

Министерство образования и науки Республики Башкортостан  
Государственное автономное профессиональное образовательное  
учреждение  
Уфимский колледж статистики, информатики и вычислительной техники

Машинное обучение  
Приложения ML

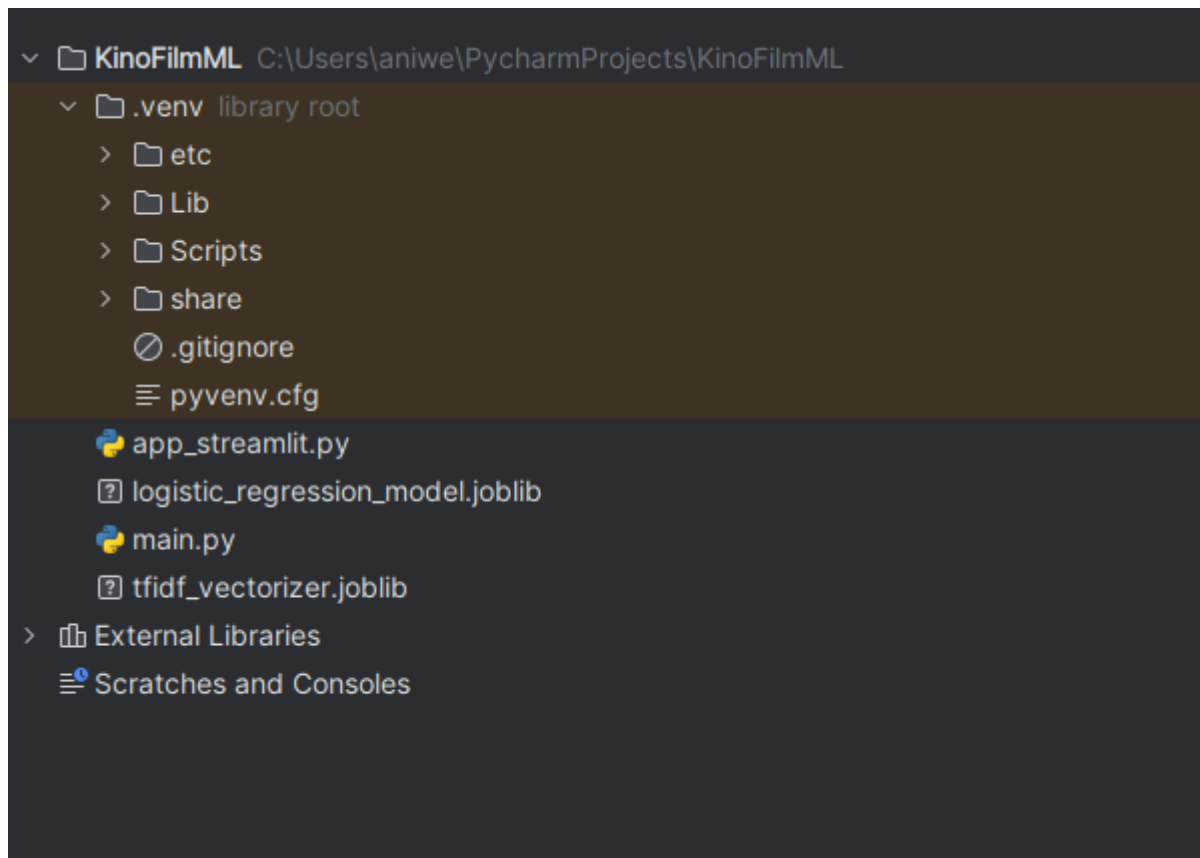
Выполнил  
студент гр. 22П-2

\_\_\_\_\_И.М.Звягинцев  
«\_\_\_»\_\_\_\_\_2025 г.

Проверил

\_\_\_\_\_К.Р.Флюинова  
«\_\_\_»\_\_\_\_\_2025 г.

Приложение ML –  
по дата-сету кино  
Структура проекта:



Библиотеки для работы веб приложения и api –

```
from fastapi import FastAPI
from pydantic import BaseModel
import joblib
import re
import string
from bs4 import BeautifulSoup
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
import pymorphy3

import streamlit as st
import requests
```

Для API:

Импорт библиотек:

Используются FastAPI (создание API), Pydantic (валидация входных данных), joblib (загрузка модели и векторизатора), а также инструменты для предобработки текста: nltk, pymorphy3, BeautifulSoup, re, string.

Инициализация приложения и загрузка моделей:

Создаётся экземпляр FastAPI: `app = FastAPI()`.

Загружаются обученная модель и TF-IDF векторизатор:

```
model = joblib.load('logistic_regression_model.joblib')
```

```
vectorizer = joblib.load('tfidf_vectorizer.joblib')
```

Инициализируется морфологический анализатор pymorphy3.

Загружаются токенизатор и стоп-слова из nltk.

Словарь жанров (кластеров):

`cluster_names` сопоставляет числовые метки кластеров с осмысленными названиями жанров фильмов.

Класс TextRequest:

Описывает структуру входящего JSON-запроса с единственным полем `text` (описание фильма). Используется для автоматической валидации данных через Pydantic.

Функции предобработки текста:

`clean_text_conditional`: очищает текст от HTML, пунктуации, цифр, приводит к нижнему регистру.

`tokenize_and_remove_stopwords`: разбивает текст на слова и удаляет стоп-слова.

`lemmatize_text`: приводит слова к нормальной форме с помощью pymorphy3.

`preprocess_text`: объединяет все этапы предобработки в одну функцию

Метод API (эндпоинт) `/predict`:

Принимает POST-запрос с описанием фильма.

Выполняет полную предобработку текста.

Векторизует текст с помощью TF-IDF.

Передаёт вектор в модель, получает предсказание кластера и вероятности по всем жанрам.

Возвращает JSON с id кластера, названием жанра и вероятностями для всех жанров.

```
main.py × app_streamlit.py
1 from fastapi import FastAPI
2 from pydantic import BaseModel
3 import joblib
4 import re
5 import string
6 from bs4 import BeautifulSoup
7 import nltk
8 from nltk.tokenize import word_tokenize
9 from nltk.corpus import stopwords
10 import pymorphy3
11
12 app = FastAPI()
13
14 # Загружаем сохранённые модель и векторизатор
15 model = joblib.load('logistic_regression_model.joblib')
16 vectorizer = joblib.load('tfidf_vectorizer.joblib')
17
18 morph = pymorphy3.MorphAnalyzer()
19 nltk.download('punkt')
20 nltk.download('stopwords')
21 russian_stopwords = set(stopwords.words('russian'))
22
23 # Словарь с названиями жанров (кластеров) фильмов
24 cluster_names = {
25     0: 'Анимационные приключения',
```

```

25         0: 'Анимационные приключения',
26         1: 'Триллеры и драмы',
27         2: 'Криминальные и психологические драмы',
28         3: 'Научная фантастика и романтика',
29         4: 'Боевики и комедии'
30     }
31
32     1 usage
33     class TextRequest(BaseModel):
34         text: str
35
36     1 usage
37     def clean_text_conditional(text: str) -> str:
38         if not isinstance(text, str):
39             return ""
40         if re.search(pattern: r'<[^>]+>', text):
41             text = BeautifulSoup(text, features: 'lxml').get_text()
42         text = text.lower()
43         text = "".join([ch if ch not in string.punctuation else " " for ch in text])
44         text = "".join([ch if not ch.isdigit() else " " for ch in text])
45         text = re.sub(pattern: r'\s+', repl: ' ', text).strip()
46         return text
47
48     1 usage
49     def tokenize_and_remove_stopwords(text: str) -> str:

```

```

1 usage
2 def tokenize_and_remove_stopwords(text: str) -> str:
3     tokens = word_tokenize(text, language='russian')
4     filtered_tokens = [word for word in tokens if word not in russian_stopwords and len(word) > 1]
5     return " ".join(filtered_tokens)
6
7 1 usage
8 def lemmatize_text(text: str) -> str:
9     tokens = word_tokenize(text, language='russian')
10    lemm_tokens = [morph.parse(token)[0].normal_form for token in tokens]
11    return " ".join(lemm_tokens)
12
13 1 usage
14 def preprocess_text(text: str) -> str:
15     text = clean_text_conditional(text)
16     text = tokenize_and_remove_stopwords(text)
17     text = lemmatize_text(text)
18     return text
19
20 @app.post("/predict")
21 async def predict_cluster(request: TextRequest):
22     processed_text = preprocess_text(request.text)
23     vectorized = vectorizer.transform([processed_text])
24     prediction = model.predict(vectorized)[0]
25     probabilities = model.predict_proba(vectorized)[0]

```

```
response = {
    "cluster_id": int(prediction),
    "cluster_name": cluster_names[int(prediction)],
    "probabilities": {cluster_names[i]: float(probabilities[i]) for i in range(len(probabilities))}
}
return response
```

Для приложения:

Импорт библиотек:

Используются streamlit для создания веб-интерфейса и requests для отправки HTTP-запросов к API.

Заголовок приложения:

st.title("Классификация фильмов по жанрам по описанию") — отображает заголовок на странице, информируя пользователя о назначении сервиса.

Текстовое поле для ввода:

st.text\_area — многострочное поле, куда пользователь вводит описание фильма для классификации.

Кнопка для запуска предсказания:

st.button("Предсказать жанр") — при нажатии инициирует процесс классификации.

Проверка ввода:

Если поле пустое, с помощью st.warning выводится предупреждение о необходимости ввести текст.

Отправка запроса к API:

При наличии текста отправляется POST-запрос на локальный сервер FastAPI (<http://127.0.0.1:8000/predict>) с JSON-объектом: {"text": input\_text}

Обработка ответа:

Если запрос успешен, из ответа берётся предсказанный жанр (cluster\_name), его ID (cluster\_id) и вероятности по всем жанрам (probabilities).

Результаты выводятся на страницу с помощью st.markdown и st.write.

Обработка ошибок:

Если запрос к API не удался, выводится сообщение об ошибке с помощью st.error.

```

import streamlit as st
import requests

st.title("Классификация фильмов по жанрам по описанию")

input_text = st.text_area("Введите описание фильма для классификации", height=200)

if st.button("Предсказать жанр"):
    if not input_text.strip():
        st.warning("Пожалуйста, введите описание фильма.")
    else:
        try:
            url = "http://127.0.0.1:8000/predict"
            response = requests.post(url, json={"text": input_text})
            response.raise_for_status()
            data = response.json()

            st.markdown(f"### Предсказанный жанр: {data['cluster_name']} (ID: {data['cluster_id']})")

            st.markdown("### Вероятности по жанрам:")
            for genre, prob in data['probabilities'].items():
                st.write(f"- {genre}: {prob:.3f}")

        except requests.exceptions.RequestException as e:
            st.error(f"Ошибка при запросе к API: {e}")

```

Запуск и работа приложения:

Сначала API –

```

(.venv) PS C:\Users\aniwe\PycharmProjects\KinoFilmML> uvicorn main:app --reload
INFO: Will watch for changes in these directories: ['C:\\Users\\aniwe\\PycharmProjects\\KinoFilmML']
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: Started reloader process [13064] using WatchFiles
C:\Users\aniwe\PycharmProjects\KinoFilmML\.venv\Lib\site-packages\sklearn\base.py:380: InconsistentVersionWarning

```

Потом приложение –

```

Установите последнюю версию PowerShell для новых функций и улучшения! https://aka.ms/PSWindows

(.venv) PS C:\Users\aniwe\PycharmProjects\KinoFilmML> streamlit run app_streamlit.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.24.132:8501

```

Сама работа –

# Классификация фильмов по жанрам по описанию

Введите описание фильма для классификации

Предсказать жанр



# Классификация фильмов по жанрам по описанию

Введите описание фильма для классификации

фильм рик мортить обзор фильм рик мортить это анимационный сериал который сочетать элемент научный фантастика комедия сюжет рассказывать приключение рик санчез безумный учёный внук мортить смит путешествовать различный измерение сталкиваться необычный существо ситуация приводить множество комичный п...

Предсказать жанр

**Предполагаемый жанр: анимационные приключения (ID: 0)**

## Вероятности по жанрам:

- Анимационные приключения: 0,427
- Триллеры и драмы: 0,169
- Криминальные и психологические драмы: 0,153
- Научная фантастика и романтика: 0,127
- Боевики и комедии: 0,124

# Классификация фильмов по жанрам по описанию

Введите описание фильма для классификации

фильм виктор франкенштейн фильм виктор франкенштейн выйти год представлять себя  
уникальный интерпретация классический история создание монстр режиссёр картина  
выступить пол макгиган сценарий написать мэри шеффера пол макгиган главный роль сняться  
известный актёр дэниэл рэдклифф роль игорь джеймс мак...

Предсказать жанр

**Предсказанный жанр: Триллеры и драмы (ID: 1)**

## Вероятности по жанрам:

- Анимационные приключения: 0.230
- Триллеры и драмы: 0.306
- Криминальные и психологические драмы: 0.157
- Научная фантастика и романтика: 0.133
- Боевики и комедии: 0.174

# Классификация фильмов по жанрам по описанию

Введите описание фильма для классификации

фобзор фильм белоснежок смотреть онлайн хороший качество год экран выйти новый адаптация классический сказка музыкальный фэнтези фильм белоснежок snow white режиссёр который стать марк уэбба это современный ремейк знаменитый мультфильм disney год вдохновить одноимённый сказка брат гримм фильм сразу в...

Предсказать жанр

**Предсказанный жанр: Криминальные и психологические драмы (ID: 2)**

## Вероятности по жанрам:

- Анимационные приключения: 0.203
- Триллеры и драмы: 0.244
- Криминальные и психологические драмы: 0.247
- Научная фантастика и романтика: 0.154
- Боевики и комедии: 0.152

# Классификация фильмов по жанрам по описанию

Введите описание фильма для классификации

романтический история сила любовь фильм клятва the vow это трогательный романтический драма основать реальный событие центр сюжет находится молодой пара пейдж рэйчел макадамс лео ченнинга татум страшный автомобильный авария пейдж впадать кто приходит обнаруживать полностью потерять память последни...

Предсказать жанр

**Предсказанный жанр: Научная фантастика и романтика (ID: 3)**

**Вероятности по жанрам:**

- Анимационные приключения: 0.191
- Триллеры и драмы: 0.203
- Криминальные и психологические драмы: 0.197
- Научная фантастика и романтика: 0.272
- Боевики и комедии: 0.136

# Классификация фильмов по жанрам по описанию

Введите описание фильма для классификации

фильм тигр живой tiger zinda hai год это захватывать индийский боевик продолжать история два супер шпион тигр зоя сюжет разворачиваться вокруг спасение индийский пакистанский медсестра взять заложник террористический организация ирак главный роль исполнять салман хан катрина каифа фильм снятой жанр ...

Предсказать жанр

**Предсказанный жанр: Боевики и комедии (ID: 4)**

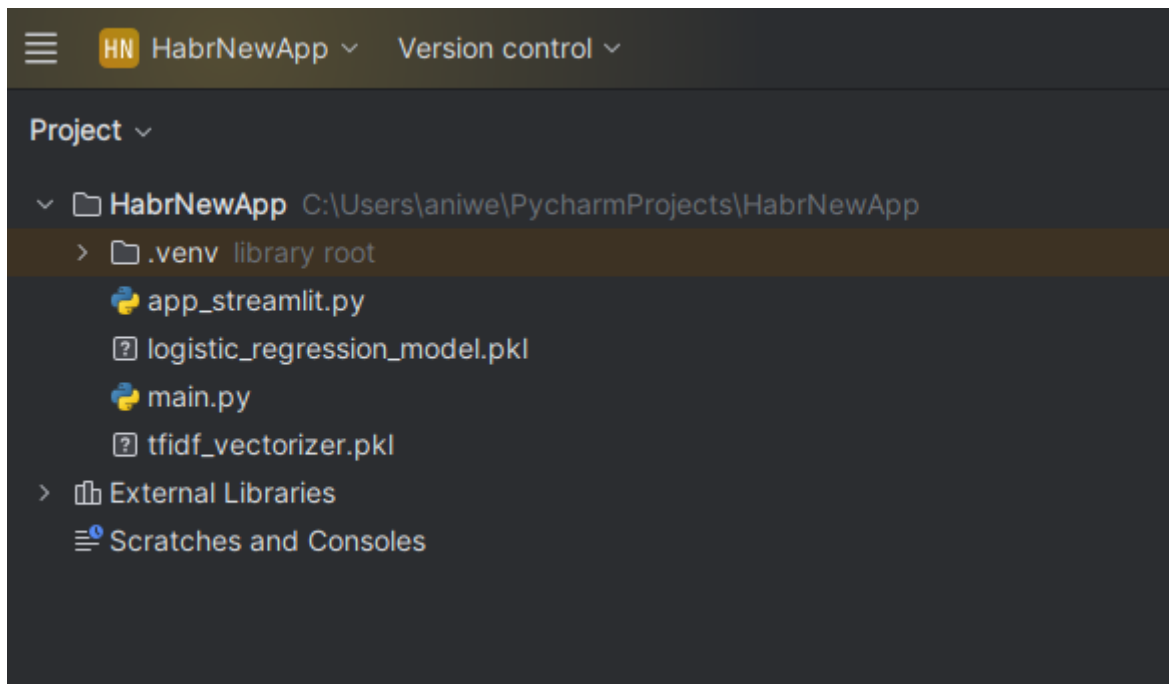
## Вероятности по жанрам:

- Анимационные приключения: 0.178
- Триллеры и драмы: 0.174
- Криминальные и психологические драмы: 0.163
- Научная фантастика и романтика: 0.133
- Боевики и комедии: 0.352

Приложение ML –

по дата-сету хабра

Структура проекта:



Для API:

FastAPI — основной фреймворк для создания REST API.

Pydantic BaseModel — используется для валидации и описания входных данных.

joblib — для загрузки сериализованных (сохранённых) модели и векторизатора.

nlTK, pymorphy3, BeautifulSoup — для предобработки текста: очистка, токенизация, удаление стоп-слов, лемматизация.

Основные классы и переменные:

```
app = FastAPI()
```

Экземпляр приложения FastAPI.

```
model = joblib.load('logistic_regression_model.pkl')
```

Загрузка обученной модели логистической регрессии.

```
vectorizer = joblib.load('tfidf_vectorizer.pkl')
```

Загрузка обученного TF-IDF векторизатора.

```
morph = pymorphy3.MorphAnalyzer()
```

Морфологический анализатор для лемматизации.

Множество русских стоп-слов для фильтрации. Словарь, сопоставляющий номер кластера с его осмысленным названием. Модель входных данных для POST-запроса, содержит одно поле — текст.

Основные функции (методы) -

Очищает текст от HTML-тегов, приводит к нижнему регистру, удаляет пунктуацию, цифры и лишние пробелы.

Токенизирует текст и удаляет стоп-слова и короткие токены.

Лемматизирует каждое слово в тексте с помощью `ru morphology`.

Последовательно применяет все этапы предобработки: очистку, токенизацию, удаление стоп-слов и лемматизацию.

Основной маршрут API

`@app.post("/predict")`

Обрабатывает POST-запросы по адресу `/predict`.

Принимает текст в формате JSON.

Применяет к тексту полную предобработку.

Векторизует текст с помощью TF-IDF.

Передаёт вектор в модель для предсказания кластера.

Возвращает JSON с:

ID кластера,

осмысленным названием кластера,

вероятностями принадлежности к каждому кластеру.

API принимает текст, обрабатывает его так же, как при обучении модели, и возвращает тематическую категорию (кластер) с вероятностями. Классы и функции четко разделяют этапы обработки: валидация, очистка, токенизация, лемматизация, векторизация, предсказание.

main.py × app\_streamlit.py

```
1 from fastapi import FastAPI
2 from pydantic import BaseModel
3 import joblib
4 import re
5 import string
6 from bs4 import BeautifulSoup
7 import nltk
8 from nltk.tokenize import word_tokenize
9 from nltk.corpus import stopwords
10 import pymorphy3
11
12 # Инициализация
13 app = FastAPI()
14 model = joblib.load('logistic_regression_model.pkl')
15 vectorizer = joblib.load('tfidf_vectorizer.pkl')
16
17 morph = pymorphy3.MorphAnalyzer()
18 nltk.download('punkt')
19 nltk.download('stopwords')
20 russian_stopwords = set(stopwords.words('russian'))
21
22 # Словарь с названиями кластеров
23 cluster_names = {
24     0: 'Ребрендинг и маркетинг',
25     1: 'Маркетплейс и доставка',
26     2: 'Техническое программирование',
27     3: 'Государственные и социальные проекты',
28     4: 'Обучение и курсы 60'
29 }
30
31 1 usage
32 class TextRequest(BaseModel):
33     text: str
34
35 1 usage
36 def clean_text_conditional(text: str) -> str:
37     if not isinstance(text, str):
38         return ""
```



```

1 usage
def clean_text_conditional(text: str) -> str:
    if not isinstance(text, str):
        return ""
    if re.search(pattern: r'<[^>]+>', text):
        text = BeautifulSoup(text, features: 'lxml').get_text()
    text = text.lower()
    text = "".join([ch if ch not in string.punctuation else " " for ch in text])
    text = "".join([ch if not ch.isdigit() else " " for ch in text])
    text = re.sub(pattern: r'\s+', repl: ' ', text).strip()
    return text

1 usage
def tokenize_and_remove_stopwords(text: str) -> str:
    tokens = word_tokenize(text, language='russian')
    filtered_tokens = [word for word in tokens if word not in russian_stopwords and len(word) > 1]
    return " ".join(filtered_tokens)

1 usage
def lemmatize_text(text: str) -> str:
    tokens = word_tokenize(text, language='russian')
    lemm_tokens = [morph.parse(token)[0].normal_form for token in tokens]
    return " ".join(lemm_tokens)

1 usage
def preprocess_text(text: str) -> str:
    text = clean_text_conditional(text)
    text = tokenize_and_remove_stopwords(text)
    text = lemmatize_text(text)
    return text

@app.post("/predict")
async def predict_cluster(request: TextRequest):
    processed_text = preprocess_text(request.text)
    vectorized = vectorizer.transform([processed_text])
    prediction = model.predict(vectorized)[0]
    probabilities = model.predict_proba(vectorized)[0]

```

```

response = {
    "cluster_id": int(prediction),
    "cluster_name": cluster_names[int(prediction)],
    "probabilities": {cluster_names[i]: float(probabilities[i]) for i in range(len(probabilities))}
}
return response

```

Для приложения:

Streamlit — библиотека для быстрого создания веб-интерфейсов на Python.

requests — библиотека для отправки HTTP-запросов к API.

Основные шаги и логика кода —

Импортируем необходимые библиотеки: Streamlit для интерфейса и requests для общения с сервером.

st.title("Классификация текста по тематическим кластерам") - Отображаем заголовок приложения.

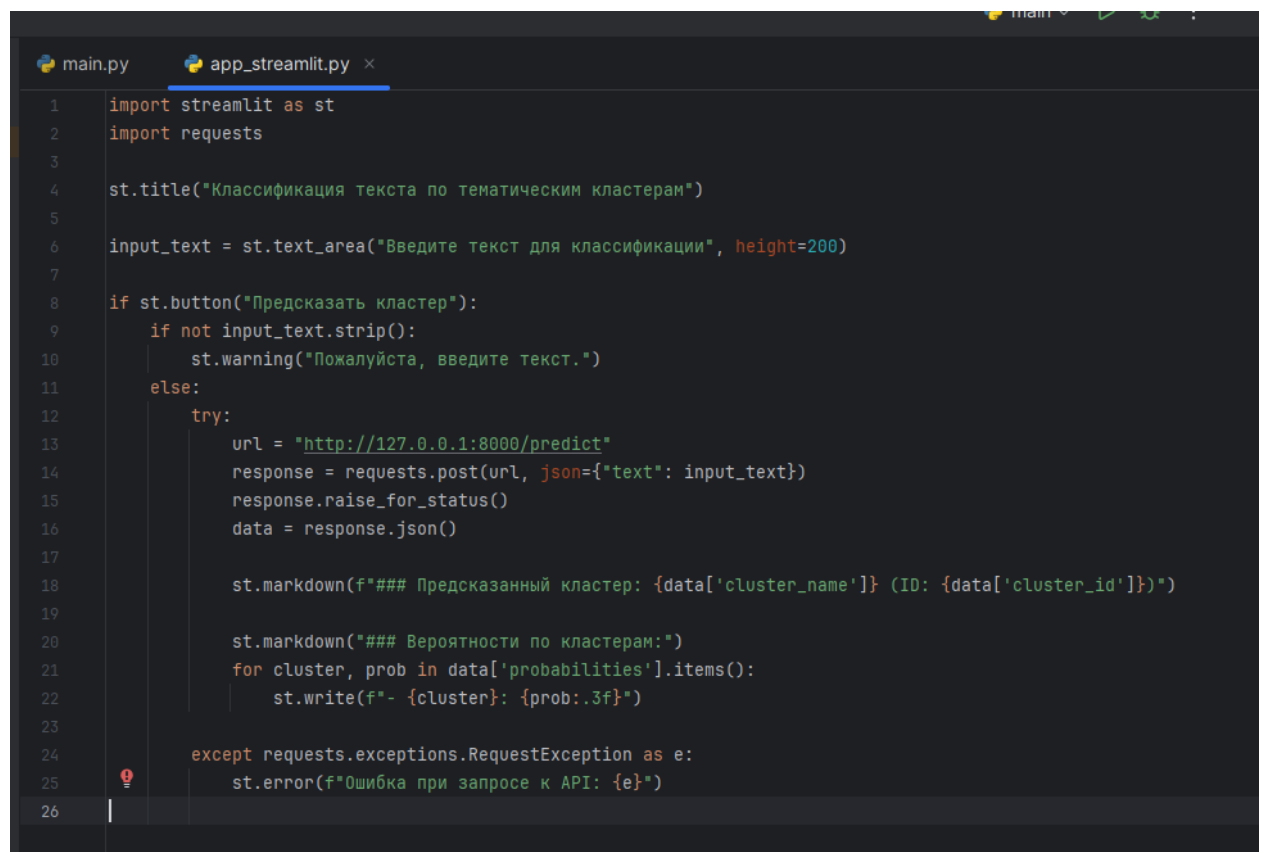
Создаём многострочное текстовое поле для ввода пользователем текста.

Отслеживаем нажатие кнопки «Предсказать кластер». Весь последующий код выполняется при нажатии.

Проверяем, что поле не пустое (без пробелов). Если пустое — показываем предупреждение. Отправляем POST-запрос на локальный API FastAPI с JSON, содержащим введённый текст.

Проверяем успешность ответа и получаем данные в формате JSON.

Выводим список вероятностей принадлежности текста к каждому из кластеров с точностью до трёх знаков. Если при запросе произошла ошибка (например, сервер не доступен), показываем сообщение об ошибке.



```
1 import streamlit as st
2 import requests
3
4 st.title("Классификация текста по тематическим кластерам")
5
6 input_text = st.text_area("Введите текст для классификации", height=200)
7
8 if st.button("Предсказать кластер"):
9     if not input_text.strip():
10         st.warning("Пожалуйста, введите текст.")
11     else:
12         try:
13             url = "http://127.0.0.1:8000/predict"
14             response = requests.post(url, json={"text": input_text})
15             response.raise_for_status()
16             data = response.json()
17
18             st.markdown(f"### Предсказанный кластер: {data['cluster_name']} (ID: {data['cluster_id']})")
19
20             st.markdown("### Вероятности по кластерам:")
21             for cluster, prob in data['probabilities'].items():
22                 st.write(f"- {cluster}: {prob:.3f}")
23
24         except requests.exceptions.RequestException as e:
25             st.error(f"Ошибка при запросе к API: {e}")
26
```

Проверка работы приложения —

# Классификация текста по тематическим кластерам

Введите текст для классификации

сегодня запустить проект tagliner циничный пародия весь известный рейтинг многий другой посчитать обиженный публикация результат тэглайна поэтому решить дать возможность другой оказаться первый место помощь тэглайнер высокий строчка оказаться любой веб студия достаточно заполнить нехитрый форма рез...

Предсказать кластер

**Предсказанный кластер: Ребрендинг и маркетинг (ID: 0)**

## Вероятности по кластерам:

- Ребрендинг и маркетинг: 0,367
- Маркетплейс и доставка: 0,203
- Техническое программирование: 0,145
- Государственные и социальные проекты: 0,165
- Обучение и курсы Go: 0,120

# Классификация текста по тематическим кластерам

Введите текст для классификации

часть вступление современный мир высокий технология осваивать человек желать достигнуть  
полезный большой окружающий делать наш среда комфортный гармоничный тот сознательно  
заниматься разрушение делать человек несчастный дать статья описать реальный история  
стык криминал высокий технология жадность г...

Предсказать кластер

**Предсказанный кластер: Маркетплейс и доставка (ID: 1)**

**Вероятности по кластерам:**

- Ребрендинг и маркетинг: 0,168
- Маркетплейс и доставка: 0,347
- Техническое программирование: 0,132
- Государственные и социальные проекты: 0,207
- Обучение и курсы Go: 0,145

# Классификация текста по тематическим кластерам

Введите текст для классификации

вывод лог это столкнуться нужно конвертировать значение переменный строка это обычно  
делаться вывод поток вариант использование boost lexical cast наш случай практически один  
встроить тип это проблема нужно вывести скажем std vector увы std vector оператор вывод  
поток результат решение проблема напи...

Предсказать кластер

**Предсказанный кластер: Техническое программирование (ID: 2)**

**Вероятности по кластерам:**

- Ребрендинг и маркетинг: 0,168
- Маркетплейс и доставка: 0,226
- Техническое программирование: 0,335
- Государственные и социальные проекты: 0,168
- Обучение и курсы Go: 0,103

# Классификация текста по тематическим кластерам

Введите текст для классификации

компания gett смело смотреть будущее основатель компания сахара вайсер рассчитывать  
утроить оборот онлайн сервис вызов такси год год прогноз вайсер конец годовой оборот  
составить миллиард весь мир оборот утраиваться каждый год начинать момент создание  
компания год заявить господин вайсер считать реа...

Предсказать кластер

**Предсказанный кластер: Государственные и социальные проекты (ID: 3)**

**Вероятности по кластерам:**

- Ребрендинг и маркетинг: 0.119
- Маркетплейс и доставка: 0.119
- Техническое программирование: 0.096
- Государственные и социальные проекты: 0.542
- Обучение и курсы Go: 0.123

# Классификация текста по тематическим кластерам

Введите текст для классификации

июнь москва пройти mydribbble meetup неформальный конференция дизайнер выступить  
полтора десяток сильный начинающий специалист многие который представленный один  
главный тематический социальный сеть dribbble создатель инструмент principle sympli георгий  
квасник fantasy олег береснев beresnev design ...

Предсказать кластер

**Предсказанный кластер: Обучение и курсы Go (ID: 4)**

## Вероятности по кластерам:

- Ребрендинг и маркетинг: 0.197
- Маркетплейс и доставка: 0.127
- Техническое программирование: 0.109
- Государственные и социальные проекты: 0.245
- Обучение и курсы Go: 0.322