МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Кафедра дискретной математики и алгоритмики

СКОКЛЕЕНКО ДАНИИЛ АРТУРОВИЧ

АНАЛИЗ СООБЩЕНИЙ ПОЛЬЗОВАТЕЛЕЙ СОЦИАЛЬНЫХ СЕТЕЙ НА ПРЕДМЕТ ПРИНАДЛЕЖНОСТИ К СПАМОВЫМ

Отчёт по практике

Специальность 1-31 81 09 «Алгоритмы и системы обработки больших объёмов информации»

Руководитель практики от кафедры
Соболевская Елена Павловна
доцент, кандидат физико-математических наук

Руководитель практики от организации
Куцевол Ольга Николаевна
Ведущий инженер-программист

Минск, 2017

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Проблема спама в социальных сетях, иначе говоря, социального спама, наносит все больший ущерб для таких компаний, как Facebook, Twitter, Pinterest и т.д. Согласно исследованию компании Nexgate, специализирующейся на безопасности пользователей социальных сетей, вышеупомянутые платформы столкнулись с 355% ростом объёма социального спама в первой половине 2013 года [1].

Социальный спам влечет за собой последствия самого различного рода, и как результат, существует несколько определений понятия «социальный спам». Одна из самых популярных социальных платформ Twitter дает собственное определение спаму и предоставляет несколько различных способов сообщать о спамовых активностях. Примером такого сообщения может служить твит «@spam @username», где @username – никнейм потенциального спамера. При этом, будучи коммерческой платформой, Twitter достаточно лояльно относится к сообщениям рекламного характера, при условии, что они не нарушают правила и политику использования социальной сети. В последние годы вышло в свет огромное количество работ и исследований, посвященных выявлению различных скрытых трендов и тенденций в самых разнообразных областях, начиная от финансовых рынков и заканчивая общественно-политической сферой, огромная часть которых строилась на основе открытых данных пользователей Twitter, что подтверждает огромную значимость этой платформы, как для академической, так и для индустриальной части общества. Потребность в выявлении значимой информации в шумовом поле Twitter является одной из ключевых задач подобных исследований, и в этом смысле, огромные обьемы социального спама очень серьезно усложняют эту задачу.

Существующие техники и методы классификации социального спама в основной своей массе используют обширный обьем исторических данных о пользователе. В данной работе была предпринята попытка создания классификатора спама на основе не столь большого набора необходимых данных, а лишь той информации, которую можно извлечь непосредственно из данных об авторе твита в момент его публикации. Значимым плюсом подобного классификатора может служить его способность своевременно определять спамовых пользователей и, как следствие, потенциальное применение в режиме реального времени. Используя классифицированный вручную набор данных твитов, среди которых находились спамовые твиты, была предпринята попытка использования пяти алгоритмов классификации на основе четырех различных групп признаков.

Целью работы является исследование и разработка методов автоматического распознавания спам-сообщений в социальной сети Twitter. При написании работы были поставлены следующие задачи:

1. Исследование политики, применяемой социальной сетью Twitter относительно спама, а также техник, используемых спамерами.

2. Проведение анализа существующих методов распознавания спама, а также оцен-

ки их эффективности.

3. Разработка методов автоматического распознавания спама.

4. Реализация разработанных методов и проведение экспериментальной оценки результатов их работы.

Структура данной работы следующая. Глава 1 знакомит читателя с основной характеристикой социальной сети Twitter, а также с существующей на данный момент политикой этой компании относительно спама. В главе 2 приведен обзор и анализ ключевых исследований в сфере классификации социального спама. Глава 3 посвящена описанию техник и приемов, которые я использовал для своего классификатора. В главе 4 приведены результаты попытки построения классификатора спама и оценка его качества.

1. Политика Twitter в отношении спама

1.1. Структура Twitter

Twitter - социальная сеть, позволяющая пользователям писать сообщения, не превышающие 140 символов, которые могут быть увиденными и прокомментированными любым другим пользователем этой социальной сети. Структура Twitter представляет собой направленный граф, где пользователи — узлы, а ребра отражают отношения между ними. Некоторые определения:

Твит (сообщение) - текст длиной до 140 символов, в котором могут содержаться ссылки, хэштеги и упоминания;

Ретвит - перепубликация твита, созданного другим пользователем;

Хэштег - специально зарезервированный символ «#», употребляемый перед словом, обозначающим принадлежность к определенной теме;

Упоминание - специально зарезервированный символ «@», употребляемый перед именем пользователя;

Тренд - набор слов и хэштегов, популярность использования которых на определенный момент времени резко превышает остальные;

В Twitter пользователь может формировать связь с другими пользователями путем подписки на их твиты. Простейший пример взаимоотношений пользователей показаны на Рисунке 1.

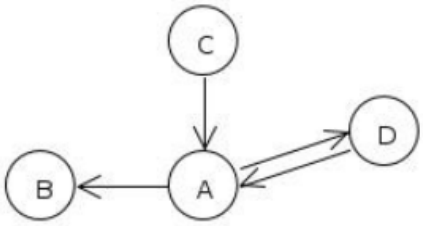

Рисунок 1 — Пример взаимоотношения пользователей Twitter

1.2. Распространенность спама в Twitter

Результаты исследований показывают, что более 3% сообщений в Twitter являются спамом [2], [3], около 8% всех ссылок указывают на вредоносный контент [4]. Также сообщается, что более 90% всех спамовых сообщений содержат ссылки [5].

В силу ограниченности длины статуса 140 символами спамеры распространяют вредоносный контент преимущественно посредством ссылок. Twitter трансформирует все ссылки с помощью собственного сервиса по сокращению URL, что позволяет спамерам не затрачивать дополнительных усилий на маскировку источника. В основной массе исследований приводятся описания следующих выявленных стратегий спамеров:

1. Спам в отношении пользователей, подписанных на спамера (любой твит пользователя автоматически появляется в новостной ленте его подписчиков) [6], [7];

2. Использование упоминаний (твит с упоминанием пользователя появляется в новостной ленте упомянутого, независимо от того, является он подписчиком спамера или нет) [7];

3. Покупка подписчиков с целью создания видимости благонадежной репутации аккаунта [6], покупка ретвитов [4];

4. Спам в отношении пользователей, чьи твиты содержат слова релевантные проводимой спам-кампании [6];

5. Использование в твитах популярных поисковых слов (техника напоминает SEO) [7];

6. Эксплуатация хэштегов популярных тем [7], [8], создание спамерских тем [4];

7. Изменение частей URL, с целью создания видимости различия ссылок [6], [9];

8. Атака фолловеров знаменитости посредством упоминания ее в своих твитах [6];

9. Использование специальных приложений, таких как TweetAttacks и TweetAdder, облегчающих использование техник 4, 8 [6];

10. Взлом аккаунтов пользователей (подбором паролей или фишингом) и рассылка спама от их лица [4], [8];

11. Копирование твитов знаменитостей с добавлением спам-ссылки [4];

12. Использование сервисов для сокращения ссылок (может быть сформирована длинная цепь перенаправлений) [4];

1.3. Политика Twitter относительно спама

Хотя алгоритм борьбы со спамом в Twitter не афишируется, чтобы не облегчить его обход, раздел «Abuse and Spam» Twitter rules [10] содержит перечисление действий, за которые аккаунт может быть заблокирован. Данный список позволяет составить некоторое представление о принципах борьбы Twitter со спамом. Некоторые условия, которые могут служить поводом для блокировки аккаунта, перечисленные в правилах Twitter:

1. Создание множественных аккаунтов;

2. Публикация ссылок на вредоносный контент;

3. Создавние статусов, содержащих преимущественно ссылки;

4. Создавние дублированных статусоа от имени одного или разных аккаунтов;
5. Создавние множественных нерелевантных статусов в темах, в т.ч. популярных;
6. Использование множественных упоминаний в целях привлечения внимания к аккаунту, сервису или ссылке;
7. Многочисленные блокировки и жалобы на спам со стороны других пользователей;

Несмотря на предупреждение о блокировке за вышеперечисленные действия, Twitter по тем или иным причинам блокирует далеко не все аккаунты, удовлетворяющие одному или нескольким из вышеперечисленных условий. Так, существует список спамерских аккаунтов [11], на странице источника которого указано, что автор регулярно отправляет списки обнаруженных спамерских аккаунтов в группу Twitter по борьбе со спамом.

Список был создан в 2009 году, тем не менее, лишь 27% из 632 аккаунтов были заблокированы Twitter к 2016.

2. Методы классификации спама

Методы обнаружения спама в социальных сетях берут начало с техник классификации сообщений электронной почты на спамерские и легитимные. Техники варьируются от надстроек над SMTP, анализа последовательностей его транзакций и составления черных списков отправителей до идентификации спам-сообщений по определенным правилам, и, наконец, применения различных методов машинного обучения: байесовской классификации, нейросетей, марковских моделей [12]. Среди традиционные методов классификации для электронной почты наиболее популярны Наивный байесовский классификатор и SVM, как признанные наилучшими для категоризации текстов [13].

Использование для обнаружение спама в Twitter теми же методами что и в электронной почте неэффективно, в первую очередь из-за ограничения на длину сообщения. Кроме того, спам в Twitter распространяется преимущественно посредством ссылок, поэтому при использовании черных списков ссылок за время до блокировки вредоносной ссылки по ней успевает перейти большое количество пользователей. Поэтому для решения задачи обнаружения спама в Twitter должны применяться методы, учитывающие специфику этой социальной сети.

В этой главе приведено описание существующих подходов классификации спама в социальной сети Twitter.

2.1. Выявление социальных спамеров

В опубликованных ранее работах, задача автоматизации поиска спама рассматривалась с 2-х точек зрения. Первая – подход, основанный на классификации конкретного пользователя, как спамера или не спамера. Формальная постановка задачи для подобного подхода выглядит следующим образом: Для аккаунта $\alpha \in A$, где A - множество аккаунтов, построить функцию $\Theta(\alpha) : A \to \{Spam, Ham\}$, то есть аккаунт классифицируется как спамерский, если функция Θ принимает значение Spam, и как благонадежный иначе.

Подобный подход наиболее популярен в большинстве работ на данный момент (см. [3], [5], [14], [9], [15], [16]) и требует наличия признаков, которые должны быть собраны на основе поведения пользователя до этого, например, прирост или убытие подписок и подписчиков, среднее количество хэштегов, ссылок и упоминаний в предыдущих сообщениях и т.д. Однако это требование не всегда достижимо из-за ограничений в Twitter API.

Ли [9] и Янг [15] использовали различные методы для сбора данных о спамовых аккаунтах (см. пункт 3.1), и проведили всестороннее исследование поведения спамеров. Они оба полагались на твиты, размещенные пользователями в прошлом, а также такие признаки, как частота публикации твитов, частота подписок, процент взаимных подписок и коэффициент локальной кластеризации сетевого графа, и боролись с тактикой уклонения спамеров, поскольку эти признаки сложны для симулирования.

Феррара [16] помимо вышеперечисленных признаков использовал сентимент-оценку текстов сообщений из все того же набора данных [9].

Миллер [17] трактует задачу обнаружения спама, как проблему обнаружения аномалий и предлагает алгоритм кластеризации для ее решения. Такой алгоритм строится на множестве легитимных пользователей, выбросы которых классифицируются как спамовые аккаунты.

Набор признаков, используемых в указанных выше работах, требует сбора исторических данных для каждого пользователя, что не отвечает требованиям обозначенного сценария обнаружения спама в реальном времени.

2.2. Выявление социального спама

Второй, альтернативный вариант, который не так популярен в литературе, это перевести задачу классификации с пользователя на отдельный твит [5]. В этом случае предполагается, что той информации, которая может быть извлечена из сообщения пользователя, достаточно для классификации сообщения как спамового. В данной работе используется именно этот подход.

Сантош [18] исследовал два различных подхода, а именно алгоритмы классификации текста на основе сжатия (т.е. динамическое марковское сжатие и предсказание путем частичного сопоставления) и использование модели «мешок слов» для обнаружения спам-твитов.

Мартинес-Ромо [8] применил расстояние Кульбака-Лейблера и исследовал различие языка, используемого в наборе твитов, содержащих трендовые слова и хэштеги, подозрительные твиты (твиты, содержащие ссылку на веб-страницу) и страницы, на которую ссылаются подозрительные твиты. Эти различия в языковой дивергенции использовались в качестве признаков для классификации. Для разметки спамовых твитов в своей работе авторы использовали несколько черных списков спамовых URL-адресов, поэтому каждый из помеченных ими спам-твитов содержит URL-ссылку и не может идентифицировать другие типы спам-твиты.

2.3. Прочие подходы

Существуют и иные методы обнаружения спама в социальных сетях, в частности, подход, основанный на тенденции спамеров образовывать связи друг с другом (см. Рис. 2), алгоритм поиска спамеров Criminal account Inference Algorithm (CIA) — на основе начального множества вредоносных аккаунтов с помощью «оценки зловредности» калькулируемой из степени связанности аккаунтов и семантической схожести их статусов отслеживаются остальные спамеры. Тем не менее, алгоритм CIA позиционируется не как полноценный алгоритм детектирования, а как «легковесный алгоритм вывода и ранжирования», который может быть встроен в систему обнаружения спамеров в комбинации с другими методами [19].

В своей работе Жань [20] предлагает находить потенциальных спамеров как поль-

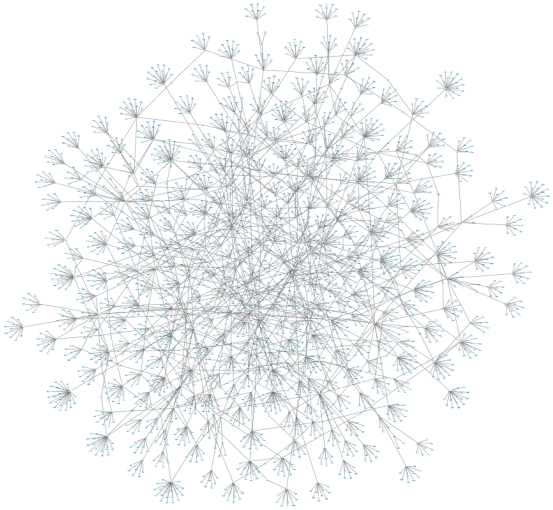

Рисунок 2 — Пример взаимосвязей спамовых аккаунтов Twitter

зователей, имеющих достаточно большое количество твитов, являющимися дубликатами сообщений других пользователей. Для выявления дублированных твитов среди набора твитов всех пользователей в наборе данных используется метод locality-sensitive hashing (LSH), позволяющий оценить коэффициент схожести Жаккара для n-грамм над словами в статусах. Твиты, для которых коэффициент Жаккара превышает 0.8, считаются дубликатами. Использование данного метода в детектировании спамеров предполагает дальнейшую классификацию на основе свойств сообщений и профилей, характерных для спамеров, имеющих дублированные сообщения.

3. Описание методологии

В этой главе будет дано описание начального набора данных, его предабработка, признаки, которые были использованы для построения классификатора, а также описание использовавшихся алгоритмов.

3.1. Набор данных

Наличие размеченной коллекции твитов в задаче классификации спама имеет критически важное значение. Набора данных, который бы полностью удовлетворял всем требованиям найдено не было. В качестве базы был использован набор данных из [9], который распространяется по открытой некоммерческой лицензии Creative Commons. Датасет представляет из себя случайно собранную в период с 30.11.2009 г. по 02.08.2010 г. информацию о 22,223 спамерах и 19,276 легитимных пользователях, а также 2,353,473 и 3,259,693 спамовых и легитимных сообщениях этих аккаунтов соответственно. Данные были получены путем регистрации 60 аккаунтов-приманок для спама, целью которых было иммитировать пользовательское поведение. Аккаунт-приманка имел возможность совершать одно из 4-х действий: (1) Пубиковать обычное текстовое сообщение (2) Упоминать один из других аккаунтов-приманок с помощью символа «@» (3) Публиковать сообщение, содержащее ссылку (4) Публиковать сообщение, содержащее одну из ТОП-10 трендовых слов Twitter Каждый из аккаунтов-приманок не мог взаимодействовать с пользователями не из круга других аккаунтов-приманок. Как только один из искуственных аккаунтов упоминался в сообщении аккаунта «извне» или появлялся у него в подписках, информация о новом спамовом аккаунте заносились в датасет.

3.2. Предобработка данных

Перед извлечением признаков из набора данных к тексту твитов были применены несколько техник предобработки с целью нормализации и уменьшения «шума» на стадии классификации. Также для извлечения семантических признаков из текста твита игнорировались ссылки, хештеги и упоминания.

3.3. Описание признаков

Поскольку спамеры и обычные пользователи имеют различные цели при размещении твитов или взаимодействии с другими пользователями в Twitter, справедливо предположение, что характеристики спам-твитов сильно отличаются от обычных твитов. Признаки, присущие твиту, включают, помимо самого твита, набор метаданных, включая информацию о пользователе, разместившем твит, который также легко доступен в потоке твитов, к которому у нас есть доступ. Анализу подвергается широкий спектр признаков, которые не требуют больших вычислительных затрат,

а также описывают лингвистические свойства, которые могут извлекаться из текста твита. Признаки были разделены по следующим группам:

1. Пользовательские признаки
2. Признаки контента
3. N-граммы
4. Сентимент-признаки

Пользовательские признаки включают в себя список из 11 атрибутов об авторе твита (см. Таблицу 1), который генерируется из метаданных каждого твита, таких как репутация пользователя [3] (см. Рис 3), который определяется как соотношение между числом подписчиков к а общему числу подписчиков и подписок и используется для измерения степени влиятельности пользователя. Такие признаки как число ретвитов не использовались, поскольку они содержат в себе историчность.

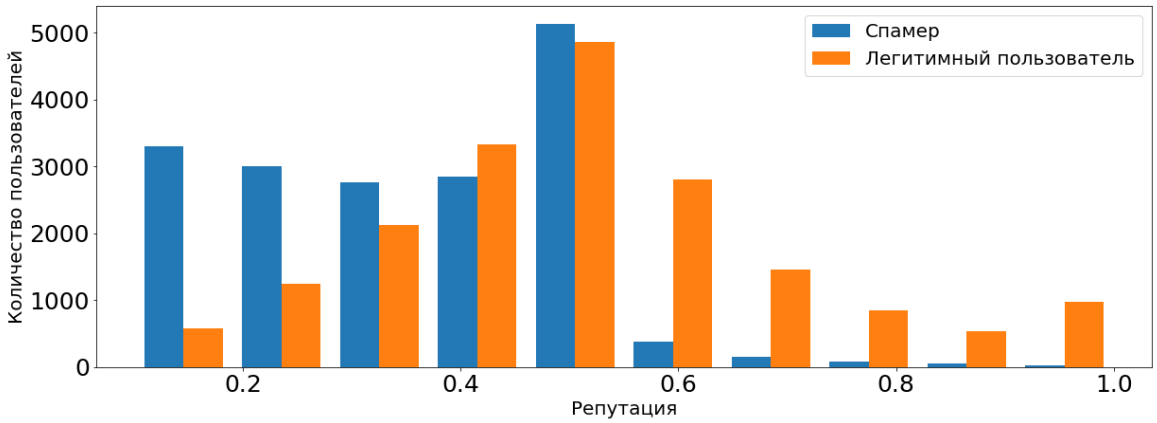

Рисунок 3 — Распределение пользователей по репутации Twitter

Признаки контента фиксируют лингвистические свойства текста каждого твита (табл. 1), включая список атрибутов контента и тегов части речи. Среди 17 контентных атрибутов присутствует количество спам-слов и количество спам-слов на одно слово, которые генерируются путем сопоставления с известными спамовыми словами [1]. Распознавание части речи (part-of-speech или POS) обеспечивает синтаксическую (или грамматическую) информацию о предложении и был использован в сообществе обработки естественного языка для оценки текстовой информативности (например, **??** использовали подсчеты POS как показатель информативности для твитов). Я использовл тэггер для работы с Твиттером [21], и признак POS состоит из униграмма и 2-skip-bi-gram представления тегов POS для каждого твита, чтобы уловить структуру и, следовательно, информативность текста.

[1] https://github.com/splorp/wordpress-comment-blacklist/blob/master/blacklist.txt

N-граммы Модели N-грамма уже давно используются в обработке естественного языка для выполнения самых различных задач, включая текстовую классификацию. Хотя он часто подвергается критике за отсутствие какого-либо явного представления долгосрочной или семантической зависимости, он удивительно эффективен для простой текстовой классификации с разумным количеством данных обучения. Для того, чтобы дать лучший результат классификации при вычислительной эффективности, был использован uni + bi-gram или bi + tri-gram с двоичными данными (т.е. 1 при присутствии признака, и 0 при отсутствии), term-frequency (tf) и tf-Idf (т.е. Частота терминов, умноженная на обратную частоту документа).

Сентимент-признаки Феррара [16] использовал сентимент-признак на уровне текста твита как часть признакового описания обьекта. Я использовал тот же список лексиконов из [22] (который отлично показал себя на Semeval-2014 Task 9 для анализа настроений в Twitter) для генерирования наших характеристик настроения, включая вручную созданные лексиконы чувств: лексикон AFINN [23] , Лексикон Bing Liu [**?**], лексикон MPQA [19]; И автоматически сгенерированные лексиконы чувств: лексика чувственного восприятия Hashtag NRC [13] и лексика Sentiment140 [13].

Пользовательские признаки	**Признаки контента**	**N-граммы**	**Сентимент-признаки**
Длина названия профиля	Количество слов (КС)	Uni + bi-gram or bi + tri-gram	Автоматически созданный сентимент-лексикон
Длина описания профиля	Количество символов		Созданный вручную сентимент-лексикон
Количество подписок (ПС)	Количество пробелов		
Количество подписчиков (ПЧ)	Количество слов с заглавной буквы (КЗ)		
Количество твитов	КЗ/КС		
Количество твитов	Максимальная длина слова		
«Возраст» аккаунта, ч. (ВА)	Средняя длина слова		
Соотношение подписчиков и подписок (ПЧ/ПС)	Количество символов «!»		
Репутация пользователь (ПЧ/(ПС + ПЧ)))	Количество символов «?»		
Прирост подписок (ПС/ВА)	Количество URL (КЛ)		
Количество твитов в день	КЛ/КС		
Количество твитов в неделю	Количество хэштегов (КХ)		
	КХ/КС		
	Количество упоминаний (КУ)		
	КУ/КС		
	Количество спамовых слов (КСП)		
	КСП / КС		
	Тэг части речи		

Таблица 1 — Список итоговых признаков

3.4. Используемые классификаторы

Наиболее популярным и эффективным подходом на данный момент является использование машинного обучения с учителем с различными признаками, основанными как на содержании сообщений, так и на свойствах отдельных профилей пользователей.

Машинное обучение (machine learning) – это область научного знания, имеющая дело с алгоритмами, «способными обучаться». Необходимость использования методов машинного обучения объясняется тем, что для многих сложных – «интеллектуальных» – задач (например, распознавание рукописного текста, речи и т. п.) очень сложно (или даже невозможно) разработать «явный» алгоритм их решения, однако часто можно научить компьютер обучиться решению этих задач. Одним из первых, кто использовал термин «машинное обучение», был изобретатель первой

самообучающейся компьютерной программы игры в шашки А. Л. Самуэль в 1959 г. [24]. Под обучением он понимал процесс, в результате которого компьютер способен показать поведение, которое в нее не было заложено «явно». Это определение не выдерживает критики, так как не понятно, что означает наречие ”явно”. Более точное определение дал намного позже Т. М. Митчелл [25]: говорят, что компьютерная программа обучается на основе опыта E по отношению к некоторому классу задач T и меры качества P, если качество решения задач из T, измеренное на основе P, улучшается с приобретением опыта E.

В настоящее время машинное обучение имеет многочисленные сферы приложения, такие, как компьютерное зрение, распознавание речи, компьютерная лингвистика и обработка естественных языков, медицинская диагностика, биоинформатика, техническая диагностика, финансовые приложения, поиск и рубрикация текстов, интеллектуальные игры, экспертные системы и др.

На этапе классификации и оценки были протестированы 5 алгоритмов классификации, реализованных с использованием scikit-learn[2]: Наивный Байесовский классификатор, Метод *k* ближайших соседей, метод опорных векторов (SVM), решающее дерево, случайные леса.

3.4.1. Наивный Байесовский классификатор

Идея Байесовского классификатора для спам-фильтра основана на предположении, что некоторые слова особенно часто встречаются в спаме. Отсюда возникает идея посчитать для каждого слова w из коллекции текстов количество писем с ним n_{ws} в спаме (spam) и количесвто писем с ним n_{wh} в «не спаме» (ham), а затем оценить вероятность появления каждого слова w в спамном и неспамном тексте:

$$P(w|spam) = n_{ws}/n_s; P(w|ham) = n_{wh}/n_s; \quad (1)$$

Получив текст письма, для которого нужно определить, относится оно к спаму или нет, мы можем оценить вероятность появления всего текста в классе «спам» и в классе «не спам» произведением вероятностей слов:

$$P(text|spam) = P(w_1|spam)P(w_2|spam)...P(w_N|spam) \quad (2)$$

$$P(text|ham) = P(w_1|ham)P(w_2|ham)...P(w_N|ham) \quad (3)$$

«Наивность» подхода в этом случае состоит в предположении, что вхождение разных слов в текст – это независимые события.

Получив текст письма, для которого нужно определить, относится оно к спаму или нет, мы можем выбрать тот класс, в котором вероятность возникновения этого текста больше:

$$a(text) = argmaxP(y|text) \quad (4)$$

[2]http://scikit-learn.org/

3.4.2. Метод *k* ближайших соседей

Метод *k* ближайших соседей (*k* nearest-neighbor, k-NN) относится к наиболее простым и в то же время универсальным методам, используемым как для решения задач классификации, так и восстановления регрессии. В случае классификации новый объект классифицируется путем отнесения его к классу, являющемуся преобладающим среди *k* ближайших (в пространстве признаков) объектов из обучающей выборки. Если *k* = 1, то новый объект относится к тому же классу, что и ближайший объект из обучающей выборки.

Аналогичный способ используется и в задаче восстановления регрессии, с той лишь разницей, что в качестве ответа для объекта выступает среднее ответов *k* ближайших к нему объектов из обучающей выборки.

Опишем метод более формально. Пусть $N_k(x)$ - множество k ближайших к x объектов из обучающей выборки. Тогда для задачи классификации положим

$$f(x) = arg\ min_y\ |\{i\colon y^{(i)} = y, x^{(i)} \in N_k(x)\}| \tag{5}$$

а для задачи восстановления регрессии –

$$f(x) = \frac{1}{k} \sum_{x^{(i)} \in N_k(x)} y^{(i)} \tag{6}$$

В некоторых случаях данные ответы учитываются с весами, обратно пропорциональными расстоянию до объекта. Это особенно полезно для решения задачи классификации с несбалансированными данными, т. е. когда число объектов, относящихся к разным классам, сильно различно.

Для определения ближайших соседей обычно используется евклидово расстояние:

$$p(x, x') = \sqrt{\sum_{j=1}^{d} |x_j - x'_j|^2} \tag{7}$$

однако оно применимо только для признаков, описываемых количественными переменными. Если все переменные качественные, то можно использовать расстояние Хэмминга:

$$p(x, x') = \sum_{j=1}^{d} I(x_j \neq x'_j) \tag{8}$$

В общем случае используют функцию:

$$p(x, x') = \sum_{j=1}^{d} a_j p_j(x_j, x'_j) \tag{9}$$

где a_j – неотрицательные параметры,$p_j(x_j, x'_j) = (x_j - x'_j)^2$ для количественных переменных, $p_j(x_j, x'_j) = I(x_j \neq x'_j)$ для качественных переменных. Заметим, что функция расстояния не обязательно должна быть метрикой и неравенство треугольника может быть не выполнено. Для повышения точности модели также могут использоваться специальные алгоритмы обучения метрики расстояния (например, Large Margin Nearest Neighbour [26]). Одним из основных параметров, влияющих на обобщающую способность алгоритма, является число «ближайших соседей» k. В целом, выбор определенного значения обусловлен характером данных задачи. Большие значения k могут привести как к более точному описанию границы, разделяющей классы, так и переобучению. Обычно для выбора k применяют различные эвристики, в частности, метод перекрестного контроля.

3.4.3. Метод опорных векторов (SVM)

Один из самых популярных методов машинного обучения – машина опорных векторов (SVM – Support Vector Machine) – является развитием идей, предложенных в 1960–1970 гг. В. Н. Вапником и А. Я. Червоненкисом. Окончательное очертание метод принял в 1995 г., когда было показано, как в этом методе можно эффективно использовать ядра [27]. Данный метод изначально относится к бинарным классификаторам, хотя существуют способы заставить его работать и для задач мультиклассификации. Идею метода удобно проиллюстрировать на следующем простом примере: даны точки на плоскости, разбитые на два класса (Рисунок 4).

Проведем линию, разделяющую эти два класса. Далее, все новые точки (не из обучающей выборки) автоматически классифицируются следующим образом: точка выше прямой попадает в класс A, точка ниже прямой — в класс B.

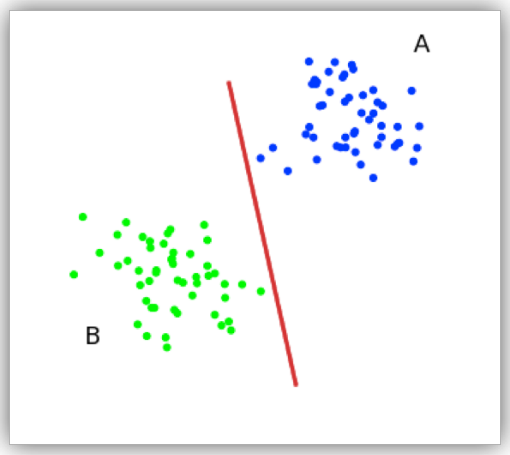

Рисунок 4 — Демонстрация метода опорных векторов

Такую прямую назовем разделяющей прямой. Однако в пространствах высоких размерностей прямая уже не будет разделять наши классы, так как понятие «ниже прямой» или «выше прямой» теряет всякий смысл. Поэтому вместо прямых необходимо рассматривать гиперплоскости — пространства, размерность которых на еди-

ницу меньше, чем размерность исходного пространства. В $\mathbb{R}^3$, например, гиперплоскость — это обычная двумерная плоскость. В нашем примере существует несколько прямых, разделяющих два класса (Рисунок 5):

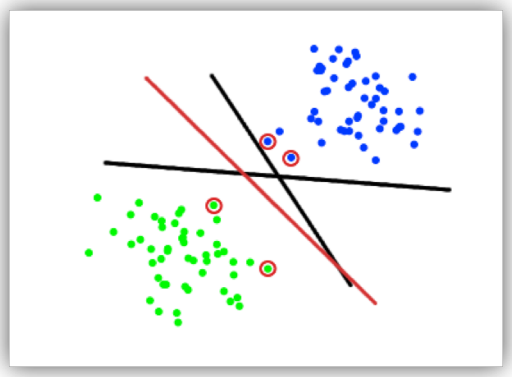

Рисунок 5 — Демонстрация метода опорных векторов

С точки зрения точности классификации лучше всего выбрать прямую, расстояние от которой до каждого класса максимально. Другими словами, выберем ту прямую, которая разделяет классы наилучшим образом (красная прямая на рис. 5). Такая прямая, а в общем случае — гиперплоскость, называется оптимальной разделяющей гиперплоскостью. Другими словами, оптимальной разделяющей гиперплоскостью называется гиперплоскость, ортогональная отрезку, соединяющему ближайшие точки выпуклых оболочек двух классов, и проходящая через середину этого отрезка.

Вектора, лежащие ближе всех к разделяющей гиперплоскости, называются *опорными векторами* (support vectors). На Рисунке 5 они помечены красными кружочками.

Для дальнейшей формализации положим в математической постановке задачи обучения с учителем,$\mathbb{Y} = \{-1, 1\}, \mathbb{X} = \mathbb{R}^n$. Рассмотрим случай линейной разделимости. Пусть имеется обучающая выборка: $(x_1, y_1), \ldots, (x_m, y_m), x_i \in \mathbb{R}^n, y_i \in \{-1, 1\}$. Метод опорных векторов строит классифицирующую функцию в виде

$$f(x) = sign(w \times x + b) \tag{10}$$

где $w \times x$ — скалярное произведение, w — нормальный вектор к разделяющей гиперплоскости,b — вспомогательный параметр. Те объекты, для которых $f(x) = 1$ попадают в один класс, а объекты с $f(x) = -1$ — в другой. Выбор именно такой функции неслучаен: любая гиперплоскость может быть задана в виде $w \times x + b = 0$ для некоторых w и b.

Далее, мы хотим выбрать такие w и b которые максимизируют расстояние до каждого класса. Можно подсчитать, что данное расстояние равно $\frac{1}{||w||}$ (Рисунок 6) Проблема нахождения максимума $\frac{1}{||w||}$ эквивалентна проблеме нахождения миниму-

ма $||w||^2$. Запишем все это в виде задачи оптимизации:

$$\begin{cases} arg\ min_{(w,b)}\ ||w||^2, \\ y_i(w \times x + b) \geqslant 1, i = 1, \ldots, m \end{cases} \tag{11}$$

которая является стандартной задачей квадратичного программирования и решается с помощью множителей Лагранжа.

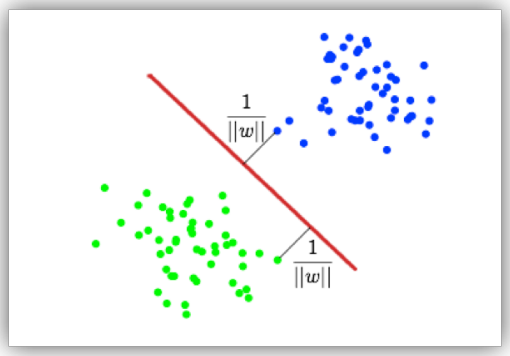

Рисунок 6 — Демонстрация метода опорных векторов

На практике случаи, когда данные можно разделить гиперплоскостью, или, как еще говорят, *линейно*, довольно редки. Пример линейной неразделимости можно видеть на Рисунок 7

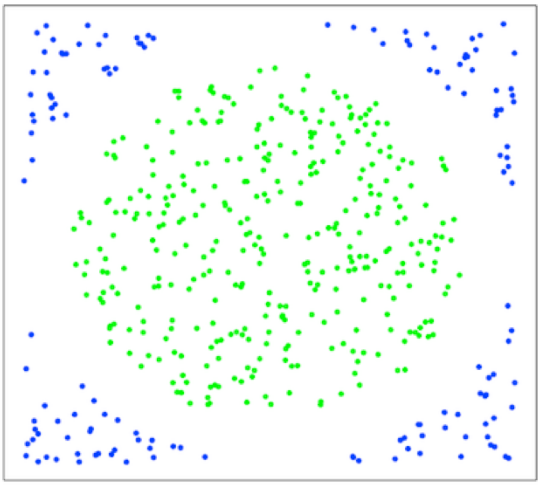

Рисунок 7 — Демонстрация метода опорных векторов

В этом случае поступают так: все элементы обучающей выборки вкладываются в пространство $\mathbb{Z}$ более высокой размерности с помощью специального отображения $\varphi : \mathbb{R}^n \to Z$. При этом отображение выбирается так, чтобы в новом пространстве Z выборка была *линейно* разделима. Классифицирующая функция f принимает вид

$$f(x) = sign(w \times \varphi(x) + b) \tag{12}$$

Выражение $K(x, x') = \varphi(x) \times \varphi(x')$ называется *ядром* классификатора. С математической точки зрения ядром может служить любая положительно определенная симметричная функция двух переменных. Положительная определенность необходимо

для того, чтобы соответствующая функция Лагранжа в задаче оптимизации была ограничена снизу, т.е. задача оптимизации была бы корректно определена. Точность классификатора зависит, в частности, от выбора ядра. Чаще всего на практике встречаются следующие ядра:

1. Полиномиальное: $K(x, x') = (x \times x' + const)^d$
2. Радиальная базисная функция: $K(x, x') = e^{-\gamma|x-x'|^2}, \gamma > 0$
3. Гауссова радиальная базисная функция: $K(x, x') = e^{-\frac{|x-x'|^2}{(2\sigma^2)}}$
4. Сигмоид:$K(x, x') = \text{tanh } (k(x \times x') + c), k > 0, c < 0$

Для того чтобы использовать метод опорных векторов для задачи классификации с числом классом $N > 2$, возможно также использовать следующие стратегии:

1. "Каждый против каждого": построить $N(N-1)/2$ классификаторов на всех возможных подзадачах бинарной классификации. Новый объект классифицируется всеми построенными решающими правилами, затем выбирается преобладающий класс.
2. "Один против всех": обучить N моделей на задачах бинарной классификации вида "один класс против всех остальных". Класс нового объекта выбирается по максимальному значению отступа.

3.4.4. Метод решающего дерева

Метод деревьев решений (decision trees) является одним из наиболее популярных методов решения задач классификации и прогнозирования. Иногда этот метод Data Mining также называют деревьями решающих правил, деревьями классификации и регрессии.

Как видно из последнего названия, при помощи данного метода решаются задачи классификации и прогнозирования. Основная идея деревьев решений состоит в рекурсивном разбиении пространства признаков с помощью *разбиений* (*splits*) гиперплоскостей, параллельных координатным гиперплоскостям (если признак – количественный), либо по категориям (если признак номинальный). В каждом из полученных в конце процедуры "ящиков"$R_1, R_2, \ldots, R_M$ функция аппроксимируется константой.

Впервые деревья решений были предложены Ховилендом и Хантом (Hoveland, Hunt) в конце 50-х годов прошлого века. Самая ранняя и известная работа Ханта и др., в которой излагается суть деревьев решений - "Эксперименты в индукции"("Experiments in Induction") – была опубликована в 1966 году.

В наиболее простом виде дерево решений – это способ представления правил в иерархической, последовательной структуре. Основа такой структуры – ответы

”Да”или ”Нет”на ряд вопросов. В этом случае решается задача бинарной классификации, т.е. создается дихотомическая классификационная модель, или, так называемые, бинарные деревья.

В узлах бинарных деревьев ветвление может вестись только в двух направлениях, т.е. существует возможность только двух ответов на поставленный вопрос (”да”и ”нет”). Бинарные деревья являются самым простым, частным случаем деревьев решений. В остальных случаях, ответов и, соответственно, ветвей дерева, выходящих из его внутреннего узла, может быть больше двух.

Как правило, внутренние узлы дерева являются атрибутами некой базы данных (или признаками). Эти атрибуты называют прогнозирующими, или атрибутами расщепления (splitting attribute). Конечные узлы дерева, или листы, именуются метками класса, являющимися значениями зависимой категориальной переменной.

Каждая ветвь дерева, идущая от внутреннего узла, отмечена предикатом расщепления. Последний может относиться лишь к одному атрибуту расщепления данного узла. Характерная особенность предикатов расщепления: каждая запись использует уникальный путь от корня дерева только к одному узлу-решению. Объединенная информация об атрибутах расщепления и предикатах расщепления в узле называется критерием расщепления (splitting criterion).

Для любой задачи может быть построено множество деревьев решений различного качества, с различной прогнозирующей точностью.

Качество построенного дерева решения весьма зависит от правильного выбора критерия расщепления. Над разработкой и усовершенствованием критериев работают многие исследователи.

Преимущества деревьев решений:

1. Возможность производить обучение на исходных данных без их дополнительной предобработки (нормализация и т. п.);
2. Не требуют от пользователя выбора входных атрибутов (независимых переменных)
3. Нечувствительность к монотонным преобразованиям данных;
4. Устойчивость к выбросам;
5. Возможность обрабатывать данные с пропущенными значения;
6. Поддержка работы с входными переменными разных (смешанных) типов (и с числовыми, и с категориальными типами данных);
7. Позволяют создавать классификационные модели в тех областях, где аналитику достаточно сложно формализовать знания;
8. Возможность интерпретации построенного дерева решений (его интуитивность);

9. Точность моделей, созданных при помощи деревьев решений, сопоставима с другими методами построения классификационных моделей (статистические методы, нейронные сети);

10. Разработан ряд масштабируемых алгоритмов, которые могут быть использованы для построения деревьев решения на сверхбольших базах данных;

11. Быстрый процесс обучения (на построение классификационных моделей при помощи алгоритмов конструирования деревьев решений требуется значительно меньше времени, чем, например, на обучение нейронных сетей).

Многие статистические методы являются параметрическими, и пользователь должен заранее владеть определенной информацией, например, знать вид модели, иметь гипотезу о виде зависимости между переменными, предполагать, какой вид распределения имеют данные. Деревья решений, в отличие от таких методов, строят непараметрические модели. Таким образом, деревья решений способны решать такие задачи Data Mining, в которых отсутствует априорная информация о виде зависимости между исследуемыми данными.

Процесс конструирования дерева решений

Напомним, что рассматриваемая нами задача классификации относится к стратегии обучения с учителем, иногда называемого индуктивным обучением. В этих случаях все объекты тренировочного набора данных заранее отнесены к одному из предопределенных классов.

Алгоритмы конструирования деревьев решений состоят из этапов ”построение”или ”создание ”дерева (tree building) и ”сокращение ”дерева (tree pruning). В ходе создания дерева решаются вопросы выбора критерия расщепления и остановки обучения (если это предусмотрено алгоритмом). В ходе этапа сокращения дерева решается вопрос отсечения некоторых его ветвей.

Рассмотрим эти вопросы подробней.

Критерий расщепления

Процесс создания дерева происходит сверху вниз, т.е. является нисходящим. В ходе процесса алгоритм должен найти такой критерий расщепления, иногда также называемый критерием разбиения, чтобы разбить множество на подмножества, которые бы ассоциировались с данным узлом проверки. Каждый узел проверки должен быть помечен определенным атрибутом. Существует правило выбора атрибута: он должен разбивать исходное множество данных таким образом, чтобы объекты подмножеств, получаемых в результате этого разбиения, являлись представителями одного класса или же были максимально приближены к такому разбиению. Последняя фраза означает, что количество объектов из других классов, так называемых ”примесей в каждом классе должно стремиться к минимуму.

Существуют различные критерии расщепления. Наиболее известные - мера энтропии и индекс Gini.

В некоторых методах для выбора атрибута расщепления используется так называемая мера информативности подпространств атрибутов, которая основывается на энтропийном подходе и известна под названием ”мера информационного выигрыша”(information gain measure) или мера энтропии.

Другой критерий расщепления, предложенный Брейманом (Breiman) и др., реализован в алгоритме CART и называется индексом Gini. При помощи этого индекса атрибут выбирается на основании расстояний между распределениями классов. Если дано множество T, включающее примеры из n классов, индекс Gini, т.е. gini(T), определяется по формуле 13:

$$gini(T) = 1 - \sum_{j=1}^{n} p_j \tag{13}$$

где T - текущий узел, p_J - вероятность класса j в узле p, n - количество классов.

Чем больше частных случаев описано в дереве решений, тем меньшее количество объектов попадает в каждый частный случай. Такие деревья называют ”ветвистыми”или ”кустистыми они состоят из неоправданно большого числа узлов и ветвей, исходное множество разбивается на большое число подмножеств, состоящих из очень малого числа объектов. В результате ”переполнения”таких деревьев их способность к обобщению уменьшается, и построенные модели не могут давать верные ответы.

В процессе построения дерева, чтобы его размеры не стали чрезмерно большими, используют специальные процедуры, которые позволяют создавать оптимальные деревья, так называемые деревья ”подходящих размеров”(Breiman,1984).

Какой размер дерева может считаться оптимальным? Дерево должно быть достаточно сложным, чтобы учитывать информацию из исследуемого набора данных, но одновременно оно должно быть достаточно простым. Другими словами, дерево должно использовать информацию, улучшающую качество модели, и игнорировать ту информацию, которая ее не улучшает.

Тут существует две возможные стратегии. Первая состоит в наращивании дерева до определенного размера в соответствии с параметрами, заданными пользователем. Определение этих параметров может основываться на опыте и интуиции аналитика, а также на некоторых ”диагностических сообщениях”системы, конструирующей дерево решений.

Вторая стратегия состоит в использовании набора процедур, определяющих ”подходящий размер”дерева, они разработаны Бриманом, Куилендом и др. в 1984 году. Однако, как отмечают авторы, нельзя сказать, что эти процедуры доступны начинающему пользователю.

Процедуры, которые используют для предотвращения создания чрезмерно больших деревьев, включают: сокращение дерева путем отсечения ветвей ; использование правил остановки обучения.

Следует отметить, что не все алгоритмы при конструировании дерева работают

по одной схеме. Некоторые алгоритмы включают два отдельных последовательных этапа: построение дерева и его сокращение; другие чередуют эти этапы в процессе своей работы для предотвращения наращивания внутренних узлов.

Рассмотрим правило остановки. Оно должно определить, является ли рассматриваемый узел внутренним узлом, при этом он будет разбиваться дальше, или же он является конечным узлом, т.е. узлом решением.

Остановка - такой момент в процессе построения дерева, когда следует прекратить дальнейшие ветвления.

Один из вариантов правил остановки - ”ранняя остановка”(prepruning), она определяет целесообразность разбиения узла. Преимущество использования такого варианта - уменьшение времени на обучение модели. Однако здесь возникает риск снижения точности классификации. Поэтому рекомендуется ”вместо остановки использовать отсечение”(Breiman, 1984).

Второй вариант остановки обучения - ограничение глубины дерева. В этом случае построение заканчивается, если достигнута заданная глубина.

Еще один вариант остановки - задание минимального количества примеров, которые будут содержаться в конечных узлах дерева. При этом варианте ветвления продолжаются до того момента, пока все конечные узлы дерева не будут чистыми или будут содержать не более чем заданное число объектов.

Существует еще ряд правил, но следует отметить, что ни одно из них не имеет большой практической ценности, а некоторые применимы лишь в отдельных случаях.

Сокращение дерева или отсечение ветвей

Решением проблемы слишком ветвистого дерева является его сокращение путем отсечения (pruning) некоторых ветвей.

Качество классификационной модели, построенной при помощи дерева решений, характеризуется двумя основными признаками: точностью распознавания и ошибкой.

Точность распознавания рассчитывается как отношение объектов, правильно классифицированных в процессе обучения, к общему количеству объектов набора данных, которые принимали участие в обучении.

Ошибка рассчитывается как отношение объектов, неправильно классифицированных в процессе обучения, к общему количеству объектов набора данных, которые принимали участие в обучении.

Отсечение ветвей или замену некоторых ветвей поддеревом следует проводить там, где эта процедура не приводит к возрастанию ошибки. Процесс проходит снизу вверх, т.е. является восходящим.

Это более популярная процедура, чем использование правил остановки. Деревья, получаемые после отсечения некоторых ветвей, называют усеченными.

Если такое усеченное дерево все еще не является интуитивным и сложно для понимания, используют извлечение правил, которые объединяют в наборы для опи-

сания классов. Каждый путь от корня дерева до его вершины или листа дает одно правило. Условиями правила являются проверки на внутренних узлах дерева.

3.4.5. Метод случайных лесов

Один из общих подходов в машинном обучении заключается в использовании композиции ”слабых”решающих правил. Итоговое правило строится путем взвешенного голосования ансамбля базовых правил. Для построения базовых правил и вычисления весов в последнее время часто используются две идеи:

1. Баггинг (bagging – bootstrap aggregation): обучение базовых правил происходит на различных случайных подвыборках данных или/и на различных случайных частях признакового описания; при этом базовые правила строятся независимо друг от друга.

2. Бустинг (boosting): каждое следующее базовое правило строится с использованием информации об ошибках предыдущих правил, а именно, веса объектов обучающей выборки подстраиваются таким образом, чтобы новое правило точнее работало на тех объектах, на которых предыдущие правила чаще ошибались.

Эксперименты показывают, что, как правило, бустинг работает на больших обучающих выборках, тогда как баггинг – на малых. Одной из реализаций идеи баггинга является случайный лес [6]. Случайный лес, а точнее – случайные леса (random forests), является одним из наиболее универсальных и эффективных алгоритмов обучения с учителем, применимым как для задач классификации, так и для задач восстановления регрессии. Идея метода [6] заключается в использовании ансамбля из деревьев решений (например,), которые обучаются независимо друг от друга. Итоговое решающее правило заключается в голосовании всех деревьев, входящих в состав ансамбля. Для построения каждого дерева решений используется следующий алгоритм: Пусть обучающая выборка состоит из примеров, размерность пространства признаков равна , и задан параметр (в задачах классификации обычно). Все деревья комитета строятся независимо друг от друга по следующей процедуре:

1. Сгенерируем случайную подвыборку с повторением размером из обучающей выборки. (Таким образом, некоторые примеры попадут в неё несколько раз, а в среднем , т.е. примерно примеров не войдут в неё вообще).

2. Построим решающее дерево, классифицирующее примеры данной подвыборки, причём в ходе создания очередного узла дерева будем выбирать признак, на основе которого производится разбиение, не из всех признаков, а лишь из случайно выбранных. Выбор наилучшего из этих признаков может осуществляться различными способами. В оригинальном коде Бреймана используется критерий Джини (28), применяющийся также в алгоритме построения решающих деревьев CART. В некоторых реализациях алгоритма вместо него используется критерий прироста информации.

3. Дерево строится до полного исчерпания подвыборки и не подвергается процедуре pruning (отсечения ветвей) (в отличие от решающих деревьев, построенных по таким алгоритмам, как CART или C4.5).

Классификация объектов проводится путём голосования: каждое дерево комитета относит классифицируемый объект к одному из классов, и побеждает класс, за который проголосовало наибольшее число деревьев.

Оптимальное число деревьев подбирается таким образом, чтобы минимизировать ошибку классификатора на тестовой выборке. В случае её отсутствия, минимизируется оценка ошибки out-of-bag: доля примеров обучающей выборки, неправильно классифицируемых комитетом, если не учитывать голоса деревьев на примерах, входящих в их собственную обучающую подвыборку.

Одной из модификаций метода случайных деревьев является алгоритм крайне случайных деревьев (extremely random forests), в котором на каждом этапе для выбора признака, по которому будет проводиться разбиение, используется вновь сгенерированная случайная бутстрэп-выборка.

Среди достоинств алгоритма случайных деревьев можно выделить:

1. высокое качество предсказания;
2. способность эффективно обрабатывать данные с большим числом классов и признаков;
3. внутреннюю оценку обобщающей способности модели;
4. легко построить параллельную высоко масштабируемую версию алгоритма;
5. метод обладает всеми преимуществами деревьев решений, в том числе о отсутствием необходимости предобработки входных данных, обработкой как вещественных, так и категориальных признаков, о поддержкой работы с отсутствующими значениями. К недостаткам можно отнести:
6. склонность к переобучению на некоторых задачах, особенно на зашумленных задачах;
7. большой размер получающихся моделей. Требуется памяти для хранения модели, где — число деревьев.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Nguyen, H. Research report: 2013 state of social media spam. — 2013. http://nexgate.com/wp-content/uploads/2013/09/Nexgate-2013-State-of-Social-Media-Spam-Research-Report.pdf.

2. R, Kelly. Twitter study. — 2009. www.pearanalytics.com/wp-content/uploads/2012/12/Twitter-Study-August-2009.pdf.

3. Wang, A. H. Don't follow me - spam detection in twitter. / A. H. Wang. — 2010. — P. 142–151.

4. @spam: the underground on 140 characters or less / Grier C., Kurt T., Paxson V., Zhang M. // 17th ACM conference on Computer and communications security. — 2010. — P. 27–37.

5. Detecting spammers on twitter / F. Benevenuto, G. Magno, T. Rodrigues, V. Almeida. // CEAS. — 2010.

6. S., Vasumathi. M. twitter games: how successful spammers pick targets / Vasumathi S., Shankar V., Gupta // Proceedings of the 28th Annual Computer Security Applications Conference. — 2012. — P. 389–398.

7. Suspended accounts in retrospect: an analysis of twitter spam / Kurt T., Grier C., Song D., Paxson V. // Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference. — 2011. — P. 243–258.

8. J., Martinez-Romo. Detecting malicious tweets in trending topics using a statistical analysis of language / Martinez-Romo J., Lourdes A. // Expert Systems with Applications. — 2013. — P. 2992–300.

9. Lee, K. Seven months with the devils: A long-term study of content polluters on twitter / K. Lee, B. D. Eoff, J. Caverlee. — 2011.

10. Правила twitter. https://support.twitter.com/articles/18311-the-twitter-rules.

11. Twitter spammers list. http://www.infochimps.com/datasets/twitter-spammers-list.

12. A survey of emerging approaches to spam filtering // ACM Computing Surveys (CSUR). — 2012. — P. 1–27.

13. A., Almeida T. Advances in spam filtering techniques / Almeida T. A., Yamakami A. // Computational Intelligence for Privacy and Security. — 2012.

14. McCord, M. Spam detection on twitter using traditional classifiers. / M. McCord, M. Chuah. — 2011. — Vol. 6906 of Lecture Notes in Computer Science. — P. 175–186.

15. Yang, C. Die free or live hard? empirical evaluation and new design for fighting evolving twitter spammers / C. Yang, R. C. Harkreader, G. Gu. // RAID'11. — 2011. — P. 318–337.

16. The rise of social bots / E. Ferrara, O. Varol, C. Davis et al. // CoRR. — 2014. — Vol. abs/1407.5225.

17. Twitter spammer detection using data stream clustering / Z. Miller, B. Dickinson, W. Deitrick et al. — 2014.

18. amd I. Minambres-Marcos, I. Santos. Twitter content-based spam filtering. in international joint conference soco'13-cisis'13-iceute'13 / I. Santos amd I. Minambres-Marcos, C. Laorden, P. G. Bringas. — 2014. — P. 449–458.

19. Analyzing spammers' social networks for fun and profit: a case study of cyber criminal ecosystem on twitter / Chao Y., Harkreader R., Zhang J. et al. // Proceedings of the 21st international conference on World Wide Web. — 2012.

20. Duplicate detection for identifying social spam in microblogs / Zhang Q., Haixin M., Weining Q., Aoying Z. // IEEE International Congress on Big Data. — 2013.

21. Part-of-speech tagging for twitter: Annotation, features, and experiments. / K. Gimpel, N. Schneider, B. O'Connor et al. — 2011. — P. 42–47.

22. Mohammad, S. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets / S. Mohammad, S. Kiritchenko, X. Zhu. // In Proceedings of the seventh international workshop on Semantic Evaluation Exercises (SemEval-2013), Atlanta, Georgia, USA, June 2013.

23. Nielsen, F.A. A new anew: Evaluation of a word list for sentiment analysis in microblogs. arxiv preprint arxiv:1103.2903, 2011. / F.A. Nielsen.

24. Some studies in machine learning using the game of checkers // ibm journal. — 1959. — vol. 3, № 3. — P. 210–229.

25. T., Mitchell. Machine learning /t. mitchell. — new york : Mcgraw-hill science/engineering/math / Mitchell T. — P. 432.

26. Weinberger, K. Q. Distance metric learning for large margin nearest neighbor classification / k. q. weinberger, l. k. saul. // journal of machine learning researc. — 2009. — vol. 10. / K. Q. Weinberger. — P. 207 – 244.

27. Cortes, C. Support-vector networks / c. cortes, v. n. vapnik // machine learning. — 1995. — T. 20, 3. — c. / C. Cortes. — P. 273 – 297.