

# Отчет

## по построению модели

### по классификации отзывов, с разработкой веб-сервиса на Django .

**Исходные данные:** открытый набор данных, который содержит в себе отзывы о фильмах, а также соответствующие им оценки рейтинга. Рейтинг может служить ориентиром для построения модели классификации отзывов.

<https://ai.stanford.edu/~amaas/data/sentiment>. Более подробное описание структуры файлов данных, а также сами данные можно найти по ссылке: [https://ai.stanford.edu/~amaas/data/sentiment/aclImdb\\_v1.tar.gz](https://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz)

#### Задача:

1. Обучить модель на языке Python для классификации отзывов.
2. Разработать веб-сервис на базе фреймворка Django для ввода отзыва о фильме с автоматическим присвоением рейтинга (от 1 до 10) и статуса комментария (положительный или отрицательный).

В рамках данного задания были проведены следующие этапы:

1. Загрузка и изучение данных
2. Обработка текста
3. Предсказание тональности отзывов (positive / negative)
4. Предсказание рейтинга
5. Разработка веб-сервиса на Django

#### 1. Загрузка и изучение данных

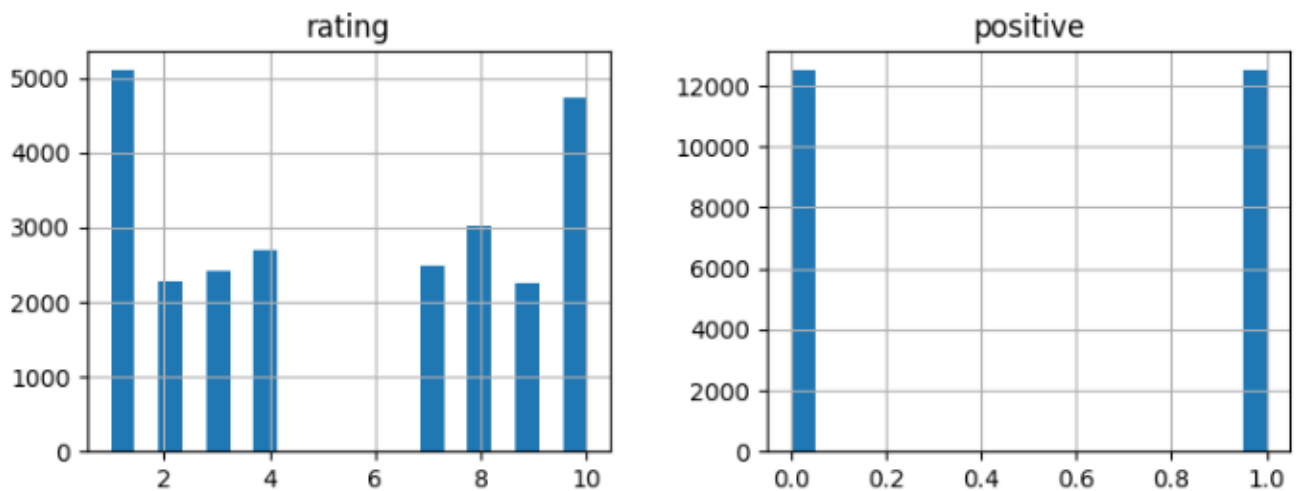
- 1.1. Данные были загружены в Google Colaboratory. Обучающая и тестовая выборки содержат данные по 25 тыс. записей с отзывами о фильмах, оценками (1-4 и 7-10) и статусами позитивный и негативный.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25000 entries, 0 to 24999
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  -
0   id          25000 non-null  int64
1   review      25000 non-null  object
2   rating      25000 non-null  int64
3   positive    25000 non-null  int64
dtypes: int64(3), object(1)
```

- 1.2. Отзывы с нейтральным рейтингом 5-6 не включены в обучающий и тестовый наборы данных.
- 1.3. Пропусков в данных не обнаружено.
- 1.4. Классы статуса находятся в равновесии 1:1 в обеих выборках (по 12,5 тыс.).
- 1.5. Классы оценок в обучающей и тестовой выборках имеют похожее распределение: самые популярные оценки - 1 и 10 (по ~5 тыс.); остальные оценки встречаются реже, но распределены по классам достаточно равномерно (от ~2 до 3 тыс.).

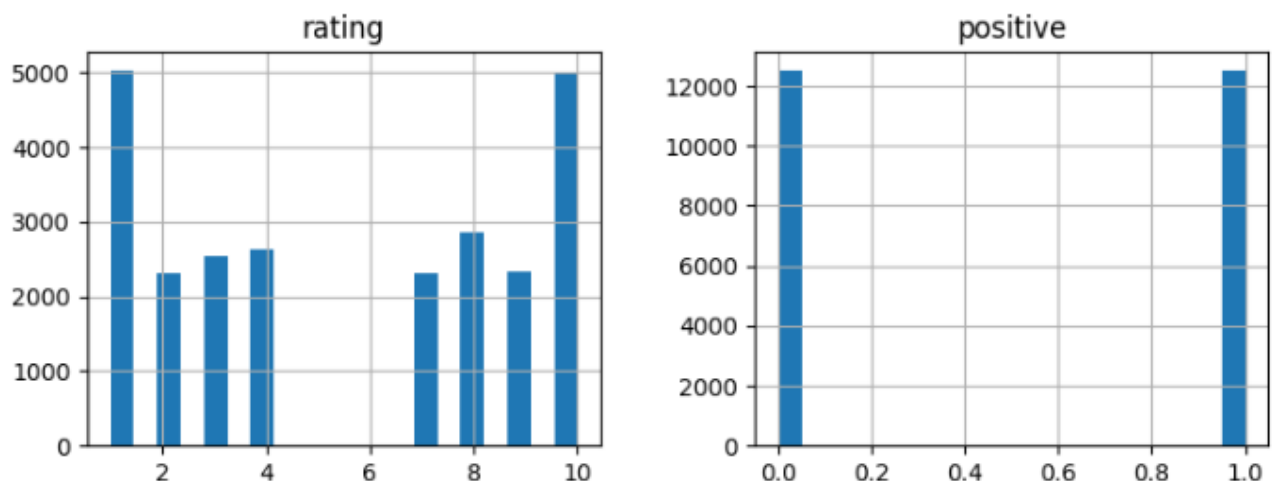
Распределение классов в обучающем наборе данных:

```
train_data[['rating', 'positive']].hist(bins=20, figsize=(9,3))  
plt.show()
```



Распределение классов в тестовом наборе данных:

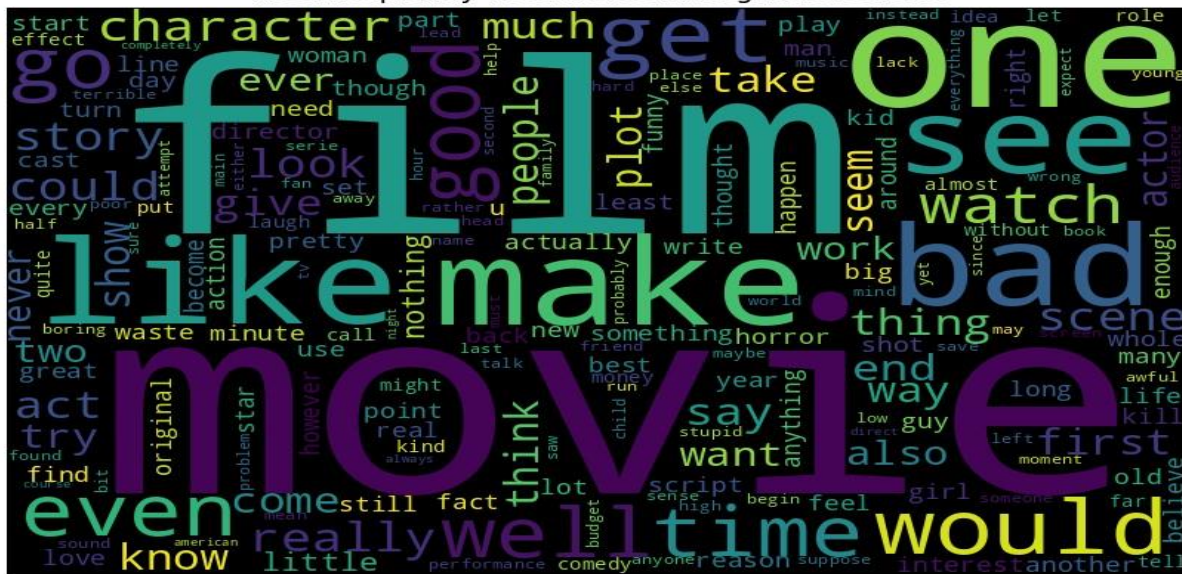
```
test_data[['rating', 'positive']].hist(bins=20, figsize=(9,3))  
plt.show()
```



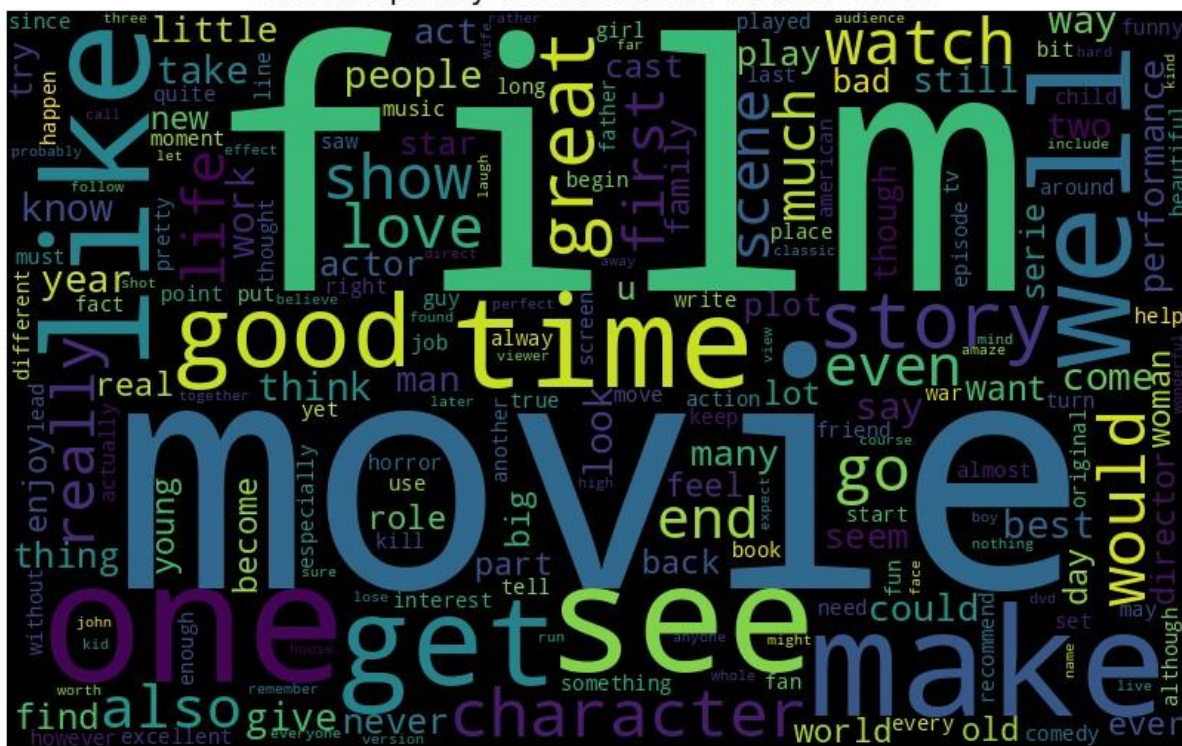
## 2. Обработка текста

- 2.1. Для обучения моделей были очищены от лишних символов, спецсимволов и стоп-слов, проведена лемматизация с помощью WordNetLemmatizer().
- 2.2. Приведены облака слов для негативных и позитивных отзывов: наиболее популярные слова во многом совпадают (film, movie, one, see, make, like и др.), но при этом есть и заметные отличия: например, в негативных отзывах намного чаще встречается слово bad, а в позитивных - слово love.

### Most frequently used words in Negative reviews



### Most frequently used words in Positive reviews



### 3. Предсказание тональности отзывов (positive / negative)

Выбор лучшей модели и подбор параметров будем проводить на кросс-валидации с помощью RandomizedSearchCV.

Перед подачей текста в модель будем проводить его векторизацию с помощью TfidfVectorizer.

Так как классы целевого признака сбалансированы, то качество модели можно оценивать метрикой accuracy (доля верно угаданных ответов).

Результаты лучших-моделей на кросс-валидации:

```
cv_results = cv_results.sort_values(by='accuracy', ascending=False).reset_index(drop=True)
cv_results.round(3)
```

	model	accuracy	roc_auc	recall	precision	params
0	SGDClassifier	0.887	0.887	0.902	0.876	{'sgdclassifier__loss': 'hinge', 'sgdclassifie...
1	LogisticRegression	0.886	0.886	0.897	0.879	{'logisticregression__solver': 'saga', 'logist...
2	LGBMClassifier	0.870	0.870	0.878	0.864	{'lgbmclassifier__n_estimators': 300, 'lgbmcla...
3	DummyClassifier	0.500	0.500	0.000	0.000	{'dummyclassifier__strategy': 'most_frequent'}

```
[ ] # параметры лучшей модели
    rs_model_sgd.best_params_

{'sgdclassifier__loss': 'hinge', 'sgdclassifier__alpha': 0.0001}
```

Лучшие результаты на кросс-валидации (accuracy 89,1%) показала модель SGDClassifier с параметрами {'sgdclassifier\_\_loss': 'hinge', 'sgdclassifier\_\_alpha': 0.0001}

Результаты лучшей модели на тестовых данных:

```
best_model_status = SGDClassifier(loss='hinge', alpha=0.0001, random_state=RANDOM_STATE)
best_model_status.fit(tf_idf_train, target_train)
predicted_test = best_model_status.predict(tf_idf_test)

print(classification_report(target_test, predicted_test))
```

	precision	recall	f1-score	support
0	0.88	0.87	0.88	12500
1	0.87	0.88	0.88	12500
accuracy			0.88	25000
macro avg	0.88	0.88	0.88	25000
weighted avg	0.88	0.88	0.88	25000

На тестовой выборке **accuracy = 88%**, что является неплохим результатом.

## 3.2. Нейросеть

Построим простую нейросеть для бинарной классификации:

```
def build_neural_network():
    model = tf.keras.Sequential([
        layers.Dense(64, input_shape=(10000, )),
        layers.Dropout(0.5),
        layers.Dense(10, activation='relu'),
        layers.Dropout(0.5),
        layers.Dense(1, activation='sigmoid')
    ])
    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['binary_accuracy'])

    return model
```

Результаты обучения:

```
history = model_nn.fit(tf_idf_train_slect.toarray(), target_train_nn,
                        epochs=4,
                        batch_size=100,
                        validation_data=(tf_idf_valid_slect.toarray(), target_valid_nn))
```

```
Epoch 1/4
200/200 [=====] - 12s 46ms/step - loss: 0.5229 - binary_accuracy: 0.7659 - val_loss: 0.3126 - val_binary_accuracy: 0.8894
Epoch 2/4
200/200 [=====] - 4s 21ms/step - loss: 0.2912 - binary_accuracy: 0.9049 - val_loss: 0.2609 - val_binary_accuracy: 0.8956
Epoch 3/4
200/200 [=====] - 6s 29ms/step - loss: 0.2159 - binary_accuracy: 0.9336 - val_loss: 0.2677 - val_binary_accuracy: 0.8912
Epoch 4/4
200/200 [=====] - 4s 21ms/step - loss: 0.1753 - binary_accuracy: 0.9508 - val_loss: 0.2950 - val_binary_accuracy: 0.8880
```

Построил графики изменения точности и потерь в зависимости от количества эпох.



Видно, что на валидации потери после 2 эпох начинают возрастать, а точность начинает падать. Это говорит о переобучении модели после 2 эпох



Переобучил нейросеть на 2 эпохах:

```
model_nn = build_neural_network()
```

Добавить текстовую ячейку

```
history = model_nn.fit(tf_idf_train_slct.toarray(),target_train_nn,  
                        epochs=2,  
                        batch_size=100,  
                        validation_data=(tf_idf_valid_slct.toarray(), target_valid_nn))
```

Epoch 1/2

200/200 [=====] - 6s 27ms/step - loss: 0.4999 - binary\_accuracy: 0.7951 - val\_loss: 0.2956 - val\_binary\_accuracy: 0.8912

Epoch 2/2

200/200 [=====] - 4s 21ms/step - loss: 0.2749 - binary\_accuracy: 0.9057 - val\_loss: 0.2584 - val\_binary\_accuracy: 0.8984

На валидации ассурасу = 89,1%. Это результат сопоставимый с выбранной лучшей ML-моделью SGDClassifier.

Результаты нейросети на тестовых данных:

```
predictions = model_nn.predict(tf_idf_test_slct.toarray())
```

782/782 [=====] - 3s 4ms/step

```
print(classification_report(target_test, predicted_test))
```

	precision	recall	f1-score	support
0	0.88	0.87	0.88	12500
1	0.87	0.88	0.88	12500
accuracy			0.88	25000
macro avg	0.88	0.88	0.88	25000
weighted avg	0.88	0.88	0.88	25000

Видим, что результат даже немного хуже, чем у SGDClassifier, и предсказание классов менее равномерное:

**Вывод:** для предсказания тональности отзывов остановим свой выбор на более простой модели SGDClassifier, которая дает достаточно хорошие результаты (ассурасу = 88%) с равномерным попаданием в классы.

#### 4. Предсказание рейтинга

- Перед подачей текста в модель буду проводить его TF-IDF преобразование с помощью TfidfVectorizer().
- Так как классы целевого признака несбалансированны, то качество модели можно оценивать средневзвешенным по классам ассурасу.

## LGBMClassifier – результаты на валидации:

```
model = LGBMClassifier(n_estimators = 300, random_state=RANDOM_STATE, class_weight='balanced')
model.fit(tf_idf_train, target_train_rat)
predicted_valid = model.predict(tf_idf_valid)

print(classification_report(target_valid_rat, predicted_valid))
```

	precision	recall	f1-score	support
1	0.58	0.68	0.62	1020
2	0.17	0.14	0.16	457
3	0.22	0.16	0.19	484
4	0.29	0.29	0.29	539
7	0.29	0.28	0.29	499
8	0.24	0.23	0.23	602
9	0.19	0.15	0.17	453
10	0.49	0.58	0.53	946
accuracy			0.38	5000
macro avg	0.31	0.31	0.31	5000
weighted avg	0.35	0.38	0.36	5000

## SGDClassifier – результаты на валидации:

```
model = SGDClassifier(random_state=RANDOM_STATE, class_weight='balanced')
model.fit(tf_idf_train, target_train_rat)
predicted_valid = model.predict(tf_idf_valid)

print(classification_report(target_valid_rat, predicted_valid))
```

	precision	recall	f1-score	support
1	0.58	0.76	0.66	1020
2	0.20	0.13	0.16	457
3	0.22	0.15	0.18	484
4	0.30	0.32	0.31	539
7	0.30	0.32	0.31	499
8	0.31	0.22	0.25	602
9	0.17	0.13	0.15	453
10	0.51	0.62	0.56	946
accuracy			0.40	5000
macro avg	0.32	0.33	0.32	5000
weighted avg	0.37	0.40	0.38	5000

## LogisticRegression – результаты на валидации:

```
model = LogisticRegression(random_state=RANDOM_STATE, class_weight='balanced', max_iter=300)
model.fit(tf_idf_train, target_train_rat)
predicted_valid = model.predict(tf_idf_valid)

print(classification_report(target_valid_rat, predicted_valid))
```

	precision	recall	f1-score	support
1	0.65	0.62	0.63	1020
2	0.25	0.26	0.25	457
3	0.24	0.22	0.23	484
4	0.31	0.32	0.32	539
7	0.31	0.37	0.34	499
8	0.29	0.26	0.28	602
9	0.20	0.21	0.20	453
10	0.55	0.54	0.55	946
accuracy			0.40	5000
macro avg	0.35	0.35	0.35	5000
weighted avg	0.40	0.40	0.40	5000

Лучшие результаты на валидации показала LogisticRegression с параметрами {class\_weight: 'balanced', max\_iter: 300} Результаты лучшей модели **на тестовых данных**:

```
best_model_rating = LogisticRegression(random_state=RANDOM_STATE, class_weight='balanced', max_iter=300)
best_model_rating.fit(tf_idf_train_, train_data['rating'])
predicted_test = best_model_rating.predict(tf_idf_test_)

print(classification_report(test_data['rating'], predicted_test))
```

	precision	recall	f1-score	support
1	0.63	0.60	0.62	5022
2	0.21	0.21	0.21	2302
3	0.23	0.22	0.23	2541
4	0.29	0.33	0.31	2635
7	0.27	0.30	0.28	2307
8	0.25	0.22	0.23	2850
9	0.21	0.22	0.21	2344
10	0.56	0.54	0.55	4999
accuracy			0.38	25000
macro avg	0.33	0.33	0.33	25000
weighted avg	0.38	0.38	0.38	25000

Результаты именно точных ответов не очень высокие (**accuracy = 38%**), что объясняется сложностью точного предсказания мультиклассов по сравнению с бинарной классификацией.

## 4.2. Нейросеть

Построил простую нейросеть для множественной классификации:

```
def build_neural_network_rat():
    model = tf.keras.Sequential([
        layers.Dense(128, input_shape=(10000, )),
        layers.Dropout(0.5),
        layers.Dense(32, activation='relu'),
        layers.Dropout(0.5),
        layers.Dense(11, activation='softmax')
    ])
    model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['sparse_categorical_accuracy'])

    return model
```

```
model_nn_rat = build_neural_network_rat()
```

```
history = model_nn_rat.fit(tf_idf_train_slct.toarray(), target_train_rat,
                           epochs=4,
                           batch_size=100,
                           validation_data=(tf_idf_valid_slct.toarray(), target_valid_rat))
```

```
Epoch 1/4
200/200 [=====] - 9s 43ms/step - loss: 2.0615 - sparse_categorical_accuracy: 0.2632 - val_loss: 1.6419 - val_sparse_categorical_accuracy: 0.3830
Epoch 2/4
200/200 [=====] - 7s 37ms/step - loss: 1.5733 - sparse_categorical_accuracy: 0.3997 - val_loss: 1.4604 - val_sparse_categorical_accuracy: 0.4260
Epoch 3/4
200/200 [=====] - 7s 37ms/step - loss: 1.3968 - sparse_categorical_accuracy: 0.4629 - val_loss: 1.4337 - val_sparse_categorical_accuracy: 0.4282
Epoch 4/4
200/200 [=====] - 8s 42ms/step - loss: 1.2790 - sparse_categorical_accuracy: 0.5041 - val_loss: 1.4511 - val_sparse_categorical_accuracy: 0.4266
```



Результаты обучения:

```
predictions = model_nn_rat.predict(tf_idf_valid_slct.toarray())

pred=[]
for i in predictions:
    pred.append(np.argmax(i))

print(classification_report(target_valid_rat, pred))
```

```
157/157 [=====] - 1s 9ms/step
      precision    recall  f1-score   support

     1       0.53       0.81       0.64       1020
     2       0.12       0.00       0.00        457
     3       0.28       0.18       0.22       484
     4       0.32       0.36       0.34       539
     7       0.33       0.31       0.32       499
     8       0.29       0.29       0.29       602
     9       0.28       0.04       0.07       453
    10       0.49       0.72       0.58       946

 accuracy                   0.43       5000
 macro avg       0.33       0.34       0.31       5000
 weighted avg    0.37       0.43       0.37       5000
```

**Вывод:** Не смотря на то, что на валидации общая метрика ассурасу = 43% у нейросети получилась больше, чем у LogisticRegression, классы 2 и 9 вообще имеют почти нулевое f1-score. Вероятно, что более высокую точность можно достичь с помощью усложнения модели, например эмбиддингов на предобученной модели BERT, но в условиях ограниченности ресурсов остановим свой выбор на простой модели LogisticRegression.

## 5. Разработка веб-сервиса на Django

Проект расположен в репозитории по ссылке:

<https://github.com/Dimonius-73/Grinatom/tree/main>

Для запуска приложения на Windows локально скачайте репозиторий и в папке выполните команды:

1. pip install Django
2. pythpn manage.py runserver

```
Windows PowerShell
July 11, 2023 - 16:20:14
Django version 4.2.3, using settings 'rating_add.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

[11/Jul/2023 16:20:25] "GET /postuser/ HTTP/1.1" 200 1093
[11/Jul/2023 16:20:26] "GET /static/css/normalize.css HTTP/1.1" 200 6138
[11/Jul/2023 16:20:26] "GET /static/css/styles.css HTTP/1.1" 200 2010
[11/Jul/2023 16:20:26] "GET /static/img/banner.jpeg HTTP/1.1" 200 466485
[11/Jul/2023 16:20:26] "GET /static/fonts/Roboto-Regular.woff2 HTTP/1.1" 200 12264
[11/Jul/2023 16:20:26] "GET /static/fonts/Roboto-Bold.woff2 HTTP/1.1" 200 12228
Not Found: /favicon.ico
[11/Jul/2023 16:20:26] "GET /favicon.ico HTTP/1.1" 404 2230
[11/Jul/2023 16:22:29] "GET / HTTP/1.1" 200 1117
[11/Jul/2023 16:22:34] "POST /postuser/ HTTP/1.1" 200 2355
[11/Jul/2023 16:34:08] "GET / HTTP/1.1" 200 1117
[11/Jul/2023 16:34:12] "POST /postuser/ HTTP/1.1" 200 1607
[11/Jul/2023 16:34:22] "GET / HTTP/1.1" 200 1117
[11/Jul/2023 16:34:27] "POST /postuser/ HTTP/1.1" 200 1087
[11/Jul/2023 16:35:36] "GET / HTTP/1.1" 200 1117
[11/Jul/2023 16:35:40] "POST /postuser/ HTTP/1.1" 200 1771
[11/Jul/2023 16:56:57] "GET / HTTP/1.1" 200 1117
[11/Jul/2023 16:57:11] "POST /postuser/ HTTP/1.1" 200 1087
[11/Jul/2023 16:57:27] "GET / HTTP/1.1" 200 1117
[11/Jul/2023 16:57:32] "POST /postuser/ HTTP/1.1" 200 1771
```

Проект будет доступен локально <http://127.0.0.1:8000/>



**Понравился ли Вам фильм :)**

Введите отзыв о фильме (на английском языке)

Проверить

# Тестирование сервиса

## Возьмем реальные отзывы с сайта IMDb на фильм

[www.imdb.com](#)Чапаяв (1934) - Чапаяв (1934) - User Reviews - IMDb

Menu

All

Search IMDb

IMDbPro

Watchlist

Sign In

Чапаяв (1934)

User Reviews

[Review this title](#)

8 Reviews

☐ Hide Spoilers

Filter by Rating: 

Show All

Sort by: 

Review Rating

★ 10/10

Historic, small art deviations from truth

[pouolga-462-640734](#) 2 January 2014

Vasily Ivanovich Chapayev the legendary figure of the Civil war in Russia, the people's leader, the self-taught, who held high posts at the expense of its own abilities in the absence of special military education. The regiment is not a gang of marauders, dare I say. The film is based on a scenario of Anna Furmanova, created based on the diaries of Dmitry Furmanov, his novel «Chapayev» and memoirs of veterans, who fought under the leadership of Chapayev. «Best foreign language film» according to the National Board of Review USA in 1935. According to a survey of film critics of the world (1978) film has been included in the list of the hundred best films of world cinema.

Чапаяв

Opinion

Awards

FAQ

User Reviews

User Ratings

External Reviews

Metacritic Reviews

Explore More

User Lists

Create a list »

Related lists from IMDb users

Stalinist cinema

a list of 21 titles

created 10 months ago

Нбы

a list of 23 titles

created 30 Sep 2021

### Определение вероятного рейтинга по рецензии

Результат анализа отзыва

Vasily Ivanovich Chapayev the legendary figure of the Civil war in Russia, the people's leader, the self-taught, who held high posts at the expense of its own abilities in the absence of special military education. The regiment is not a gang of marauders, dare I say. The film is based on a scenario of Anna Furmanova, created based on the diaries of Dmitry Furmanov, his novel «Chapayev» and memoirs of veterans, who fought under the leadership of Chapayev. «Best foreign language film» according to the National Board of Review USA in 1935. According to a survey of film critics of the world (1978) film has been included in the list of the hundred best films of world cinema.

Отзыв: Позитивный

Рейтинг: 10

Попробовать снова



www.imdb.com

Андрей Рублёв (1966) - Андрей Рублёв (1966) - User Reviews - IMDb

IMDbMenuAllSearch IMDbIMDbProWatchlist

Андрей Рублёв (1966)

User Reviews

Review this title

4 Reviews

Hide Spoilers

Filter by Rating: 3 Stars

Sort by: Featured

★ 3/10

I'm Apparently the Only Tarkovsky Fan with Enough Objectivity to Admit that this Film Sucks

ebossert · 26 November 2008

After being tortured with poor Russian film-making for my first three outings □ One Day In the Life of Ivan Denisovich (1970), Night Watch (2004), and Russian Ark (2002) □ I was lucky enough to experience the greatness of Andrei Tarkovsky with four consecutive quality films □ Solaris (1972), Stalker (1979), The Mirror (1975), and Ivan's Childhood (1962) □ the last of which is no less than spectacular. Riding on cloud 9, I popped Andrei Rublev into my DVD player with quiet anticipation as this film is considered by many to be Tarkovsky's greatest film (and one of the best films ever made).

Sufficed to say that my excitement subsided slightly after a ridiculous opening air balloon sequence, and dissipated completely within the first hour of mind-numbing boredom. Tarkovsky's trademark atmospheric are nowhere to be found, the dialogue consists of tireless sermonizing, and the scoring is hopelessly over-dramatic. I

90 out of 174 found this helpful. Was this review helpful? [Sign in](#) to vote.

Permalink

Андрей Рублёв

Opinion

Awards

FAQ

User Reviews

User Ratings

External Reviews


Metacritic Reviews

Explore More

User Lists


Create a list >

Related lists from IMDb users




2023

a list of 33 titles created 5 months ago



Top 50 (In Progress)

a list of 26 titles created 26 Oct 2021



Fuck church

a list of 26 titles created 4 months ago

Определение вероятного рейтинга по рецензии

## Результат анализа отзыва

After being tortured with poor Russian film-making for my first three outings □ One Day In the Life of Ivan Denisovich (1970), Night Watch (2004), and Russian Ark (2002) □ I was lucky enough to experience the greatness of Andrei Tarkovsky with four consecutive quality films □ Solaris (1972), Stalker (1979), The Mirror (1975), and Ivan's Childhood (1962) □ the last of which is no less than spectacular. Riding on cloud 9, I popped Andrei Rublev into my DVD player with quiet anticipation as this film is considered by many to be Tarkovsky's greatest film (and one of the best films ever made).

Sufficed to say that my excitement subsided slightly after a ridiculous opening air balloon sequence, and dissipated completely within the first hour of mind-numbing boredom. Tarkovsky's trademark atmospheric are nowhere to be found, the dialogue consists of tireless sermonizing, and the scoring is hopelessly over-dramatic. I

Отзыв: Негативный

Рейтинг: 1

Попробовать снова

Как видим сервис работает. Задача решена

## Контактная информация:

Автор: Назарьянц Дмитрий Александрович  
Город Москва.

## Специализации:

- BI-аналитик, аналитик данных
- Финансовый аналитик, инвестиционный аналитик
- Финансовый менеджер
- Бизнес-аналитик
- Дата-сайентист


## Обо мне

Опыт работы в производственных компаниях на руководящих позициях в финансовой сфере более 10 лет. Высшее экономическое образование. Значительный опыт участия в проектах по инвестиционному и финансовому моделированию.

Глубокое знание принципов и методов экономического, финансового и инвестиционного анализа деятельности предприятия. Уверенное понимание методологии и взаимосвязей системы бюджетов предприятия.

Экспертное владение инструментарием: MS Excel, PowerBI, Google Формы, Google таблицы, Data Studio и другие, а также все офисные программы. Осуществляю анализ данных в Python и SQL.

Интересны сложные, нестандартные задачи в области аналитики данных, финансовой и инвестиционной аналитики, машинного обучения и внедрение процессов, имеющих стратегическое значения для компании.

 +7(926) 422-63-98

- da.naz@mail.ru
- <https://t.me/Dimonius73>