

# Классификация текста

**SBER** 

---

telegram: @igorec\_matveev

02

## Постановка задачи

- Написать классификатор, который по входным параметрам будет определять целевой класс
- Оценить качество алгоритма
- Описать предложенные алгоритмы
- Применить Gradient Boosting

## 03

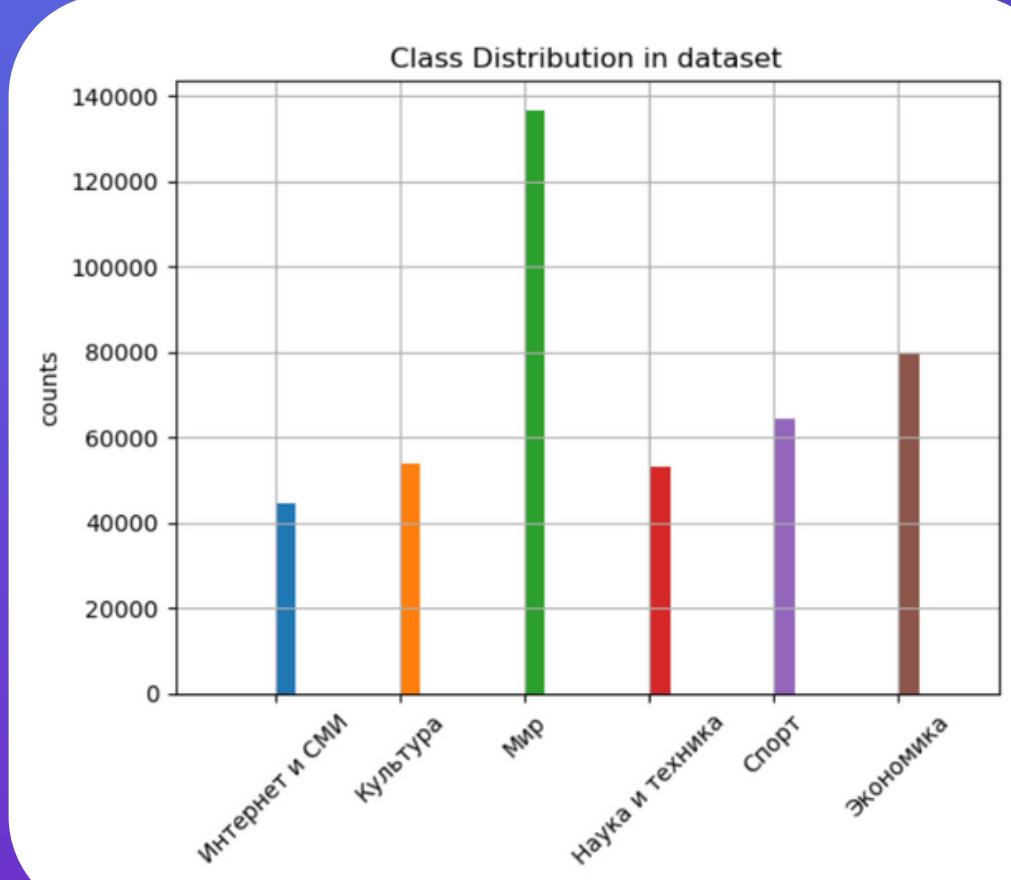
# Обработка текста. Embeddings

- Для выделения признаков применим алгоритм **Term Frequency – Inverse Document Frequency (TFIDF)**
- **Идея:** наибольший вес получают слова, которые часто встречаются в тексте (документе), но при этом они достаточно редкие в целом, то есть в коллекции текстов (документов)
- На вход классификатору будем подавать блок текста из колонки "**text**", так как он более информативен

04

# Оценка распределения классов

- Мир: 136621
- Экономика: 79528
- Спорт: 64413
- Культура: 53797
- Наука и техника: 53136
- Интернет и СМИ: 44663



05

# Оценка расположения кластеров

- Используя библиотеку **UMAP**, отобразим объекты на плоскости
- Видно, что данные линейно плохо разделимы
- Для решения задачи обучим набор нелинейных классификаторов: **Decision Tree**, **Random Forest**, **logistic Regression**

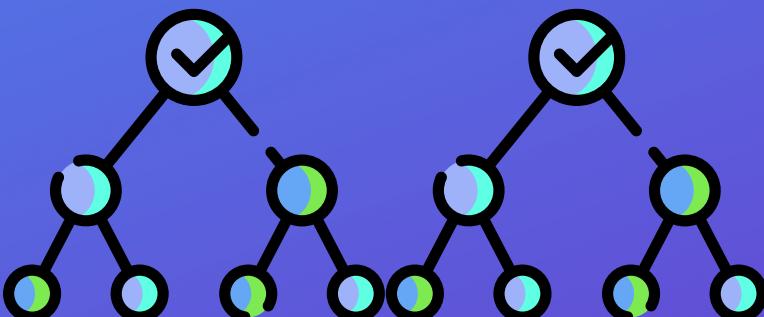


**06**

# Обучение алгоритмов



Decision Tree



Random Forest



logistic Regression

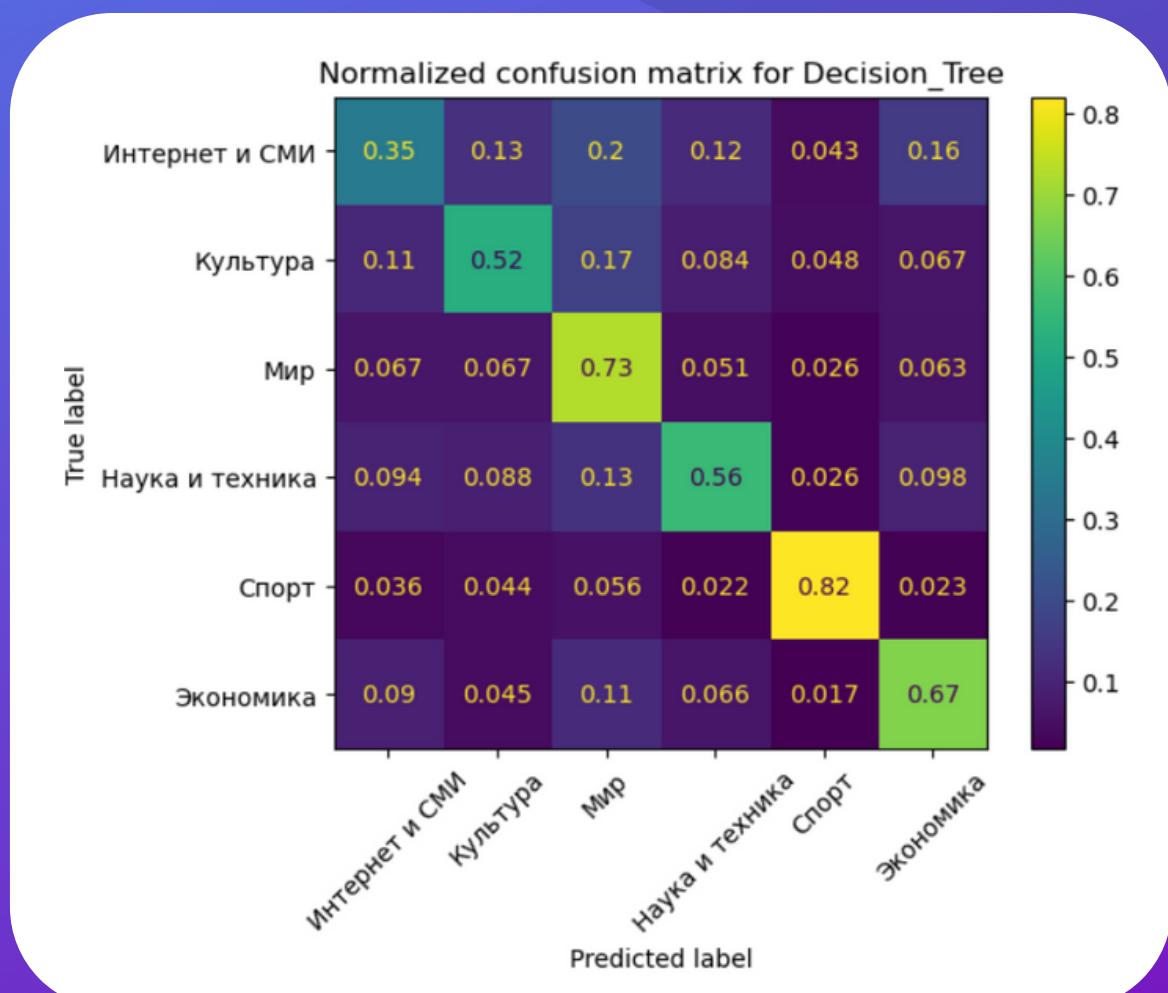


Gradient Boosting

07

# Тестирование. Decision Tree

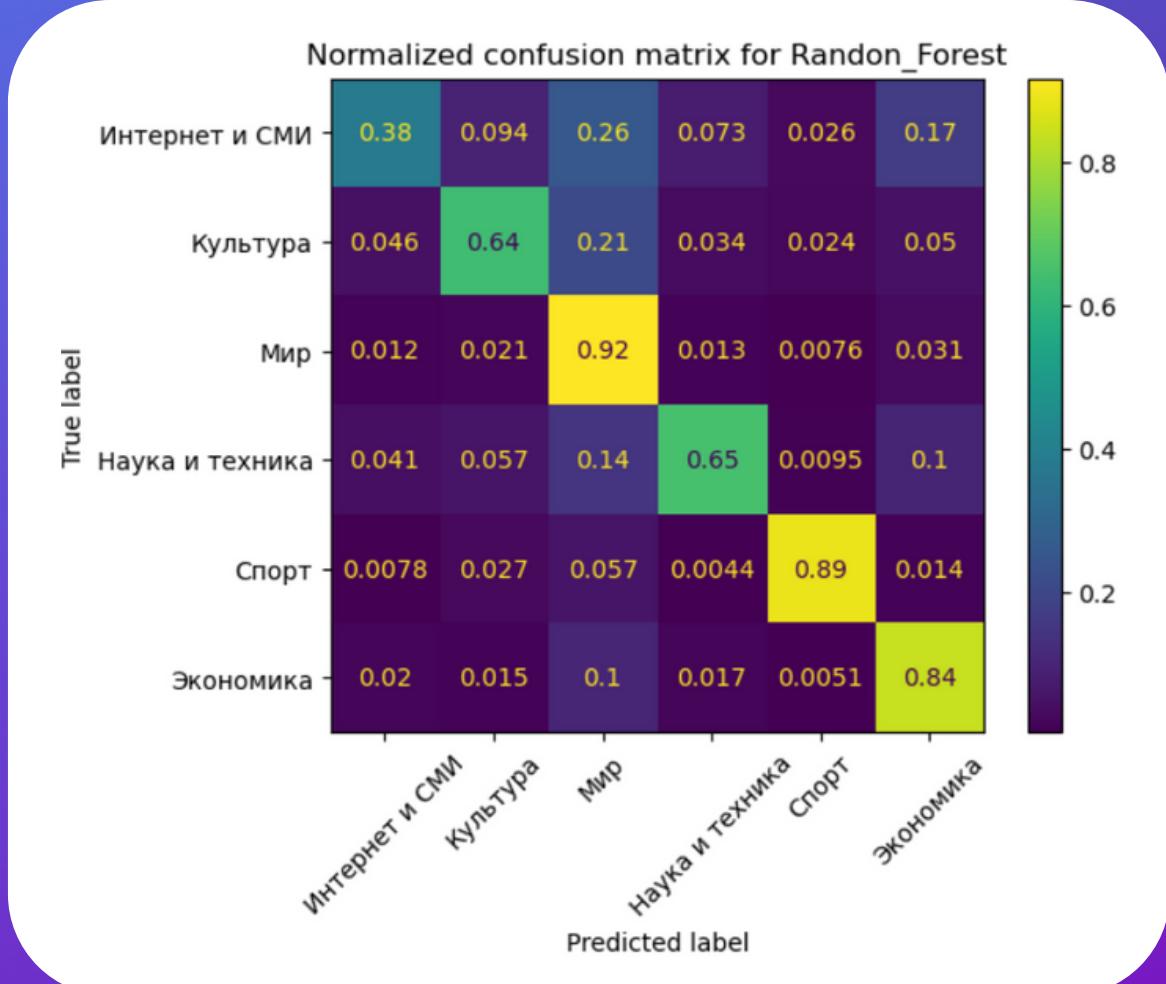
- Accuracy: 0.643
- F1-Score: 0.607
- Confusion Matrix



08

# Тестирование. Random Forest

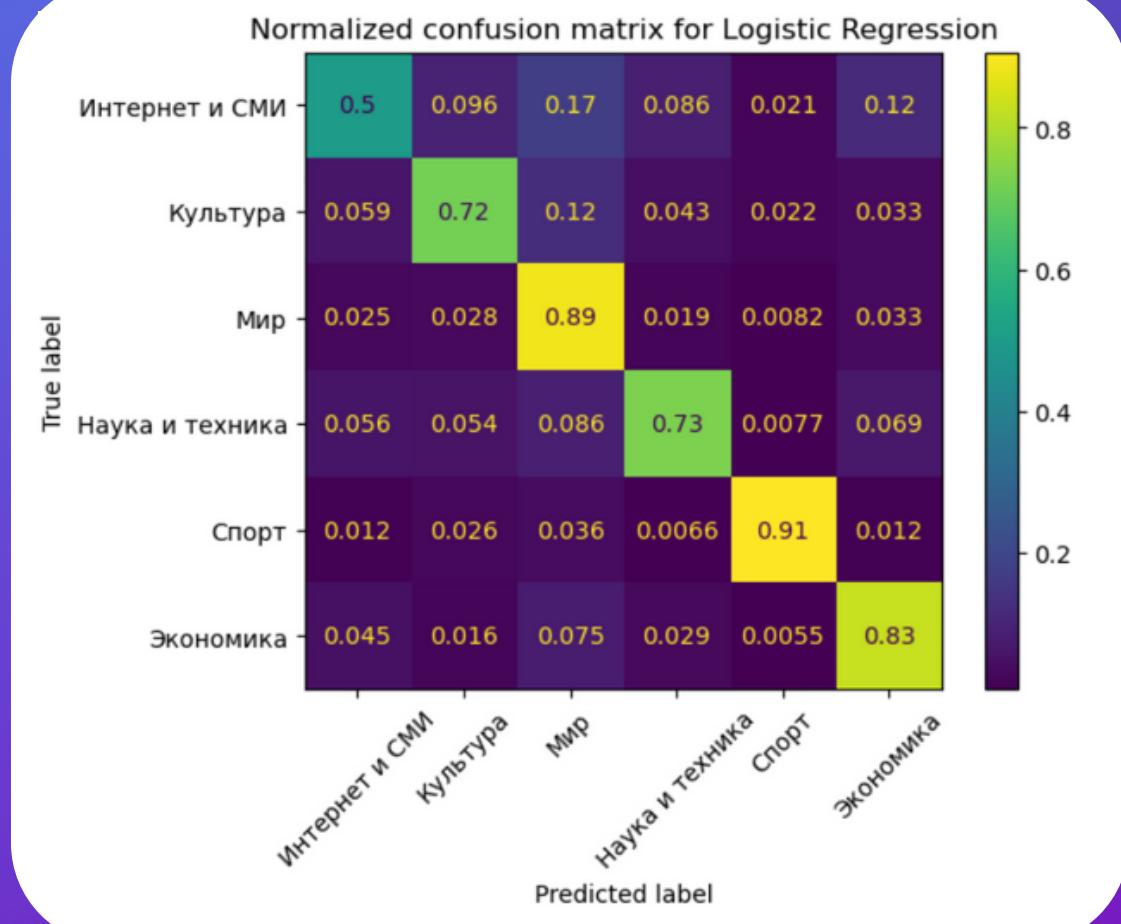
- Accuracy: 0.775
- F1-Score: 0.736
- Confusion Matrix



09

# Тестирование. Logistic Regression

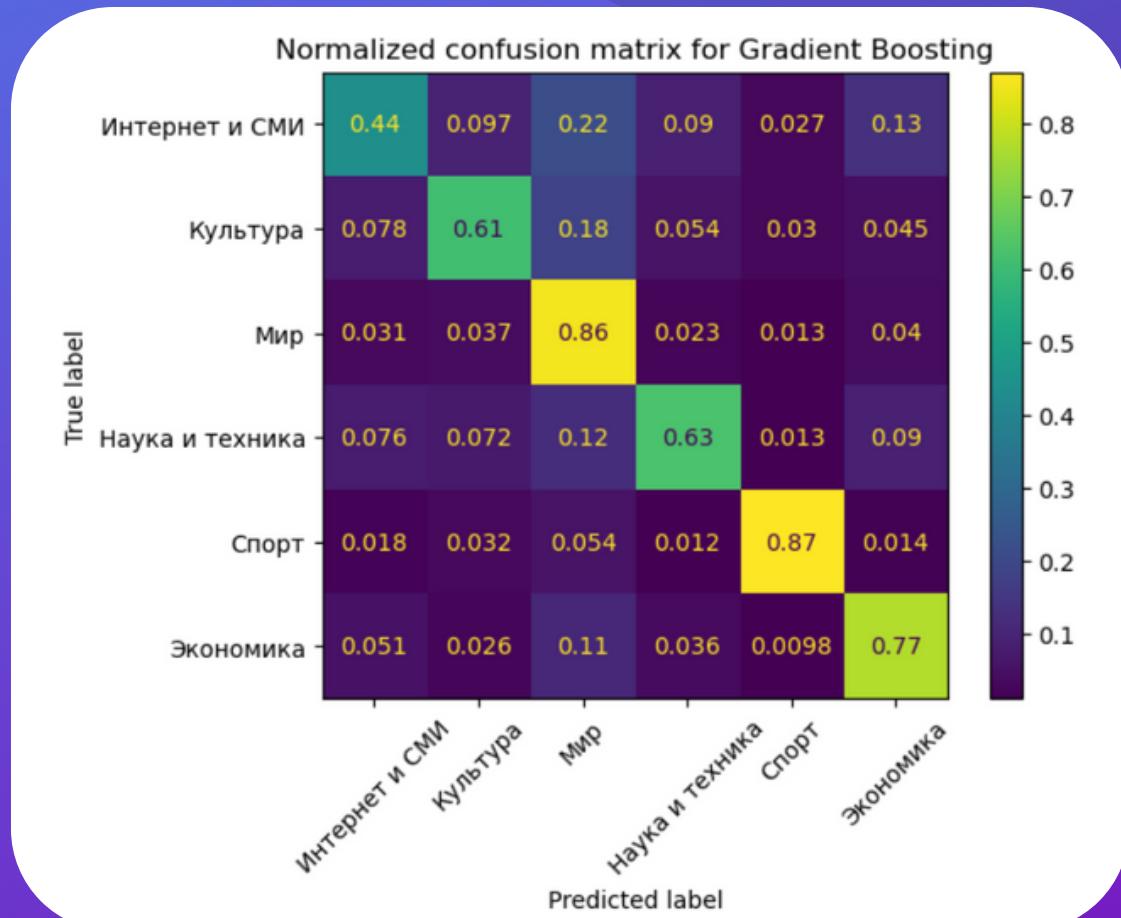
- Accuracy: 0.799
- F1-Score: 0.77
- Confusion Matrix



10

# Тестирование. Gradient Boosting

- Accuracy: 0.74
- F1-Score: 0.705
- Confusion Matrix



11

# Лучший классификатор

- Accuracy: 0.799
- F1-Score: 0.77



logistic Regression



**Матвеев Игорь**

telegram: @igorec\_matveev

**Спасибо за внимание!**