2. Юревич Е.И. Основы робототехники. – Л.: Машиностроение. Ленинградское отделение, 1985.

3. Корсунký В.А., Машков К.Ю., Наумов В.Н. Выбор критериев и классификация мобильных робототехнических систем на колесном и гусеничном ходу: Учебное пособие. – М.: МГТУ им. Н. Э. Баумана, 2014.

4. Корягин А. В. Образовательная робототехника Lego WeDo: Сборник методических рекомендаций и практикумов. – М.: ДМК Пресс, 2016.

5. Хиросэ Ш. Бионические роботы. Змееподобные мобильные роботы и манипуляторы / Шигео Хиросэ. – М.: Институт компьютерных исследований, 2014.

6. Тывес Л.И. Механизмы робототехники. Концепция развязок в кинематике, динамике и планировании движений. – М.: СИНТЕГ, 2014.

7. Потапова Р.К. Речевое управление роботом. Лингвистика и современные автоматизированные системы. – М.: Гостехиздат, 2012.

8. Краснова С.А. Блочный синтез систем управления роботами-манипуляторами в условиях неопределенности. – М.: Мир, 2014.

9. Информационные системы виртуальной реальности в мехатронике и робототехнике. Учебное пособие / Г.В. Алферов и др. – М.: Издательство СПбГУ, 2009.

УДК 004.85
doi:10.18720/SPBPU/2/id23-500

Хаммасова Луиза Шамилевна [1],
студент магистратуры;
Нестеров Сергей Александрович [2],
доцент, канд. техн. наук, доцент

РАСПОЗНАВАНИЕ СПАМ-СООБЩЕНИЙ С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ

[1, 2] Россия, Санкт-Петербург,
Санкт-Петербургский политехнический университет Петра Великого,
[1] hammasova.lsh@edu.spbstu.ru, [2] nesterov@spbstu.ru

Аннотация. В работе решается задача классификации текстов с целью обнаружения спама. Использованы различные алгоритмы машинного обучения и подходы к предобработке входных данных: нормализация исходных данных путем разделения текста по униграммам и триграммам, векторизация в соответствии с моделями «мешка» слов и TF-IDF. В качестве программных средств для решения поставленной задачи выбраны Python и Weka. Полученные модели машинного обучения, протестированные на данных, независимых от обучающих наборов, дают высокую точность.

Ключевые слова: машинное обучение, методы классификации, спам, векторы признаков, текст на естественном языке, Python, Weka.

Luiza S. Khammasova [1],
Master's Student;
Sergey A. Nesterov [2],
Candidate of Technical Sciences, Associate Professor

RECOGNITION OF SPAM MESSAGES USING MACHINE LEARNING METHODS

[1, 2] Peter the Great St. Petersburg Polytechnic University,
St. Petersburg, Russia,
[1] hammasova.lsh@edu.spbstu.ru, [2] nesterov@spbstu.ru

Abstract. The paper describes the solution of the problem of text classification for spam detection. Various machine learning algorithms and approaches to preprocessing the input data were used: normalization of the initial data by dividing the text into unigrams and trigrams, vectorization using "bag of words" and TF-IDF models. Python and Weka were chosen as software tools for solving the problem. The machine learning models were tested on new data and provided high accuracy.

Keywords: machine learning, classification methods, spam, feature vectors, natural language text, Python, Weka.

Введение

По данным Лаборатории Касперского [1] каждый год спам занимает примерно половину объема всего мирового почтового трафика. Сначала спам рассылался напрямую на единичные адреса пользователей, и его было легко блокировать. Со временем появились сложные системы массовой рассылки и высокоскоростные интернет-каналы, которые позволили быстро и дешево осуществлять рассылку спама.

В связи с этим были разработаны различные методы распознавания спам-сообщений. Обнаружение спама в электронной почте может быть выполнено как с использованием методов машинного обучения, так и с помощью других средств [2]. В представляемой работе используется подход, основанный на использовании алгоритмов машинного обучения для классификации текстов.

1. Постановка задачи и используемые алгоритмы

Пусть $L=\{L_1,\ldots,L_m\}$ — множество писем электронной почты с метками (обычное письмо или спам), а $U=\{U_1,\ldots,U_n\}$ — неразмеченное множество писем, где U_i соответствует *i*-му письму. Предполагается, что элементы L и U отличны друг от друга. Письмо представляется в виде вектора признаков, размерность которого равна размеру словаря V, $x_i=(x_{i1},x_{i2},\ldots,x_{i|v|})$, где $x_{ij}=1$, если *i*-е письмо содержит *j*-е слово и $x_{ij}=0$ – иначе. На этих данных строится фильтр $F: U\rightarrow\{1,0\}$, который классифицирует письмо как спам или реальное письмо.

Для классификации текстов в работе использовались следующие алгоритмы:

-упрощенный алгоритм Байеса [4] — вероятностный классификатор, основанный на теории Байеса с допущением о независимости признаков;

- метод опорных векторов (англ. support vector machine, SVM) [5], который строит в *n*-мерном пространстве признаков такую гиперплоскость, чтобы она разделяла объекты выборки наиболее точно;

- метод *k* ближайших соседей (англ. k-nearest neighbors, kNN) [6], суть которого заключается в том, что находятся *k* соседей, которые наиболее близко расположены к рассматриваемому объекту с неизвестной меткой. Далее новый объект относят тому классу, который является наиболее распространённым среди *k* соседей;

- многослойный персептрон (англ. multilayer perceptron, MLP) [7] — один из вариантов нейронной сети прямого распространения. Под процессом обучения нейронной сети понимается поиск таких значений весов и порогов сети, которые минимизирую ошибку. На основе собранных исторических данных веса и пороговые значения корректируются автоматически. В процессе корректировки происходит расчет ошибки путем вычисления выходных сигналов и сравнения их с целевыми.

2. Обзор данных и предобработка текста

В работе использовался набор данных Spam Email, опубликованный на платформе Kaggle [3]. В нем содержится 86,6 % реальных сообщений (4825 строк) и 13,4 % спама (747 строк), эти два вида сообщений перемешаны между собой. Для работы алгоритмов классификации требуется произвести нормализацию и векторизацию исходных сообщений, для чего использовался язык программирования Python и программная библиотека NLTK.

Нормализация текста включает в себя следующие действия:

1) приведение всех документов к нижнему регистру;
2) удаление слов, не содержащих смысловой информации;
3) удаление знаков пунктуации;
4) разбиение документов на токены; в работе для сравнения используются униграммы (токен состоит из одного слова) и триграммы (токен состоит из трех слов);
5) лемматизация текста, иными словами, приведение слова к начальной форме, учитывая морфологию слова.

Алгоритмы машинного обучения не имеют функционала для работы с текстом на естественных языках. По этой причине текст должен быть преобразован в числовые векторы. Распространенным методом извлечения признаков из текста является формирование множества, элементами которого являются отдельные слова, встречающиеся в тексте.

В качестве входных данных модели векторизации принимают токенизированные текстовые данные, над ними могут производиться не все этапы нормализации. В случае использования «мешка» слов [9] вместо

токена обозначена его частота использования в отдельном документе. В случае TF-IDF — подсчитывается важность каждого токена в документах [9]. В работе используются следующие вариации векторизации:

1) модель «мешка слов», документы разбиты на униграммы (первый вариант);

2) модель «мешка слов», документы разбиты на триграммы (второй вариант);

3) модель TF-IDF, документы разбиты на униграммы (третий вариант);

4) модель TF-IDF, документы разбиты на триграммы (четвертый вариант).

3. Полученные результаты

Обучение проводилось на 70 % данных для всех алгоритмов классификации. Распределение происходило следующим образом: тренировочному набору соответствует первые 70 % документов, остальные 30 % документов относятся к тестовому набору.

Подбор параметров алгоритмов осуществлялся методом перебора. В итоге были использованы следующие параметры:

- в качестве классификатора для реализации алгоритма Байеса взят «MultinomialNB»;
- для алгоритма SVM использовалась линейная функция ядра;
- в алгоритме KNN для обучения было выбрано 3 «соседа»;
- в MLP было 2 скрытых слоя по 2 нейрона, функция активации скрытого слоя — «logistic», функция оптимизации весов «adam», постоянная скорость обучения.

В таблицах 1 и 2 представлены лучшие результаты для каждого метода, полученные с использованием Python с библиотекой Scikit-learn и WEKA соответственно. Для оценки моделей использовались следующие показатели (используются общепринятые англоязычные названия, т. к. перевод на русский может внести неоднозначность) [9]: accuracy, recall, precision, F1-score, specificity, time.

Таблица 1

Результаты тестирования моделей, созданных на Python

Метод и характеристика	Accuracy	Recall	Precision	F1_score	Specificity	Time
Метод Байеса, «мешок» слов, униграммы	0.967	0.973	**0.988**	0.98	**0.93**	**3.1s**
SVM, «мешок» слов, униграммы	**0.972**	**0.992**	0.976	**0.984**	0.847	17.2s
KNN, TF-IDF, униграммы	0.949	0.983	0.959	0.971	0.738	5s
MLP, «мешок» слов, униграммы	0.971	0.987	0.978	0.982	0.856	67.29s

Таблица 2

Результаты тестирования моделей, Weka

Метод и характеристика	Accuracy	Recall	Precision	F1_score	Specificity	Time
Метод Байеса, «мешок» слов, униграммы	0.964	0.97	**0.985**	0.979	**0.985**	**0.65s**
SVM, «мешок» слов, униграммы	**0.982**	**0.996**	0.984	**0.99**	0.984	1.3s
KNN, «мешок» слов, униграммы	0.951	0.987	0.947	0.967	0.947	83.27s
MLP, «мешок» слов, униграммы	0.963	0.994	0.981	0.987	**0.985**	23.46s

В таблицах 1 и 2 в столбце “time” указано суммарное время обучения и тестирования модели. Из полученных результатов следует, что модели, обученные на данных с унарной токенизацией, производят классификацию точнее, а обучение производится быстрее. Скорость работы связана с тем, что в наборах с унарной токенизацией меньше элементов, поэтому параметр времени не учитывался при сравнении моделей, построенных на данных с разным содержанием токенов.

4. Тестирование на новых данных и анализ результатов

Дополнительно было произведено тестирование моделей классификации с добавлением новых данных, не входящих в первоначальный набор текстовых сообщений. Данный набор взят с платформы Kaggle [8]. Из набора были выбраны первые 2000 строк, документы были нормализованы и преобразованы в векторы признаков аналогично основному набору. После добавления новых документов общее их количество составило 7555. После нормализации и удаления документов, полностью состоящих из стоп-слов, количество документов составило 6833.

Тестирование проводилось для тех сочетаний алгоритмов, параметров и методов предобработки, которые показали лучшие результаты на предыдущей проверке для языка Python. В таблице 3 отображены значения метрик для каждого случая.

Тестирование на новых данных показало эффективность рассмотренных моделей классификации. Метод опорных векторов показал наилучшие результаты. При этом скорость работы у него не такая высокая, как у методов Байеса или KNN.

Таблица 3

Результаты классификации на новых данных

Метод и характеристика	Accuracy	Recall	Precision	F1_score	Specificity	Time
Метод Байеса, «мешок» слов, униграммы	0.985	0.989	0.993	0.99	0.958	**4.4s**
SVM, «мешок» слов, униграммы	**0.996**	**0.999**	**0.996**	**0.998**	**0.974**	28.8s
KNN, TF-IDF, униграммы	0.994	0.998	0.995	0.996	0.968	6.4s
MLP, «мешок» слов, униграммы	0.989	0.993	0.992	0.993	0.951	84.69s

Таким образом, если требуется высокая скорость обучения, стоит выбирать один из следующих методов: полиномиальный метод Байеса или KNN. Если обучение моделей производится редко или есть потребность в очень высокой точности классификации, стоит использовать метод опорных векторов.

В исследовании Н. Сутта, З. Лю, Х. Чжан [10] было проведено сравнение методов SVM, KNN, Байеса и других методов для выявления наиболее точных классификаторов. В указанной публикации для анализа тоже использовались два набора данных: первый состоял из одного набора сообщений, второй — из двух разных. Данные были представлены в виде векторов признаков по схеме TF-IDF, использовались униграммы, биграммы и триграммы. Наибольшая точность была достигнута при обучении модели SVM с линейной функцией ядра на втором виде набора данных, с токенизацией при $n = 1$ и $n = 2$. Точность (англ. accuracy) модели SVM ≈ 0.99. KNN показал точность меньше, при этом максимальная точность достигается при использовании биграмм. Классификатор, построенный по модели Байеса, показал худший результат по всем проверенным n-граммам практически среди всех моделей, проверенных в работе.

Результаты анализа, произведенного в исследовании [10], близки к тем, что были получены в представляемой работе: высокие точности достигаются при обучении на данных, состоящих из разных наборов сообщений, в обоих случаях SVM показывает более высокую точность по сравнению с другими алгоритмами. В работе Н. Сутта, З. Лю, Х. Чжан модели, обученные на n-граммах (при $n > 1$), показывали высокие результаты, что может быть объяснено большим объемом использованного набора данных (более 100 тысяч сообщений).

Заключение

В работе были построены классификаторы для определения спама в текстовых сообщениях. В качестве предварительной обработки текстов были проведены нормализация данных и представление полученных документов в векторы признаков по схемам «мешка» слов и TF-IDF.

В работе была выявлена неэффективность обучения на данных с триграммной токенизацией. Это связано с относительно небольшим набором данных для обучения. Словосочетания, состоящие из нескольких слов, встречаются реже в предложениях, чем отдельно взятые слова.

Ко второй проблеме можно отнести скорость обучения и тестирования моделей. При увеличении объема данных время, затрачиваемое на обучение модели классификации, будет увеличиваться. В таком случае требуется использовать программные реализации на языках с высокой скоростью работы. В работе были рассмотрены реализации на двух языках: Python и Java (Weka). Скорость выполнения классификации на Java выше, чем на Python. Следовательно, если требуется высокая скорость обучения и классификации, лучше использовать Java.

Высокой точности удалось достичь при обучении модели SVM на документах, разбитых на униграммы и с векторизацией «мешка слов» в Weka и Python: значения меры F1 равны 0.984 и 0.99 соответственно, что является неплохим результатом. После добавления новых документов для тестирования, значения точностей и других метрик моделей возросли. Мера F1 модели SVM стала равна 0.998.

Список литературы

1. Отчеты по спаму и фишингу | SecureList. [Электронный ресурс]. – URL: https://securelist.ru/category/spam-and-phishing-reports/ (дата обращения: 06.11.2022).

2. Мироненко А. Н. Метод распознавания спам-сообщений на основе анализа заголовка письма // МСиМ. 2010. №1 (21). – URL: https://cyberleninka.ru/article/n/metod-raspoznavaniya-spam-soobscheniy-na-osnove-analiza-zagolovka-pisma (дата обращения: 06.11.2022).

3. Spam Email | Kaggle [Электронный ресурс]. – URL: https://www.kaggle.com/mfaisalqureshi/spam-email?select=spam.csv (дата обращения: 06.11.2022).

4. Вьюгин В. В. Математические основы теории машинного обучения и прогнозирования. – М.: МЦМНО, 2013. – 387 с.

5. Журавлев Ю.И., Рязанов В.В., Сенько О.В. Распознавание. Математические методы. Программная система. Практические применения. – М.: Изд-во ФАЗИС, 2005 – 157 с.

6. Метод ближайших соседей. machinelearning.ru. [Электронный ресурс]. – URL: http://www.machinelearning.ru/wiki/index.php?title=KNN (дата обращения: 06.11.2022).

7. Raschka S., Mirjalili V. Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2. – Mumbai: Packt Publishibg, 2019 – 848 p.

8. SMS Spam Collection | Kaggle [Электронный ресурс]. – URL: https://www.kaggle.com/datasets/assumewisely/sms-spam-collection (дата обращения: 06.11.2022).

9. Мюллер А., Гвидо С. Введение в машинное обучение с помощью Python. – СПб: Альфа-книга, 2017. – 480 с.

10. Sutta N., Liu Z., Zhang X. A study of machine learning algorithms on email spam classification // CATA 2020. EPiC Series in Computing. – Vol. 69. – Pp. 170–179. – URL: https://doi.org/10.29007/qshd.

УДК 004.852, 004.62
doi:10.18720/SPBPU/2/id23-501

Гальченко Юлия Вадимовна [1],
студент магистратуры;
Нестеров Сергей Александрович [2],
доцент, канд. техн. наук, доцент

КЛАССИФИКАЦИЯ ТЕКСТОВ ПО ТОНАЛЬНОСТИ МЕТОДАМИ МАШИННОГО ОБУЧЕНИЯ

[1, 2] Россия, Санкт-Петербург, Санкт-Петербургский политехнический университет Петра Великого,
[1] artyuhova.yuv@edu.spbstu.ru, [2] nesterov@spbstu.ru

Аннотация. В данной статье рассмотрены существующие методы классификации текстов по тональности, создана модель нейронной сети, которая успешно решает поставленную задачу классификации. Нейросеть была обучена на наборах данных, которые включают отзывы о различных услугах и местах, а также рецензии на фильмы. Полученная модель нейросети показала высокий результат в задаче классификации текстов по тональности на тестовых данных.

Ключевые слова: классификация, машинное обучение, нейронные сети, сеть LSTM, методы классификации текстов, тональность текста, интеллектуальный анализ данных.

Yuliia V. Galchenko [1],
Master’s Student;
Sergey A. Nesterov [2],
Associate Professor, PhD in Technical Sciences

SENTIMENT ANALYSIS WITH MACHINE LEARNING METHODS

[1, 2] Peter the Great St. Petersburg Polytechnic University,
St. Petersburg, Russia,
[1] artyuhova.yuv@edu.spbstu.ru, [2] nesterov@spbstu.ru

Abstract. In this article, the existing methods for sentiment analysis were considered, a neural network model that successfully solves the task of classification was created. The neural network was trained on datasets that include reviews of various services and