Задание №2.1. Наивный Байес. Машины опорных векторов

В этом задании мы построим спам-фильтр с помощью алгоритма наивного Байеса и машины опорных векторов.

В последние годы спам стал существенной проблемой в электронных коммуникациях. Наша задача – построить классификатор, который будет отличать нормальное сообщение от спама. Чтобы немного упростить себе задачу мы будем работать с фильтрацией спама в SMS сообщениях. Данные уже были разбиты на обучающую и тестовую выборки и доступны в файлах `spam_train.tsv` и `spam_test.tsv`. Подробнее о формате файлов можно прочитать в `spam_readme.txt`.

Вопрос №1 [2 балла]

Напишите код, который преобразует SMS сообщения из текстового формата в numpy вектора признаков, которые уже можно подавать на вход модели машинного обучения. Для этого вам надо завершить код функций `get_words`, `create_dictionary` и `transform_text` в файле `spam.py`. Инструкции приведены в комментариях.

После этого скрипт `spam.py` с помощью ваших функций создаст словарь и сохранит его в файле `spam_dictionary`, а также запишет фрагмент матрицы обучающей выборки.

В документе-отчете о выполнении задания укажите размер полученного словаря.

Вопрос №2 [3 балла]

В этой подзадаче вам необходимо реализовать алгоритм наивного Байеса для фильтрации спама, используя мультиномиальную модель событий и сглаживание Лапласа. Подробнее о том и другом можно найти в теоретическом материале, посвященном наивному Байесу (выложен на сайте lms). Допишите код функций `fit_naive_bayes_model` и `predict_from_naive_bayes_model` в файле `spam.py`.

Запустите скрипт `spam.py`, чтобы он обучил вашу модель наивного Байеса и вычислил точность ее работы, сохранив результат в файл `spam_naive_bayes_predictions`.

В документе-отчете о выполнении задания укажите точность классификатора на **тестовой выборке**.

Замечание. Если вы будете программировать алгоритм наивного Байеса «в лоб», то увидите, что $p(x|y) = \prod_i p(x_i|y)$ часто равно нулю. Это связано с тем, что $p(x|y)$, являющаяся произведением большого количества чисел меньших единицы, получается очень маленьким числом. Стандартное представление вещественных чисел в компьютере не позволяет сохранять их с произвольной точностью, поэтому слишком маленькие значения обнуляются (это называется потерей значимости). Вам нужно будет найти способ обойти эту проблему (подсказка – возможно как-то поможет логарифм).

Вопрос №3 [2 балла]

Мы знаем, что есть слова, которые очень часто встречаются именно в спам сообщениях. Можно неформально оценить степень «характерности» того или иного слова для спама с помощью следующей формулы:

$$\log \frac{p(x_j = i \,|y = 1)}{p(x_j = i \,|y = 0)} = \log \left(\frac{P(token\ i \mid message\ is\ SPAM)}{P(token\ i \mid message\ is\ NOTSPAM)} \right).$$

Завершите код функции `get_top_five_naive_bayes_words`, чтобы он с помощью вышеприведенной формулы возвращал пять наиболее характерных для спама слов.

В документе-отчете о выполнении задания напишите полученные пять слов.

Вопрос №4 [1 балл]

Машины опорных векторов (Support Vector Machines, SVM) являются еще одной моделью машинного обучения. Мы уже предоставили код для SVM (используя радиальные базисные функции (РБФ) в качестве ядра) в файле `svm.py`. Вам не нужно ничего менять или дописывать в этом файле.

Одна из важных задач при обучении SVM-классификатора, параметризованного РБФ ядром (а именно гауссовским ядром), это подбор подходящего радиуса указанного ядра.

Завершите код функции `compute_best_svm_raidus` так, чтобы он находил наилучший радиус, при котором точность классификатора на валидационной выборке является максимальной.

В документе-отчете о выполнении задания укажите найденное значение радиуса.