УДК 81.32

ПОСТРОЕНИЕ И ПРИМЕНЕНИЕ КЛАССИФИКАТОРА ТЕКСТОВЫХ ОБРАЩЕНИЙ В ТЕХНИЧЕСКУЮ ПОДДЕРЖКУ

А.А. Кириллов, В.И. Виноградов

Целью данной работы является разработка веб-сервиса для классификации текстовых заявок по отделам технической поддержки. Для достижения поставленной цели решаются такие задачи, как предобработка текстовых данных, формирование и выбор признаков, сравнительный анализ методов машинного обучения, реализация веб-приложения с использованием фреймворка «Flask».

The purpose of this work is to develop a web service for the classification of text requests by technical support departments. To achieve this goal, tasks such as preprocessing text data, generating and selecting features, a comparative analysis of machine learning methods, and the implementation of a web application using the Flask framework are solved.

КЛЮЧЕВЫЕ СЛОВА
Обработка естественного языка, машинное обучение, классификация, веб-приложение.

ДЛЯ ЦИТАТЫ
*А.А. Кириллов, В.И. Виноградов.*Построение и применение классификатора текстовых обращений в техническую поддержку// Моделирование и анализ данных. 2019. №3. С. 37-42.
A.A. Kirillov, V.I. Vinogradov. Construction and application of the classifier text requests for technical support. Modelirovaniyeianalizdannykh=Modelling and data analysis *(Russia)*.2019,no.3, pp. 37-42.

ВВЕДЕНИЕ

Обработка обращений клиентов и партнеров является неотъемлемой частью рабочего процесса в компаниях, предоставляющих IT-услуги. В крупных IT-компаниях технической поддержкой клиентов информационных автоматизированных систем могут заниматься несколько отделов с десятками или сотнями сотрудников в каждом. Отделы технической поддержки обычно специализируются на решениях определенного класса вопросов, поэтому правильное направление заявки по технической поддержке в специализирующийся на ее решении отдел специалистам является важной задачей технического обслуживания клиентов.

Зачастую сотрудникам необходимо не только умение работы с той или иной системой отслеживания задач, но и знание внутренней организации компании, чего может не быть у нового сотрудника.

Исходя из желания автоматизировать процесс постановки задач отделам технической поддержки, целью данной работы является разработка веб-сервиса способного определять отдел на основе содержащейся в заявке текстовой информации.

1. ПРОЦЕДУРА ПОСТРОЕНИЯ И ПРИМЕНЕНИЯ КЛАССИФИКАТОРА ОБРАЩЕНИЙ

Рисунок 1 показывает технологию обучения модели классификатора. Входные данных предобрабатываются для выбора признаков. Позднее классификация выполняется с использованием выбранного классификатора для получения выходных данных.

Рис.1. Обучение модели классификации.

Рисунок 2 показывает технологию применения классификатора для определения отдела технической поддержки на основе текстовой информации.

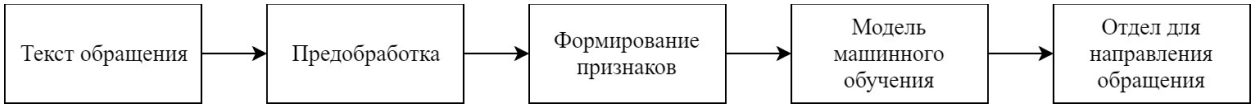

Рис. 2. Применения классификатора.

Для решения данной задачи используется язык программирования Python с дополнительными модулями. SciPy — основанная на Python экосистема с открытым исходным кодом для математических вычислений, которая так же включает в себя средства построения графиков (Matplotlib), структуры для больших данных и средства их анализа (Pandas). Scikit-learn содержит средства для анализа данных и машинного обучения. NLTK (NaturalLanguageToolkit) — это платформа для работы с данными на естественном языке. Он предоставляет наборы библиотек обработки текста для его токенизации, стемминга, разметки, синтаксического анализа.

2. ВЫБОРКА ВХОДНЫХ ДАННЫХ

Исходная выборка включала в себя тему заявки, подробное описание проблемы, наименование отдела, которому соответствует заявка. Исходная выборка состояла из 708 заявок (Рисунок 3) и была случайно разделена на обучающую и тестовую в соотношении 80:20.

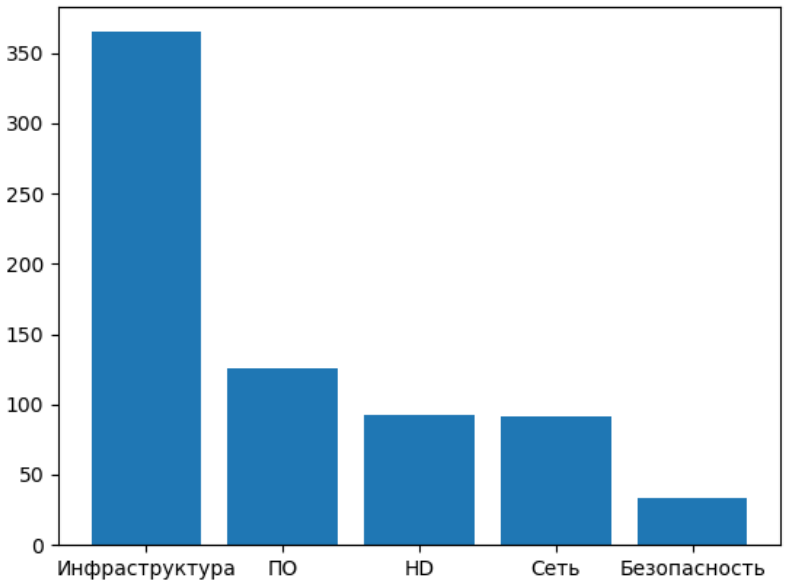

Рис. 3. Распределение заявок по отделам в исходной выборке.

Таблица 1. Исходные тексты обращений.

Тема	Описание	Отдел
eTokenчерезUSBAnywhereна Windows Server	На данный момент сервер расположен на виртуаль...	Инфраструктура
VisualStudio 2017	Для работы программистов на виртуаль...	Инфраструктура
Процесс управления PKI	За управление PKI отвечает СКБ Процесс след...	Безопасность
Выгрузка списка пользователей	Необходим список пользователей с их правами до...	Инфраструктура
Упорядочить данные по всем печатным устройствам	Проблема - в акте выполненных работ от ЮНИТ ес...	HD

3. ПРЕДОБРАБОТКА ТЕСТОВ ОБРАЩЕНИЙ

1. Удалить все нерелевантные символы (например, любые символы, не относящиеся к цифробуквенным).
2. Провести стемминг, т.е. свести различные формы одного слова к словарной форме.
3. Удалить нерелевантные слова — например, упоминания названия компании или URL-ы.
4. Перевести все символы в нижний регистр.

Таблица 2. Предобработанные тексты обращений.

Текст	Отдел
etokenusbanywherewindowsserver дан момент сервер расположвиртуальн машин эт ...	Инфраструктура
visualstudio работ программист виртуальн машин нужн ли...	Инфраструктура
процесс управленуправленотвечаскб процесс с...	Безопасность
выгрузкспискпользователнеобход список польз...	Инфраструктура
упорядоч дан всем печатн устройств проблем акт...	HD

4. ВЫБОР ПРИЗНАКОВ ДЛЯ ПОСТРОЕНИЯ КЛАССИФИКАТОРА

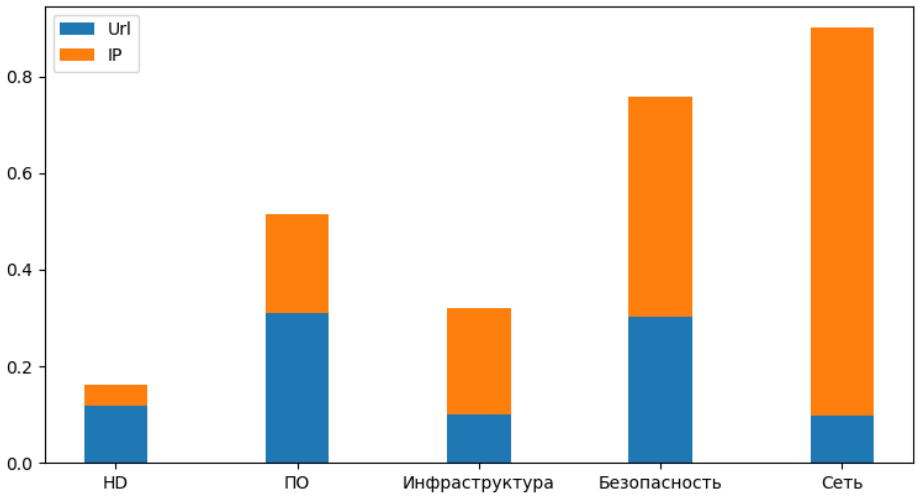

Рис. 4. Среднее количество URL-ов и IP-адресов в заявке по отделам.

Как показывает рисунок 4 среднее количество URL-ов и IP-адресов в заявке достигает наибольшего значения в отделах, связанных с безопасностью и сетью.

В результате изучения исходной выборки было решено расширить векторные представления, полученные с помощью алгоритмов выбора признаков, некоторыми признаками на основе предметной области: количество URL-ов и IP-адресов в заявке, количество Email-адресов в заявке.

BagofWords (BOW)

Самое простое представление текстовых данных в векторном виде. В ходе его работы строится словарь из всех встречающихся в документе слов. Предполагая, что значимость слова в документе тем больше, чем чаще оно встречается, метод учитывает количество вхождений слова в документ. Таким образом каждый признак показывает, как часто соответствующее слово появляется в документе.

Для того чтобы снизить влияние длинны текста на его признаки используют следующий метод выбора признаков.

Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF — статистическая мера, используемая для оценки важности слова в контексте документа, являющегося частью коллекции документов или корпуса. TF-IDF сначала вычисляется из TF (частота слова) по формуле:

$$TF_{i,j} = \frac{f_{i,j}}{\max_k f_{k,j}}, \quad (1)$$

где $f_{i,j}$ — число вхождений слова i в документе j, $max_k\, f_{k,j}$ — частота наиболее распространенного термина (слово с самой высокой частотой) в документе j (обозначается термином k).

IDF— инверсия частоты, с которой некоторое слово встречается в документах коллекции.

$$\boldsymbol{IDF_i = \log_2\left(\frac{N+1}{n_i+1}\right) + 1}, \quad (2)$$

где N — число документов в коллекции, а n_i — число документов из коллекции, в которых встречается слово i.

В этом случае добавляется 1, чтобы избежать деления на ноль. После того, как значения TF и IDF получены, TFIDF может быть получены с помощью следующей формулы:

$$TFIDF_{i,j} = TF_{i,j} \times IDF_i \quad (3)$$

5. ПОСТРОЕНИЕ КЛАССИФИКАТОРА

Признаки, полученные на предыдущем шаге, используются для алгоритмов классификации. В данной работе рассматриваются следующие алгоритмы классификации: метод опорных векторов, наивный байесовский классификатор.

Методопорныйвекторов (Support Vector Machine)

Метод опорных векторов — бинарный классификатор, но с помощью некоторых методов его можно использовать для решения мультиклассовой классификации, например, стратегия «один против одного». Стратегия объединяет каждый возможный класс друг с другом, в результате чего получается более 1 классификатора.

После создания необходимого количества классификаторов каждый классификатор будет искать разделяющую гиперплоскость. Разделяющей гиперплоскостью будет гиперплоскость, максимизирующая расстояние до двух параллельных гиперплоскостей. Алгоритм работает в предположении, что чем больше разница или расстояние между этими параллельными гиперплоскостями, тем меньше будет средняя ошибка классификатора.

Для классификаторов на базе метода опорных векторов, как правило, не требуется уменьшать размерность пространства признаков; они довольно устойчивы к переобучению и хорошо масштабируются.

Наивный байесовский классификатор (NaiveBayesClassifier)

Наивный байесовский классификатор основывается на теореме Байеса с допущением, что признаки в документе независимы.

$$\widehat{\boldsymbol{P}}(\boldsymbol{c}) = \frac{\boldsymbol{f_c}}{\boldsymbol{f_d}} \tag{4}$$

где f_c— число документов класса c в обучающей выборке, а f_d— общее число документов в обучающей выборке. Оценка вероятности принадлежности документа классу вычисляется по данной формуле:

$$\hat{P}(c|d) = P(c)\prod_{i=1}^{n} P(x_i \in d|c) \tag{5}$$

Согласно этой формуле, классом документа d будет являться класс с наибольшей оценкой вероятности. В данной работе рассматривается полиномиальный наивный байесовский классификатор (MultinomialNaiveBayesClassifier).

6. РЕЗУЛЬТАТЫ

После экспериментов с алгоритмами выбора признаков, классификаторами и их параметрами были получены следующие результаты:

Таблица 3. Оценки комбинаций моделей классификатора.

Комбинация	Precision	Recall	F1
TF-IDF + SVM	0.752	0.732	0.711
TF-IDF + NB	0.728	0.725	0.706
BOW + SVM	0.740	0.746	0.734
BOW + NB	0.717	0.718	0.698

Для сравнительного анализа используются следующие показатели $Recall = \frac{kr}{r}$ (полнота), $Precision = \frac{kr}{n}$ (точность), $F1 = \frac{2 * (Precision * Recall)}{(Precision + Recall)}$ (F-мера), где kr — количество документов, которые классификаторов правильно отметил, как относящиеся к искомой категории, $r -$ общее количество документов, относящихся к искомой категории по мнению классификатора, n — общее количество документов, относящихся к искомой категории.

7. РАЗРАБОТКА ВЕБ-СЕРВИСА

Для использования полученной модели разрабатывается веб-сервис с использованием фреймворка Flask, предназначенного для создания веб-приложений на Python.

Веб-сервис, обрабатывающий текстовую информацию о заявке и отдающий на выходе наименование отдела, является классическим клиент-серверным приложением. Клиент отправляет HTTP-запрос, в теле которого содержится JSON с темой заявки и подробным описанием заявки, на сервер, где происходит обработка информации согласно описанным в статье методам. Результат работы приходит в ответ на запрос. Благодаря такому способу взаимодействия с данным сервисом можно легком интегрироваться из других приложений.

ЗАКЛЮЧЕНИЕ

В работе описаны процессы предобработки текстовых данных, выбор признаков, построение классификатора. В результате сравнения методов машинного обучения был выбран метод опорных векторов, не требующий снижения размерности данных и в меньшей степени подверженный эффекту переобучения. Была оценена точность алгоритма на выборке из 708 заявок. Разработано прикладное решение в виде веб-сервиса, общение с которым может осуществляться на основе простых HTTP-запросов.

ЛИТЕРАТУРА

1. Aggarwal C.C., Zhai C. A Survey of Text Classification Algorithms // Mining Text Data. 2012. P. 163–222.

Работа поступила 07.02.2019г.