatles"--a MUST for any "real" or casual Beatle fan.',
'The Straight Story is a truly beautiful movie about an elderly man named Alvin Str
aight, who rides his lawnmower across the country to visit his estranged, dying brot
her. But that\'s just the basic synapsis...this movie is about so much more than tha
t. This was Richard\'s Farnworth\'s last role before he died, and it\'s definitely o

```
ne that he will be remembered for. He\'s a stubborn old man, not unlike a lot of the
old men that you and I probably know. <br /><br />"The Straight Story" is a movie th
at everyone should watch at least once in their lives. It will reach down and touch
 some part of you, at least if you have a heart, it will.',
 'Four teenage girlfriends drive to Fort Laurdale for spring break.Unfortunately the
y get a flat tire in Medley,Georgia and one of the girls witnesses a brutal murder d
eep in the woods.The local sheriff is behind the crime and the nightmare begins..."S
hallow Grave" is a pleasant low-budget surprise.The cast is likable enough,the direc
tion is steady and the violence is particularly nasty and misogynistic.Especially th
e second murder is pretty grim.The murderous sheriff isn\'t one-dimensional characte
r-in a couple of scenes it seems that he feels remorse for what he\'s done.The subpl
ot involving the two boys they meet in the diner goes nowhere,but the stalking scene
s in the woods are tense and exciting.7 out of 10.',
 'I haven\'t seen all of Jess Franco\'s movies, I have seen 5, I think, and there ar
e more than 180 of them. So maybe it\'s a bit early to say so but "Necronomicon Getr
äumte Sünden" (better known as \'Succubus\', but that is the cut version) is accordi
ng to me if not the best, certainly on of Franco\'s best. Franco is best known (alth
ough \'known\' might be slightly exaggerated) for "Vampiros Lesbos", a weird cultish
movie that got more acclaim in the mid 90\'s when people found out Jess Franco was a
lso an interesting composer. Through the soundtrack a happy few discovered the man a
nd found out what was to be expected after seeing the video clip of \'The lion and t
he cucumber\' (\'Vampyros Lesbos OST\'): Jess Franco is an overwhelming director. Wh
en the phone rang during \'Vampiros\', I let it ring. I just wanted to see more of t
he movie. Since that moment Franco never could grip me that much. But then I stumble
d on this movie. It is even better than "Vampiros Lesbos", I think. Franco is lookin
g for what he can do with a story and a camera. We find out he can do a lot. I certa
inly didn\'t expect to find "Necronomicon" that great: its beginning didn\'t impress
me at all. Remember, I had seen "Vampiros Lesbos" before (although chronologically t
hat came only three years later) and both movies kinda start the same. But then the
 story went on, puzzling and gripping, beautiful camera work and the stuff you would
like to see Godard do if he weren\'t so occupied with spreading his political messag
es. Later on in the movie I heard a dialogue about which art was or wasn\'t old-fash
ioned. The man says that all movies have to be old-fashioned because it takes weeks
 before the audience sees what got filmed. But the girl replies that "Bunuel, Fritz
 Lang and Godard yesterday made movies for tomorrow". Janine Reynaud is an interesti
ng lead actress and of course Howard Vernon, a Franco regular, is also there. Luckil
y the acting is good (something that can spoil a lot of Franco movies for you, but n
ot this one). But certainly watch out for the dummy scene. The erotic tension, the w
ild directing and the fact that it\'s a yesterday\'s movie for tomorrow make it a mo
vie a lot of people should see. The fact that it is a bit more accessible than "Vamp
iros Lesbos" certainly helps.',
 "What's in a name? If the name is Jerry Bruckheimer expect it to be filled with act
ion.<br /><br />In producer Bruckheimer's latest film, Gone in 60 Seconds, its all a
bout the nomenclature. With character monikers like Kip, Sway and The Sphinx and car
s idealized with names like Diane, Sue and the elusive Eleanor, it's only the non-st
op action that keeps you from wanting to just play the name game.<br /><br />Not a d
eep script by any means, but it is a great vehicle for action as Nicolas Cage as Mem
phis Raines, along with Angelina Jolie and Robert Duvall, comes out of car-thievery
 retirement to save his brother's life by stealing a list of 50 exotic cars in one n
ight. A remake of the 1974 cult hit, this film may not be destined for the same cult
status but it is entertaining.<br /><br />Surprisingly, it's the action that keeps y
ou watching not the acting. Although loaded with stars, none of them have standout p
erformances, including a very weak performance by one of my favorite up and comers,
 Giovanni Ribisi. Even Jolie, coming off her recent Oscar win, is just a token love
 interest with hardly any screen time.<br /><br />Can a series of beautiful cars and
the car chases they become involved in make a great film? I think so. The film is a
 pleasure to look at and although one particular scene takes you into the realm of u
nbelieveablity, the action is non-stop and the suspense is compelling. Just be wary
 of other drivers fighting for a pole position as you leave the theatre.<br /><br />
3 1/2 out of 5"]
```

In [7]:

```
vocabVect = CountVectorizer()
vocabVect.fit(vocab_list)
corpusVocab = vocabVect.vocabulary_
print('Количество сформированных признаков - {}'.format(len(corpusVocab)))
```

```
Количество сформированных признаков - 17896
```

In [8]:
```
for i in list(corpusVocab)[0:10]:
    print('{}={}'.format(i, corpusVocab[i]))
```

```
kurt=9020
russell=13645
chameleon=2717
like=9354
performance=11707
coupled=3693
with=17607
john=8666
carpenter=2536
flawless=6190
```

Векторизация признаков на основе CountVectorizer

Подсчитывает количество слов словаря, входящих в данный текст

In [9]:
```
test_features = vocabVect.transform(vocab_list)
```

In [10]:
```
test_features
```

Out[10]:
```
<1000x17896 sparse matrix of type '<class 'numpy.int64'>'
        with 137926 stored elements in Compressed Sparse Row format>
```

In [11]:
```
test_features.todense()
```

Out[11]:
```
matrix([[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

In [12]:
```
# Размер нулевой строки
len(test_features.todense()[0].getA1())
```

Out[12]:
```
17896
```

In [13]:
```
# Непустые значения нулевой строки
[i for i in test_features.todense()[0].getA1() if i>0]
```

Out[13]:
```
[1,
 1,
 1,
 3,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 2,
 1,
```

```
1,
1,
2,
1,
1,
1,
1,
1,
1,
1,
1,
2,
1,
1,
1,
1,
1,
1,
4,
1,
1,
1,
2,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
7,
4,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
2,
8,
2,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1]
```

Векторизация признаков на основе TfidfVectorizer

Вычисляет специфичность текста в корпусе текстов на основе метрики TF-IDF

In [14]:
```
tfidfv = TfidfVectorizer(ngram_range=(1,2))
tfidf_ngram_features = tfidfv.fit_transform(vocab_list)
tfidf_ngram_features
```

Out[14]:
```
<1000x138585 sparse matrix of type '<class 'numpy.float64'>'
        with 351280 stored elements in Compressed Sparse Row format>
```

In [15]:
```
tfidf_ngram_features.todense()
```

Out[15]:
```
matrix([[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]])
```

Оценка качества классификации

In [16]:
```
def VectorizeAndClassify(vectorizers_list, classifiers_list):
    for v in vectorizers_list:
        for c in classifiers_list:
            pipeline1 = Pipeline([("vectorizer", v), ("classifier", c)])
            score = cross_val_score(pipeline1, text_df['Review'], text_df['Sentiment
            print('Векторизация - {}'.format(v))
            print('Модель для классификации - {}'.format(c))
            print('Accuracy = {}'.format(score))
            print('============================')
```

In [17]:
```
vectorizers_list = [CountVectorizer(vocabulary = corpusVocab), TfidfVectorizer(vocab
classifiers_list = [ComplementNB(), KNeighborsClassifier()]
VectorizeAndClassify(vectorizers_list, classifiers_list)
```

```
Векторизация - CountVectorizer(vocabulary={'00': 0, '000': 1, '0069': 2, '007': 3,
'01': 4,
                            '06th': 5, '08': 6, '0f': 7, '10': 8, '100': 9,
                            '100th': 10, '101': 11, '102': 12, '10th': 13,
                            '11': 14, '112': 15, '11th': 16, '12': 17, '13': 18,
                            '13th': 19, '14': 20, '14th': 21, '15': 22,
                            '150': 23, '16': 24, '1600s': 25, '16éme': 26,
                            '17': 27, '1710': 28, '18': 29, ...})
Модель для классификации - ComplementNB()
Accuracy = 0.7990115864367362
============================
Векторизация - CountVectorizer(vocabulary={'00': 0, '000': 1, '0069': 2, '007': 3,
 '01': 4,
                            '06th': 5, '08': 6, '0f': 7, '10': 8, '100': 9,
                            '100th': 10, '101': 11, '102': 12, '10th': 13,
                            '11': 14, '112': 15, '11th': 16, '12': 17, '13': 18,
                            '13th': 19, '14': 20, '14th': 21, '15': 22,
                            '150': 23, '16': 24, '1600s': 25, '16éme': 26,
                            '17': 27, '1710': 28, '18': 29, ...})
Модель для классификации - KNeighborsClassifier()
```

```
Accuracy = 0.5879771987556419
============================
Векторизация - TfidfVectorizer(vocabulary={'00': 0, '000': 1, '0069': 2, '007': 3,
 '01': 4,
                               '06th': 5, '08': 6, '0f': 7, '10': 8, '100': 9,
                               '100th': 10, '101': 11, '102': 12, '10th': 13,
                               '11': 14, '112': 15, '11th': 16, '12': 17, '13': 18,
                               '13th': 19, '14': 20, '14th': 21, '15': 22,
                               '150': 23, '16': 24, '1600s': 25, '16éme': 26,
                               '17': 27, '1710': 28, '18': 29, ...})
Модель для классификации - ComplementNB()
Accuracy = 0.7889925853997711
============================
Векторизация - TfidfVectorizer(vocabulary={'00': 0, '000': 1, '0069': 2, '007': 3,
 '01': 4,
                               '06th': 5, '08': 6, '0f': 7, '10': 8, '100': 9,
                               '100th': 10, '101': 11, '102': 12, '10th': 13,
                               '11': 14, '112': 15, '11th': 16, '12': 17, '13': 18,
                               '13th': 19, '14': 20, '14th': 21, '15': 22,
                               '150': 23, '16': 24, '1600s': 25, '16éme': 26,
                               '17': 27, '1710': 28, '18': 29, ...})
Модель для классификации - KNeighborsClassifier()
Accuracy = 0.6530182877488268
============================
```

Наибольшая точность получилась при использовании CountVectorizer и ComplementNB()