

СОФИЙСКИ УНИВЕРСИТЕТ „СВ. КЛИМЕНТ ОХРИДСКИ“

ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА

КУРСОВ ПРОЕКТ

ПО СИСТЕМИ, ОСНОВАНИ НА ЗНАНИЯ

Тема:

Разпознаване на авторство

Студенти:

Ани Иванова Божкова, 8MI0700126

Стефан Стефанов Иванов, 0MI0700157

София, януари 2025 г.

1. **Формулировка на задачата**

Даден е набор от текстови файлове, съдържащи произведения на различни автори (в нашия случай Иван Вазов, Алеко Константинов и Йордан Йовков). Програмата трябва да обработи текстовите данни и да извлече характеристики на стиловете на авторите; да обучи класификатор върху тези данни и да разпознае авторството на нов, неизвестен текст въз основа на обучените модели.

Системата трябва да поддържа работа с големи текстове (напр. романи) и множество файлове.

Текстовете се обработват, за да се извлекат ключови характеристики като честота на думи, употреба на пунктуация, средна дължина на изречения и др. Всички текстове трябва да бъдат предварително почистени от специални символи и форматирани.

Очаква се програмата да демонстрира висока точност при разпознаването на авторството на текстове, при условие че тренировъчните данни са достатъчно представителни.

Програмата трябва да може:

* Да зарежда тренировъчните данни от зададената структура на директориите.
* Да обучава модела с висока ефективност.
* Да разпознава правилно автора на нови текстове с точност, надвишаваща 80%.

1. **Използвани алгоритми**

За решаването на задачата е използван Наивен Бейсов класификатор (Naive Bayes Classifier), който е подходящ за задачи, свързани с текстова класификация. Основните стъпки и методи, които са реализирани в програмата, са следните:

* Предварителна обработка на текстовете - за ефективно обучение и класификация на текстовете те биват „почистени“ (т.е. премахват се специални символи, числа и пунктуационни знаци, които не са от значение за анализа.), нормализирани (думите се сменят в малки букви) и токенизирани (текста се разделя на отделни думи (токени))
* Екстракция на характеристики (Feature Extraction) в контекста на честоти на думи и документи, което е основа за класификацията на автори

След това започва обучението на програмата с Наивния Байесов алгоритъм, който изчислява вероятността даден текст да принадлежи на конкретен автор въз основа на предположения. Приема се, че всяка дума в текста е независима от останалите.

След обучението, моделът класифицира нови текстове чрез изчисляване на вероятността за принадлежност към всеки автор. Текстът се асоциира с автора, за когото вероятността е най-голяма.

Използвани технологии и библиотеки:

* **Java**: Основният програмен език, използван за реализиране на програмната логика, структуриране на данните и работа с файлове.
* **Spring Boot**: Фреймуърк, използван за стартиране и управление на програмата. Той предоставя лесна конфигурация и инжектиране на зависимости, което улеснява структурата на проекта и работата със сървисите TrainerService и RecognitionService.
* **Jackson**: Библиотека за работа с JSON. Използва се за сериализация и десериализация на данните на модела в класовете TrainerService и RecognitionService. Тя позволява четене и запис на JSON файлове, които съхраняват информацията за модела.
* **Java** **NIO** **(java.nio.file)**: За обработка на файлове и директории, включително четене на текстови файлове и запис на резултати. Използва се в класа FileUtil.
* **Gradle**: Инструменти за управление на зависимости и изграждане на проекта. Те се използват за конфигуриране на Spring Boot и включване на необходимите библиотеки.
* **Custom Naive Bayes Implementation**: Алгоритъмът е реализиран ръчно, без използване на външни библиотеки.

Основните литературни източници за реализираните алгоритми са: Презентациите от курса по СОЗ за Наивния Бейсов класификатор, Text Classification Algorithms, Principles of Data Mining и Stack Overflow.

1. **Описание на програмната реализация**

Програмата е организирана в следната структура:

* Пакет org.example.recognition.model
  + Съдържа класа **ModelData**, който представлява данните на модела за авторско разпознаване. Класът съхранява информация за честотите на думи за всеки автор, общия брой думи, броя документи, размера на речника и други параметри.
* Пакет org.example.recognition.service
  + **TrainerService**: Отговаря за обучението на модела. Чете текстовите файлове на авторите, извлича думи, създава речник и изчислява вероятности. Моделът се основава на подходи от Наивния Бейс за текстова класификация, които използват вероятностни разпределения за предсказване.

Моделът използва условни вероятности за думите, базирани на техния контекст (автор и жанр).

Уравнение:



Където P(text∣author) се изчислява чрез вероятностите на думите в текста, а P(author) е относителната честота на автора в тренировъчните данни.

Добавя се фиктивна стойност (Laplace smoothing), за да се избегнат нулеви вероятности за думи, които не са наблюдавани.

Резултатът се записва във файл в JSON формат.

* + **RecognitionService**: Използва обучен модел за предсказване на автора на нов текст. Изчислява вероятностите чрез метода на наивния Бейс. Функцията **predictAuthor** предсказва вероятния автор на даден текст, като използва вероятностен модел, базиран на логаритмична правдоподобност. Процесът включва токенизация на въведения от входа текст, изчисление на логаритмични оценки за всеки автор (с помощна функция **calculateLogLikelihood** изчислява вероятността на базата на честотата на думите в модела на съответния автор), нормализация на резултатите (вероятността за всеки автор се изчислява, като се експонентира преместената оценка и се нормализира чрез общата сума) и преобразуване на логаритмичните оценки в проценти.
* Пакет org.example.recognition.util
  + Съдържа помощни класове, включително **FileUtil**, за работа с файлове, токенизацията на текстовете и записа на резултати.
* **RecognitionApplication**: Главен клас, който изпълнява програмата чрез Spring Boot.
* Ресурси
  + В директорията **resources/data** се намират поддиректории с текстове за обучение, групирани по автори.
* Пакетът **servicetests** съдържа тестове за проверка на точността на разпознаване и коректността на методите.

Програмната реализация използва обектно-ориентиран подход, като всеки компонент има ясно определена роля. За работа с JSON се използва библиотеката Jackson, а за организация на проекта – Spring Boot

1. **Примери, илюстриращи работата на програмната система**

* Input:

“Срещу прокурора седи една траурна слезлива дама от Елзас.”

* Output:

Точност на разпознаване: 99,88%

Предположен автор: Алеко Константинов

Предположен жанр: Пътепис

* Input:

“Славянство? — глупост, братство? — празна дума.

Достойнство! — вятър, чест? — безсмислен звук!”

* Output:

Точност на разпознаване: 81,01%

Предположен автор: Иван Вазов

Предположен жанр: Поезия

* Input:

„Защото онова, което слушаха, беше един град от псувни и мръсни думи, и ругателства против имена, високо държани в уважението на столицата;“

* Output:

Точност на разпознаване: 100,00%

Предположен автор: Иван Вазов

Предположен жанр: Разказ

* Input:

„— Дружество… Не ми трябва мене дружество. Аз съм сиромах човек…

Гроздан и учителят се спогледаха и се засмяха.“

* Output:

Точност на разпознаване: 100,00%

Предположен автор: Йордан Йовков

Предположен жанр: Повест

* Input:

„Най-после уреченият ден дохожда и вие тръгваме.“

* Output:

Точност на разпознаване: 89,24%

Предположен автор: Йордан Йовков

Предположен жанр: Разказ

* Input:

„Текст за тестване на разпознаването на автора на текста.“

* Output:

Няма намерен автор с повече от 80% точност

Както може да се забележи от примерите, текстове с по-голяма дължина и/или специфични думи (като собствени имена например) са причислени към творчеството на даден автор с доста по-висок процент на точност.

1. **Литература**

* Презентациите от курса по СОЗ 2024/2025 година
* Text Classification Algorithms: A Survey - Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes and Donald Brown
* Principles of Data Mining – David J. Hand, Heikki Mannila, Padhraic Smyth,.
* <https://chitanka.info/> - в сайта е влизано неколкократно за извличане на ресурси в промеждутъка 13-16 януари
* <https://docs.oracle.com/> - в сайта е влизано неколкократно през целия период на изготвяне на проекта
* <https://medium.com/@abdallahashraf90x/tokenization-in-nlp-all-you-need-to-know-45c00cfa2df7> - От тук е информацията за токенизацията. Посещавана в промеждутъка 14-16 януари.
* <https://stackoverflow.com/> - Относно въпроси от всякакво естество. Използван е през цялото време на реализация.
* <https://github.com/stanfordnlp/CoreNLP> - По-подробно запознаване обработката на естествен език (NLP). Използван в промеждутъка 15-18 януари.