

Московский авиационный институт  
(национальный исследовательский университет)

Факультет информационных технологий и прикладной  
математики

Кафедра вычислительной математики и программирования

Курсовой проект по курсу «Дискретный анализ»

Студент: Т. Д. Голубев  
Преподаватель: Н. К. Макаров  
Группа: М8О-306Б-22  
Дата:  
Оценка:  
Подпись:

Москва, 2024

# Курсовой проект

**Задача:** Необходимо реализовать наивный байесовский классификатор, который будет обучен на первой части входных данных и классифицировать им вторую часть.

# 1 Описание

Требуется написать реализацию наивного байесовского классификатора.

Наивный байесовский классификатор (Naive Bayes classifier) — вероятностный классификатор на основе формулы Байеса со строгим (наивным) предположением о независимости признаков между собой при заданном классе, что сильно упрощает задачу классификации из-за оценки одномерных вероятностных плотностей вместо одной многомерной[1].

Формула Байеса:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

В контексте машинного обучения формула Байеса приобретает следующий вид:

$$P(y_k|X) = \frac{P(y_k)P(X|y_k)}{P(X)}$$

В конечном счёте правило классификации будет пропорционально выбору класса с максимальной апостериорной вероятностью:

$$y_k \propto \arg \max_{y_k} P(y_k) \prod_{i=1}^n P(X_i|y_k)$$

## 2 Исходный код

kmp.h	
void ToLowerCase(std::string& str)	Преобразование строки в нижний регистр.
std::vector<std::string> Split(const std::string& s, const std::string& delimiters)	Разделить строку на слова.
void Fit(const std::vector<std::pair<TWord, TCategoryName>>& data)	Обучение модели.
TCategoryName Predict(std::string& str)	Предсказать классы предложений.

```

1 using TWord = std::string;
2 using TCategoryName = uint8_t;
3 const std::string DELIM = " .,!?";
4
5 class TNaiveBayesClassifier {
6 private:
7     using TCategory = std::unordered_map<TWord, uint64_t>;
8
9     std::unordered_map<TCategoryName, TCategory> categoriesWeights;
10 public:
11     TNaiveBayesClassifier() {}
12     void Fit(const std::vector<std::pair<TWord, TCategoryName>>& data);
13     TCategoryName Predict(std::string& str);
14
15     std::unordered_map<TCategoryName, TCategory> GetCategoriesWeights() const {
16         return categoriesWeights;
17     }
18 };

```

### 3 Консоль

```
cat-mood@nuclear-box:~/programming/mai-da-labs/kp/build$ ./kp_exe <test.txt  
0  
1
```

## 4 Тест производительности

Тест производительности представляет из себя следующее: На вход подаётся 50 строк для обучения и 50 для предсказания. Строки для предсказания уже заранее определены в класс. На основе этого считаются метрики: accuracy, precision, recall, F1-мера.

```
cat-mood@nuclear-box:~/programming/mai-da-labs/kp/build$ ./kp_benchmark <test3.txt
Accuracy: 0.8
Precision: 1
Recall: 0.611111
F1: 0.758621
```

Как видно, модель имеет высокую точность, но слабую полноту. Это означает, что много ложно отрицательных срабатываний, то есть положительный класс не полный.

## 5 Выводы

Выполнив курсовой проект, я написал наивный байесовский классификатор. Поработал с такими метриками, как accuracy, precision, recall, F1-мера. Построил confusion matrix.

## Список литературы

- [1] Наивный байесовский классификатор. Основная идея, модификации и реализация с нуля на Python // Хабр URL: <https://habr.com/ru/articles/802435/> (дата обращения: 25.12.2024).