MIPT
MOSCOW INSTITUTE OF PHYSICS AND TECHNOLOGY

АТП кафедра Алгоритмов и Технологий Программирования

ВЫСШАЯ ШКОЛА ЭКОНОМИКИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

Автоматическая обработка текстов
Классификация текстов

Лекция 4

Емельянов А. А.
login-const@mail.ru

2018

https://github.com/king-menin/nlp-course

Постановка задачи классификации

- **Дано:**
  - $d \in D$ – документы,
  - $c \in C$ – классы,

- **Требуется:**
  - Отнести неизвестный документ к одному из классов $c \in C, C = \{0,1\}$ – **бинарная классификация**.

Постановка задачи классификации

- **Дано:**
  - $d \in D$ – документы,
  - $c \in C$ – классы,

- **Требуется:**
  - Отнести неизвестный документ к одному из классов $c \in C, C = \{0,1\}$ – **бинарная классификация**.
  - Отнести неизвестный документ к одному из классов с $\in C, C = \{0,1, \dots K-1\}$ – **многотемная классификация**.

Постановка задачи классификации

- **Дано:**
  - $d \in D$ – документы,
  - $c \in C$ – классы,

- **Требуется:**
  - Отнести неизвестный документ к одному из классов $c \in C, C = \{0,1\}$ – **бинарная классификация**.
  - Отнести неизвестный документ к одному из классов с $\in C, C = \{0,1, \dots K-1\}$ – **многотемная классификация**.
  - Отнести неизвестный документ к нескольким классам $c_i, i < K \in C, C = \{0,1, \dots K-1\}$ – **многотемная классификация**.

Примеры задач классификации текстов

- **Фильтрация спама**: $C = \{spam, ham\}$ – бинарная классификация

Примеры задач классификации текстов

- **Фильтрация спама**: $C = \{spam, ham\}$ – бинарная классификация

- **Классификация по тональности**: $C = \{neutral, positive, negative\}$ – классификация с тремя классами.

Примеры задач классификации текстов

- **Фильтрация спама**: $C = \{spam, ham\}$ – бинарная классификация
- **Классификация по тональности**: $C = \{neutral, positive, negative\}$ – классификация с тремя классами.
- **Рубрикация**: $C \in \{\text{религия, праздники, спорт, фестивали, ...}\}$ – классификация на несколько тем.

Примеры задач классификации текстов

- **Фильтрация спама**: $C = \{spam, ham\}$ – бинарная классификация

- **Классификация по тональности**: $C = \{neutral, positive, negative\}$ – классификация с тремя классами.

- **Рубрикация**: $C \in \{\text{религия}, \text{праздники}, \text{спорт}, \text{фестивали}, \dots\}$ – классификация на несколько тем.

- **Определение авторства**:

  –Этим ли автором написан текст: $C = \{0,1\}$?

  –Кем из этих авторов написан текст: $C = \{a1, a2, a3, \dots\}$?

  –Пол автора: $C = \{f, m\} C = \{f, m\}$.

Метрики качества классификации

- **Accuracy –** в простейшем случае такой метрикой может быть доля документов по которым классификатор принял правильное решение.

$$Accuracy = \frac{P}{N}$$

- Тем не менее, у этой метрики есть одна особенность которую необходимо учитывать. Она присваивает всем документам одинаковый вес, что может быть не корректно в случае если распределение документов в обучающей выборке сильно смещено в сторону какого-то одного или нескольких классов.

Метрики качества классификации

Категория i		Экспертная оценка	
		Положительная	Отрицательная
Оценка системы	Положительная	TP	FP
	Отрицательная	FN	TN

- *TPTP* — истино-положительное решение;
- *TNTN* — истино-отрицательное решение;
- *FPFP* — ложно-положительное решение;
- *FNFN* — ложно-отрицательное решение.
- **Точность** (precision) **–** точность системы в пределах класса – это доля документов действительно принадлежащих данному классу относительно всех документов которые система отнесла к этому классу.

$$Precision = \frac{TP}{TP + FP}$$

Метрики качества классификации

Категория i		Экспертная оценка	
		Положительная	Отрицательная
Оценка системы	Положительная	TP	FP
	Отрицательная	FN	TN

- *TPTP* — истино-положительное решение;
- *TNTN* — истино-отрицательное решение;
- *FPFP* — ложно-положительное решение;
- *FNFN* — ложно-отрицательное решение.
- **Точность** (precision) **–** точность системы в пределах класса – это доля документов действительно принадлежащих данному классу относительно всех документов которые система отнесла к этому классу.

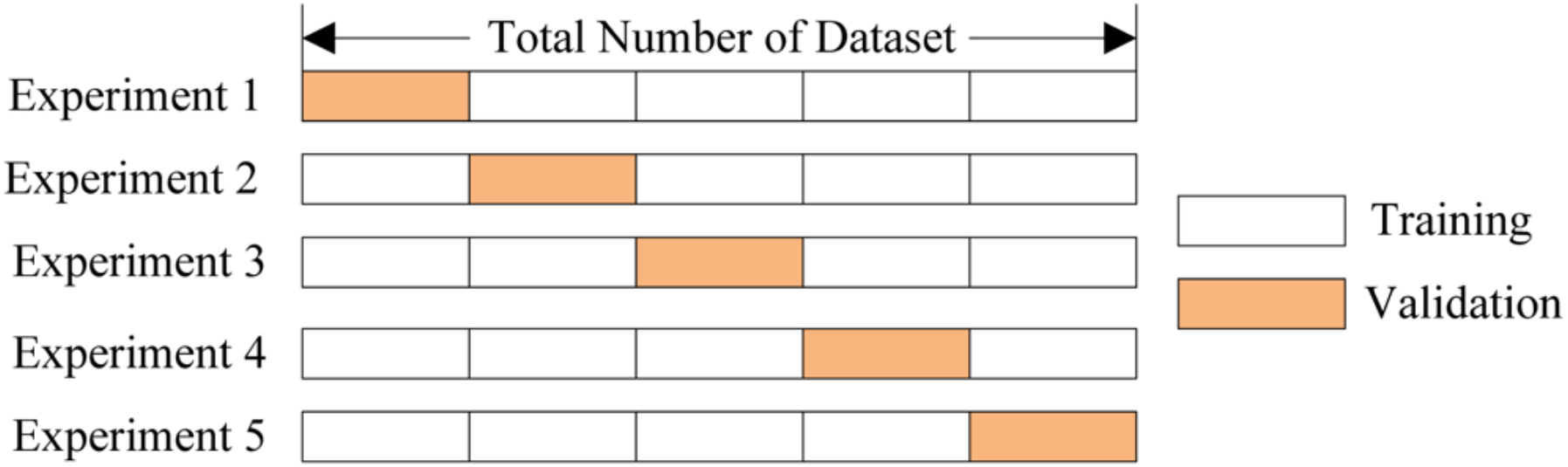
$$Precision = \frac{TP}{TP + FP}$$

- **Полнота** (recall) – это доля найденных классификатором документов принадлежащих классу относительно всех документов этого класса в тестовой выборке.

$$Recall = \frac{TP}{TP + FN}$$

Кроссвалидация

- **Скользящий контроль** или **кросс-проверка** или **кросс-валидация** (cross-validation, CV) — процедура эмпирического оценивания обобщающей способности алгоритмов, обучаемых по прецедентам.
  - Фиксируется некоторое множество разбиений исходной выборки на две подвыборки: *обучающую* и *контрольную*. Для каждого разбиения выполняется настройка алгоритма по обучающей подвыборке, затем оценивается его средняя ошибка на объектах контрольной подвыборки. *Оценкой скользящего контроля* называется средняя по всем разбиениям величина ошибки на контрольных подвыборках.



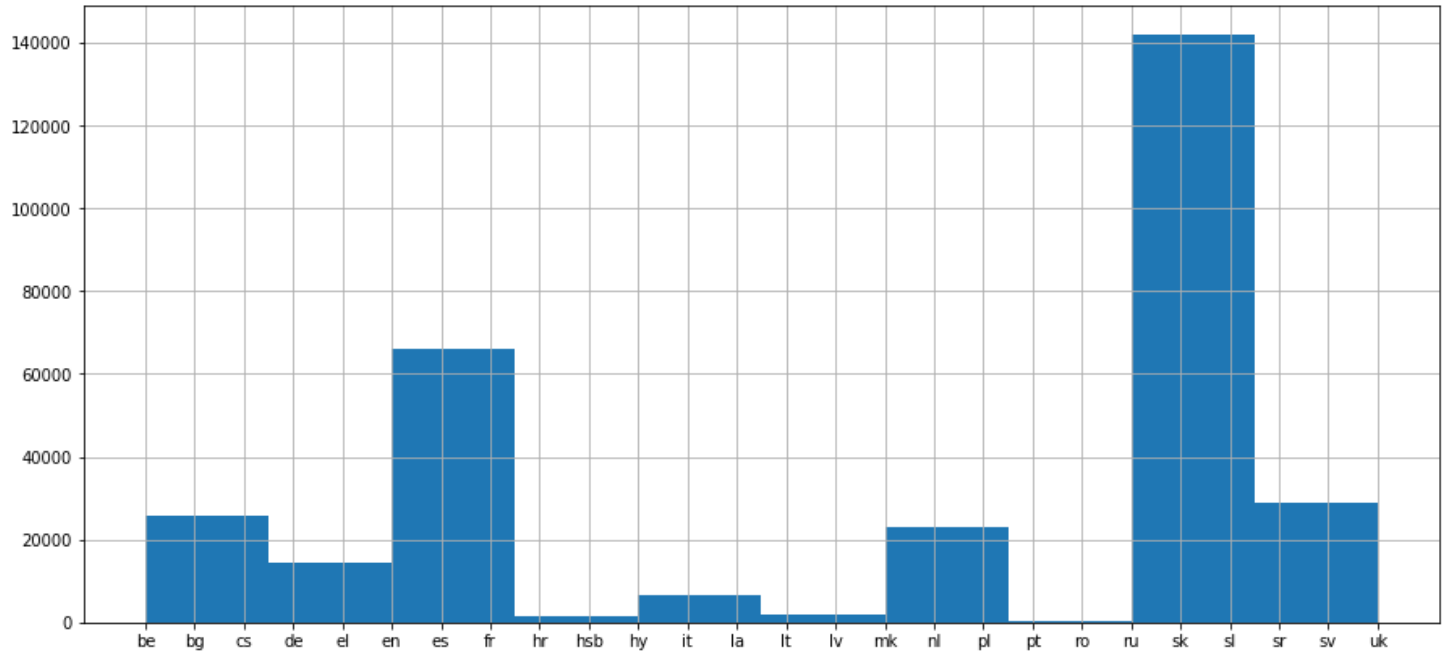
Постановка задачи (для примера)

- Рассмотрим задачу определения языка текста.
- Всего 26 языков.
- Обучающая выборка имеет вид:

	lang	text
267430	ru	Приближался рассвет. ГЛАВА ШЕСТАЯ В большом с...
182935	ru	Черт возьми, мне сказали, что вы арестованы, и...
214193	ru	Лишь первый маленький шаг на пути к нему, но н...
240204	en	Tom heard his mother's voice, the remembered c...
145677	ru	Мало, ох мало же вы понимаете в политике, пан ...

Гистограмма классов

- Какие могут быть трудности при работе с этой выборкой?
- Как лучше разбить выборку на обучающую и тестовую?



Разбиение выборки

- Какие могут быть трудности при работе с этой выборкой?
- Как лучше разбить выборку на обучающую и тестовую?

```
from sklearn.model_selection import StratifiedShuffleSplit
sss = StratifiedShuffleSplit(n_splits=1, test_size=0.2, random_state=0)
train_index, test_index = next(sss.split(data.text, data.lang))
train_data = data.iloc[train_index]
test_data = data.iloc[test_index]
```

Классификация на правилах

- Какие можно придумать правила, для определения языка в выборке?

- Протестируйте простой принцип классификации на правилах, например:
  - Если в тексте встречается буква "э", то текст относится к классу "ru" (написан на русском, то есть)
  - Если в тексте встречается буква "ł", то текст относится к классу "pl" (написан на польском, то есть)
  - И любое другое правило, которое кажется вам разумным.

Классификация на правилах: метрики качества

- Какова точность, полнота, аккуратность при использовании этих правил?

Классификация на правилах: метрики качества

- Какова точность, полнота, аккуратность при использовании этих правил?
- Причины низкого качества при классификации на правилах?

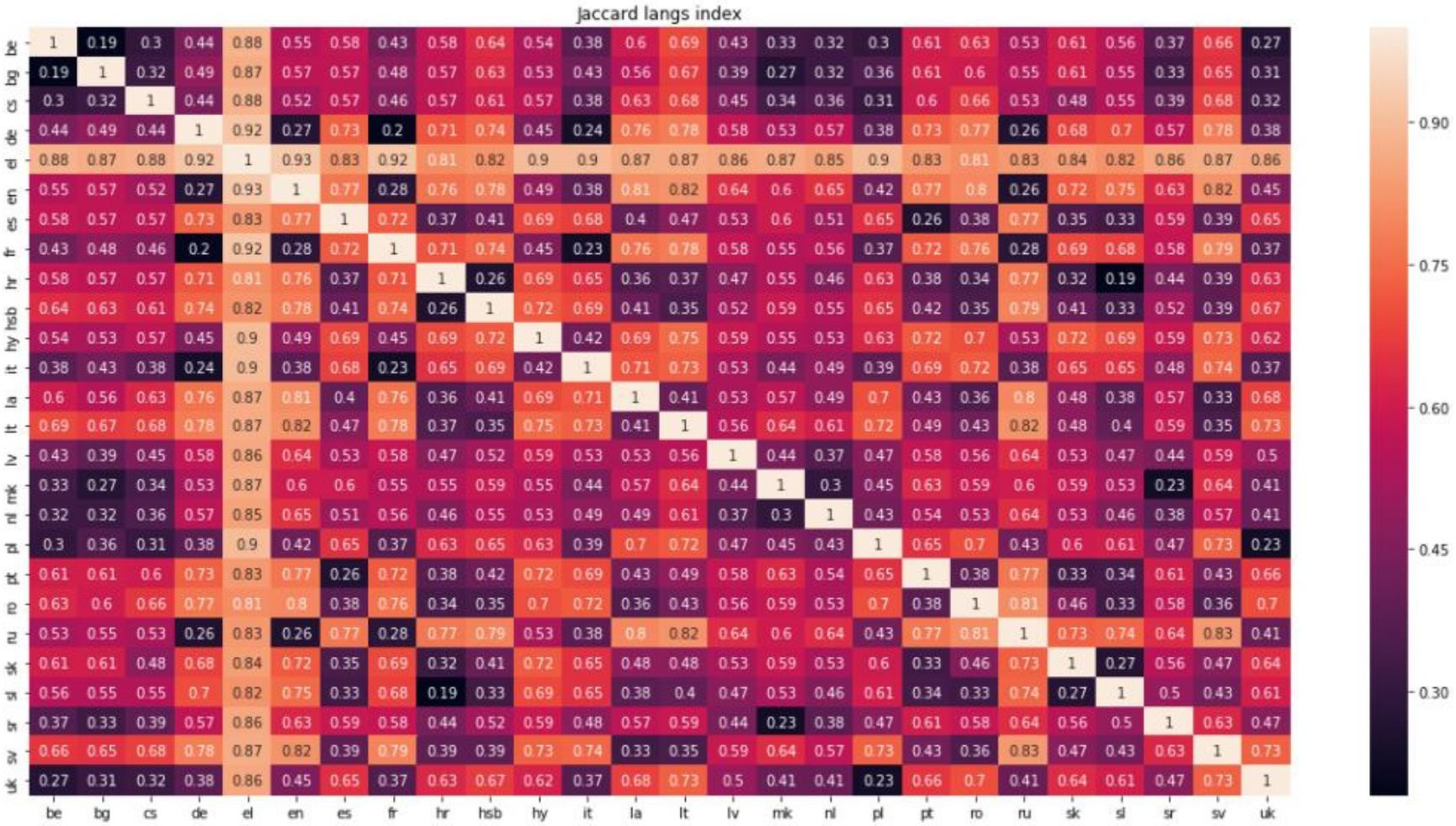
Близость языков

- Определите сходство языков. Для этого постройте heatmap коэффициентов Жаккара.

$$K_{1,-1} = \frac{n(A \cap B)}{n(A) + n(B) - n(A \cap B)} = \frac{n(A \cap B)}{n(A \cup B)}$$

Близость языков

- heatmap коэффициентов Жаккара



Метод наивного байеса

- Используйте метод наивного Байеса для классификации текстов: в качестве признаков используйте символьные n-граммы.

Метод наивного байеса

- Используйте метод наивного Байеса для классификации текстов: в качестве признаков используйте символьные n-граммы.

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.pipeline import Pipeline

clf = Pipeline([
        ('vect', CountVectorizer(ngram_range=(2, 2), analyzer="char")),
        ('clf', MultinomialNB()),
    ])
```

Метод наивного байеса

- Обучим модель и посмотрим на качество

```
%%time
clf.fit(train_data.text, train_data.lang)
```

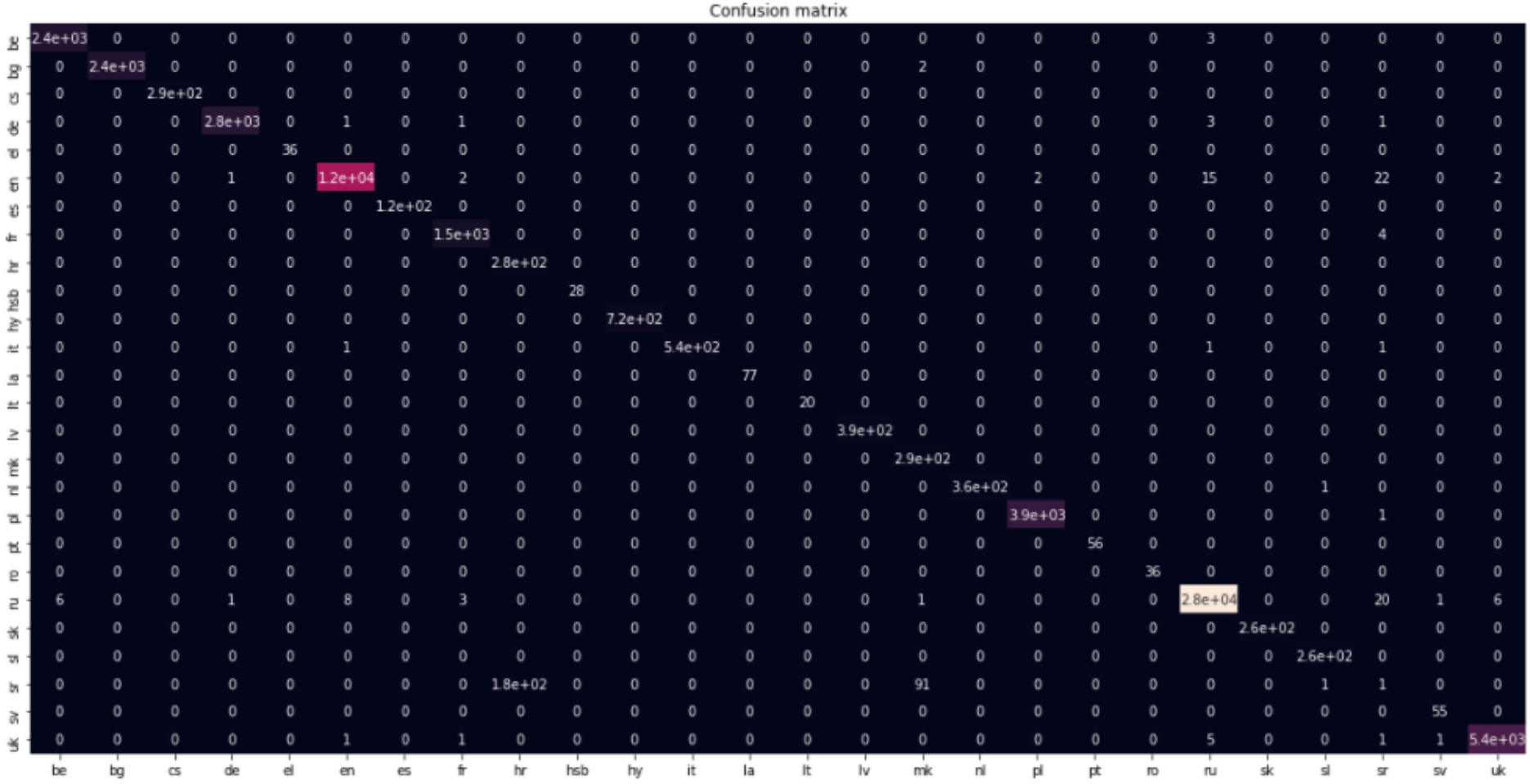
```
CPU times: user 1min 1s, sys: 1.1 s, total: 1min 2s
Wall time: 1min 2s

Pipeline(memory=None,
     steps=[('vect', CountVectorizer(analyzer='char', binary=False, decode_error='strict',
        dtype=<class 'numpy.int64'>, encoding='utf-8', input='content',
        lowercase=True, max_df=1.0, max_features=None, min_df=1,
        ngram_range=(2, 2), preprocessor=None, stop_words=None,
        strip_accents=None, token_pattern='(?u)\\b\\w\\w+\\b',
        tokenizer=None, vocabulary=None)), ('clf', MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True))])
```

```
predictions = clf.predict(test_data.text)
```

```
print("Precision: {0:6.5f}".format(precision_score(test_data.lang, predictions, average='macro')))
print("Recall: {0:6.5f}".format(recall_score(test_data.lang, predictions, average='macro')))
print("F1-measure: {0:6.5f}".format(f1_score(test_data.lang, predictions, average='macro')))
print("Accuracy: {0:6.5f}".format(accuracy_score(test_data.lang, predictions)))
```

```
Precision: 0.93564
Recall: 0.96081
F1-measure: 0.94549
Accuracy: 0.99368
```

Метод наивного байеса: матрица ошибок



Fasttext

- Архитектура нейронной сети похожа на CBOW [Mikolov et al. (2013)]
- Признаками могут быть не только слова, но и nn -грамы
- Входной слой нейронной сети: вектора слов (или nn -грам)
- На скрытом слое вектора слов суммируются и усредняются
- На выходном слое – softmaxsoftmax (или иерархический softmaxsoftmax)
- Полученное представление текста может быть переиспользовано в других задачах.

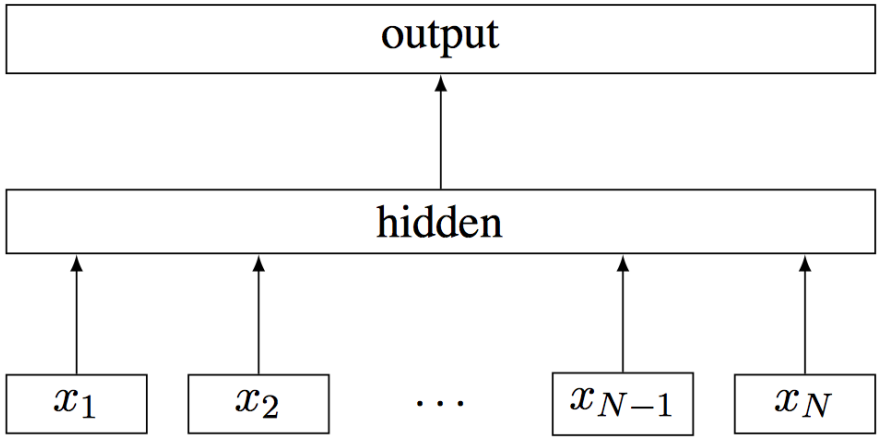


Figure 1: Model architecture of `fastText` for a sentence with N ngram features $x_1, \ldots, x_N$. The features are embedded and averaged to form the hidden variable.

1. Bag of Tricks for Efficient Text Classification – .2016. – URL: https://arxiv.org/pdf/1607.01759.pdf

Fasttext

- **Реализации:**
  - FastText
  - keras

Fasttext: формат данных

- Используйте FastText (1).
- Структура файла, который принимает fasttext должна быть следующего вида:
  - __label__ru Привет. Доброе утро. Пути народаа.
    - где "ru" метка текста "Привет. Доброе утро. Пути народаа."
- Несколько меток для одного текста можно задать так:
  - __label__ru __label__en Good Morning. Дорoy.

1. Bag of Tricks for Efficient Text Classification – .2016. – URL: https://arxiv.org/pdf/1607.01759.pdf

Fasttext

- Подготовьте данные для fasttext и запустите обучение и тест

```
%%time
ftclf = fasttext.supervised("train_data_ft.txt", "model")
```

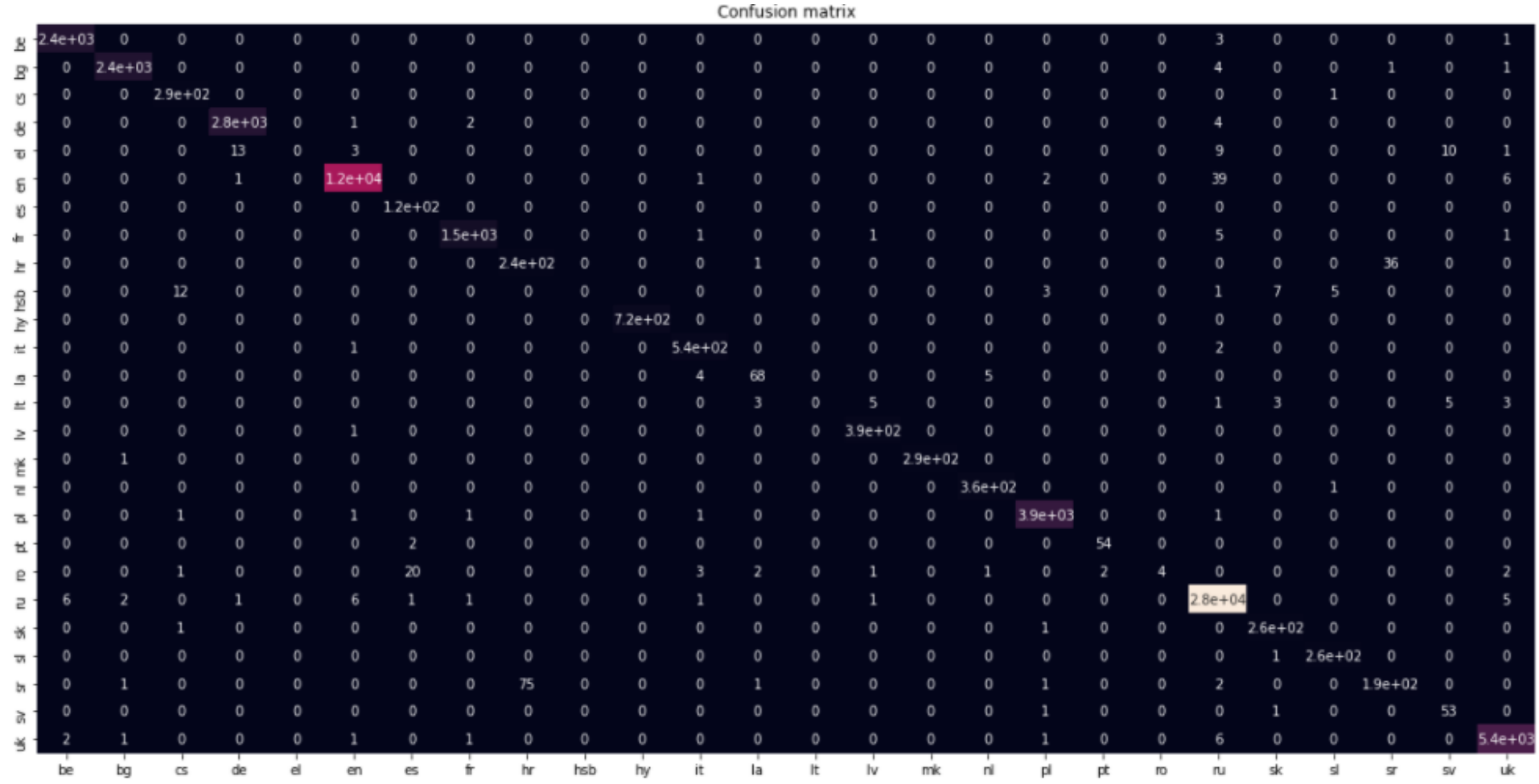
```
prc_for_fasttext(test_data, "test_data_ft.txt")
```

```
result = ftclf.test("test_data_ft.txt")
```

```
print("Precision: {0:6.5f}".format(result.precision))
print("Recall: {0:6.5f}".format(result.recall))
```

```
Precision: 0.99515
Recall: 0.99515
```

Fasttext: матрица ошибок



Что еще?

- Линейная регрессия
- Методы наивного байеса
- Машина опорных векторов
- Деревья решений и случайный лес
- FastText
- Нейронные сети

СПАСИБО ЗА ВНИМАНИЕ