Курсовой проект 23/24 по курсу дискретного анализа

Выполнил студент группы 08-307 МАИ *Путилин Дмитрий*.

Условие

Автоматическая классификация документов

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

Формат ввода

Вам даны данные в следующем формате: <training> <test> <class_1> <data_1> <class_2> <data_2> ... <class_training> <data_training> <query_1> <query_2> ... <query_test>.

В первой строке даны два числа: количество обучающих данных и количество тестовых запросов. Обучающие данные представлены парами строк, в первой строке дано одно число 0 или 1, отвечающее за номер класса, во второй строке дан весь текст документа. Тестовые данные содержат по одному документу в строке, которые необходимо классифицировать.

Тексты могут содержать любые ascii символы.

Задача реализована как интерактивная, поэтому следующий документ для классификации будет передан, только после ответа на предыдущий. Не забывайте чистить буффер!

Формат вывода

В ответ на каждый тестовый запрос выведите единственное число 0 или 1 — предполагаемый класс документа.

Метод решения

В машинном обучении под классификацией понимают задачу определения категории, к которой принадлежит ранее не встречавшийся образец, на основании обучающего множества, для элементов которого эти категории известны. Это является примером обучения с учителем (supervised learning).

Дано:

Каждый пример x принимает значения из множества V и описывается атрибутами $\langle a_1, a_2, \ldots, a_n \rangle$.

Нужно найти наиболее вероятное значение данного атрибута, т.е.

$$v_{MAP} = argmax_{v \in V} p(x = v | a_1, a_2, \ldots, a_n)$$

По теореме Байеса

$$v_{MAP} = argmax_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} = argmax_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v)$$

Предположим условную независимость атрибутов при условии данного значения целевой функции. Иначе говоря:

$$p(a_1, a_2, \ldots, a_n | x = v) = p(a_1 | x = v) p(a_2 | x = v) \ldots p(a_n | x = v).$$

Итак, наивный байесовский классификатор выбирает v :

$$v_{NB} = argmax_{v \in V} p(x = v) \prod_{i=1}^{n} p(a_i | x = v)$$

Так как на тренировочных данных может быть очень много данных, следовательно необходимо формулу для выбора класса прологарифмировать:

$$v_{NB} = argmax_{v \in V} [\log p(x = v) + \sum_{i=1}^{n} p(a_i | x = v)]$$

Также отметим, что если на тренировочных данных у нас встретилось слово, которого раньше не было, то это занулит вероятности. Во избежание данной проблемы используется метод сглаживания Лапласа.

Тюнинг модели осуществляется за счет улучшения токенизации строк, увелечения набора тренировочных данных и применения лучшего коэффициента в сглаживании Лапласа.

Описание программы

Класс NaiveBayes представляет собой класс для наивного байесовского классификатора. В привате находятся map-ы для хранения статистики слов и классов на train выборке. В публичном доступе находятся два метода: fit и predict. В методе fit происходит обучение на тренировочных данных, в predict - соответственно происходит прогнозирование для тестовых данных.

Дневник отладки

1. Первая посылка - WA, так как был не правильно реализовано считывание тестовых данных.

2. Вторая посылка - OK

Тест производительности

Для тренировочных данных используем такой набор данных: 30 тренивочных строк: 15 - 0 класса, 15 - 1. И 20 тестовых 10 - 0 класса, и 10 - 1.

		Predicted	
		Class 0	Class 1
Actual	Class 0	4	2
	Class 1	5	9

Теперь на тестовых данных посмотрим метрику. Выберем для этого accuracy_score. Для небольшого набора $accuracy\_score = 0.65$.

$$precision = \frac{TP}{TP + FP} = 0.82$$

$$recall = \frac{TP}{TP + FN} = 0.64$$

$$F1 = \frac{2 * precision * recall}{precison + recall} = 0.72$$

Выводы

Для данной задачи был реализован наивный баесовский классификатор, который классифицирует строки к определенному классу.