УПРАВЛЕНИЕ ОБРАЗОВАНИЯ ГОРОДА ПЕНЗЫ
Муниципальное бюджетное общеобразовательное учреждение
«Гимназия № 53» г. Пензы
(МБОУ «Гимназия № 53» г. Пензы)

XXVI научно-практическая конференция школьников города Пензы «Я исследую мир»

МЕТОДЫ DATA-MINING ДЛЯ РАСПОЗНАВАНИЯ СПАМА

Выполнил:
Абрамова Дарья Александровна,
учащаяся 10 класса
Научный руководитель:
Артюхина Елена Владимировна,
старший преподаватель кафедры
«Компьютерные технологии» ПГУ

Пенза, 2021

Введение

Развитие методов записи и хранения данных привело к бурному росту объемов собираемой и анализируемой информации. Объемы данных настолько внушительны, что человеку просто не по силам проанализировать их самостоятельно, хотя необходимость проведения такого анализа вполне очевидна, ведь в этих данных заключены знания, которые могут быть использованы при принятии решений.

Для того чтобы провести автоматический анализ данных, используются методы Data Mining [1]. Data Mining – это процесс обнаружения в «сырых» данных ранее неизвестных нетривиальных практически полезных и доступных интерпретации знаний, необходимых для принятия решений в различных сферах человеческой деятельности.

Актуальность:

В настоящее время остро стоит вопрос фильтрации нежелательной рассылки, так как по различным данным доля спама доходит до 90%[2], что это сказывается на эффективности работы систем электронной почты. Поэтому администратор любой почтовой системы желает получить наиболее эффективный и дешевый способ защиты от спама. Методы Data Mining возможно применить для решения поставленной проблемы.

Объектом исследования являются методы Data Minig.

Предмет исследования возможность применения методов Data Mining для распознавания спама.

Целью работы является создание интеллектуального спам-фильтра с применением методов Data Mining.

Задачи:

1. Изучить основные методы Data Mining.
2. Познакомиться с возможностями платформы Deductor, в частности с инструментарием для работы с логистической регрессией, деревьями решений, нейронными сетями.
3. Исследовать возможность применения методов для создания интеллектуального спам-фильтра.
4. Провести экспериментальные исследования для выбора лучшего метода для решения поставленной проблемы.

Ожидаемые результаты и практическая значимость

Разработанный интеллектуальный спам-фильтр позволит уменьшить вероятность пропуска спам-писем, отправляемых пользователям посредством сети интернет.

1. Обоснование необходимости создания интеллектуального спам-фильтра

Спам (англ. spam) — массовая рассылка корреспонденции рекламного характера лицам, не выражавшим желания её получить. Приведем наиболее распространенные виды спама:

Реклама. Неоспоримо, что реклама сейчас является двигателем любого бизнеса. Очень часто надоедливая реклама отмечается пользователями как спам, впрочем, по сути, она им и является. Массовые рассылки имеют низкую стоимость и охватывает большое количество адресатов, но эффективность такой рекламы достаточно мала, так как такой вид рекламы вызывает некую настороженность у потенциальных клиентов.

Реклама незаконной продукции.

Антиреклама.

Нигерийские письма. Мошенники используют спам рассылку для выманивания денег. Письмо, содержит ложную информацию о неожиданном наследстве от далекого родственника. Но для получения это наследства необходимо перевести некую несущественную сумму на оформление документов или другие операции.

«***Фишинг***» — попытка спамеров заполучить конфиденциальную информацию о его кредитных карточках логины и пароли. Такое письмо обычно маскируется под официальное сообщение от администрации банка, хотя таковым не является.

2020-2021 года с пандемией и массовым переходом на дистанционную работу и общение онлайн во многом оказался необычным, что нашло отражение в спаме. Злоумышленники использовали тематику COVID-19, приглашали жертв на несуществующие видеоконференции, требовали зарегистрироваться в «новых корпоративных сервисах». Учитывая, что борьба с пандемией еще не завершена, можно предполагать, что основные тренды этих годов в ближайшем будущем останутся актуальны. Вероятно, усилится атаки на корпоративный сектор, тем более что формат удаленной работы, делает сотрудников уязвимее. Не стоит терять бдительность и пользователям мессенджеров, так как количество спама, приходящего на мобильные устройства, скорее всего, тоже будет расти. [2].Таким образом, проблема спама и разработка эффективного спам-фильтра, способного подстраиваться под изменяющиеся параметры среды, является актуальной.

2. Методы Data Mining

Data Mining - это процесс обнаружения в сырых данных ранее неизвестных, нетривиальных, практически полезных и доступных интерпретации знаний, необходимых для принятия решений в различных сферах человеческой деятельности.

Существуют четыре основные ***сферы применения технологии Data Mining***:

- для решения бизнес-задач (банки, фондовый рынок, страхование, производство, телекоммуникации, электронная коммерция, маркетинг и другие);
- для решения задач государственного уровня (поиск лиц, уклоняющихся от налогов; борьба с терроризмом);
- для научных исследований в медицине, молекулярной генетике, биоинформатике, прикладной химии и других областях;
- для решения Web-задач (поисковые машины, счетчики, борьба со спамом).

Наиболее мощные механизмы оценки спама – это самообучающиеся алгоритмы, обладающие способностью к адаптации. В качестве подобных адаптивных механизмов рассмотрим 3 алгоритмов: логистическая регрессия; деревья решений; нейронные сети.

Чем более простой алгоритм, тем он грубее, но при этом легче объяснить полученные результаты, схематичное изображение представлено на рисунке 1. Наиболее мощные алгоритмы способны находить сложные нелинейные зависимости, но их интерпретация является непростой задачей. На практике необходимо находить компромисс между точностью и простотой.

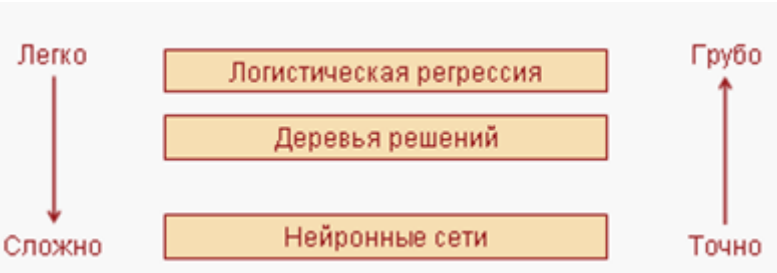

Рисунок 1 – Data Mining алгоритмы

Для реализации методов Data Mining предлагается использовать платформу Deductor, которая разработана компанией BaseGroup Labs [1]. Выбор инструментария обуславливается удобством программного обеспечения, его доступностью и правилами лицензирования. Deductor 5 является аналитической платформой, т.е. основой для создания законченных прикладных решений. Реализованные в Deductor технологии позволяют на базе единой архитектуры пройти все этапы построения аналитической системы: от создания хранилища данных до автоматического подбора моделей и визуализации полученных результатов.

3. Исходные данные

Для использования алгоритмов фильтрации для определения спам-писем, необходимо определить ключевые характеристики писем, по которым производится фильтрация.

Такими характеристиками могут являться:

– повторяющиеся слова или словосочетания;

– общие количества символов, знаков пунктуации, цифр, пробельных символов, слов;

– частота каждой буквы, специальных символов;

– средние длины слова, предложения;

– распределение частоты длин слов;

– мера разнообразия;

– количество уникальных слов.

Так как содержание писем является конфиденциальной информацией, и получить доступ к ним мы не имеем возможности. То для проведения исследования исходные данные, возьмем из Репозитория UCI (UCI Machine Learning Repository) [6]. UCI — это крупнейший репозиторий реальных и модельных задач машинного обучения, созданный в университете г. Ирвин (Калифорния, США). Содержит реальные данные по прикладным задачам в области биологии, медицины, физики, техники, социологии, и др.

Обучающая выборка содержит 4601 записей со следующими 58 атрибутами.

48 переменных word_freq_WORD = процентному отношению слов WORD в письме, т. е. 100*(число появлений слова WORD в данном письме к общему числу слов).

6 переменных char_freq_CHAR= процентному отношению появления символов CHAR в письме, к общему количеству символов в письме.

1 переменная capital_run_length_average = средняя длина непрерывной последовательности заглавных букв.

1 переменная capital_run_length_longest= наибольшая длина непрерывной последовательности заглавных букв

1 переменная типа capital_run_length_total= суммарное число заглавных букв в письме.

1 результативная переменная, говорящая является ли письмо спамом (1) или нет (0).

Приведем некоторые из них, полный список переменных приведем в приложении 1.

Таблица 1

№	Исходное название	Перевод
1	word_freq_make	Частота слова “сделать”
2	word_freq_address	Частота слова “адрес”
3	word_freq_all:	Частота слова “все”
8	word_freq_internet	Частота слова “интернет”
17	word_freq_business	Частота слова “бизнес”
20	word_freq_credit	Частота слова “кредит”
42	word_freq_meeting	Частота слова “встреча”
43	word_freq_original	Частота слова “оригинал”
44	word_freq_project	Частота слова “проект”
48	word_freq_conference	Частота слова “конференция”
49	char_freq ;	Частота символа ;
50	char_freq (	Частота символа (
53	char_freq_$	Частота символа $
54	char_freq_#	Частота символа #
55	capital_run_length_average	самая длинная последовательность прописных букв
56	capital_run_length_longest	средняя длина всех последовательностей прописных букв
57	capital_run_length_total	общая длина всех последовательностей прописных букв
58	spam	определение спам или нет {0, 1}

Распределение зависимой переменной следующее: 1813 случаев спама, 2788 –обычное письмо. На сайте приведены результаты предыдущих решений, что позволит провести сравнение полученных нами результатов. Ошибка неправильной классификации ~7%. Ложные срабатывания (пометка хорошей почты как спама) очень нежелательны. В случае, если требовали нулевом количестве ложных срабатываний в наборе обучения/тестирования, то 20-25% спама прошло через фильтр. Данная задача решалась с применением MAP-сплайнов и с помощью растущих деревьев решений [8], где получены результаты 7% и 5% соответственно.

Импортируем файл с данными в Deductor. При применении методов все множество данных будем разбивать на 2 множества: обучающее (80%) и тестовое (20%). Данные из обучающего множества используются для настройки модели (обучения параметров модели). Данные тестового множества используют для оценки качества построенной модели, когда на модель подаются аналогичные, но новые для нее данные, и результаты на этом множестве должны быть сравнимы с результатами на обучающем множестве.

4. Логистическая регрессия

Логистическая регрессия – это разновидность множественной регрессии, общее назначение которой состоит в анализе связи между несколькими независимыми переменными и зависимой переменной. Задача линейной регрессии состоит в нахождении коэффициентов уравнения

$$y = b_0 + b_1 x_1 + b_2 x_2 + ... + b_n x_n,$$

где y – выходная (зависимая) переменная модели;

$x_1, x_2 ... x_n$ – входные (независимые) переменные.

$b_0, b_1 ... b_n$ – коэффициенты регрессии.

Бинарная логистическая регрессия, как следует из названия, применяется в случае, когда зависимая переменная является бинарной, т.е. может принимать только два значения (1 – «спам»., 0 – «обычное письмо»). Иными словами, с помощью логистической регрессии можно оценивать вероятность того, что событие наступит для конкретного случая.

Построим логистическую регрессию. Добавим в сценарий обработчик *Логистическая регрессия*. На первом шаге мастера зададим входные и выходные значения, на втором шаге производится разбиение множества на обучающее и тестовое (случайным образом). На следующем шаге производится отбор переменных в регрессионную модель, Выберем *Полное включение*, т. е. в модель будут включены все переменные. Параметры обучения оставим установленными по умолчанию. На последнем шаге выберем способы отображения результатов (коэффициенты регрессии, качество классификации, таблица сопряженности, что-если, таблица)

Были рассчитаны коэффициенты логистической регрессии, по причине их большого количества опустим этим результаты. Оговоримся, что получено значение коэффициента детерминации R^2=0.699, что свидетельствует о том, что 69,9% вариации результативной переменной зависит от вариации включенных в модель факторов. На рисунке 2 представим результат работы визуализатора *Качество классификации*.

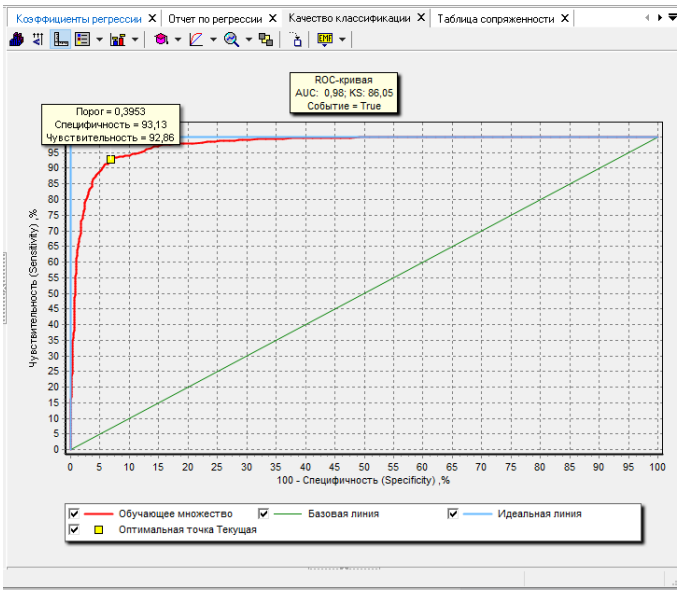

Рисунок 2 – ROC-кривая

Мы видим график ROC-кривой, на которой отмечается порог отсечения, чувствительность и специфичность. Тип порога отсечения можно задать кнопкой 

Оценим качество логистической регрессии на основе таблицы сопряженности, представленной на рисунке 3.

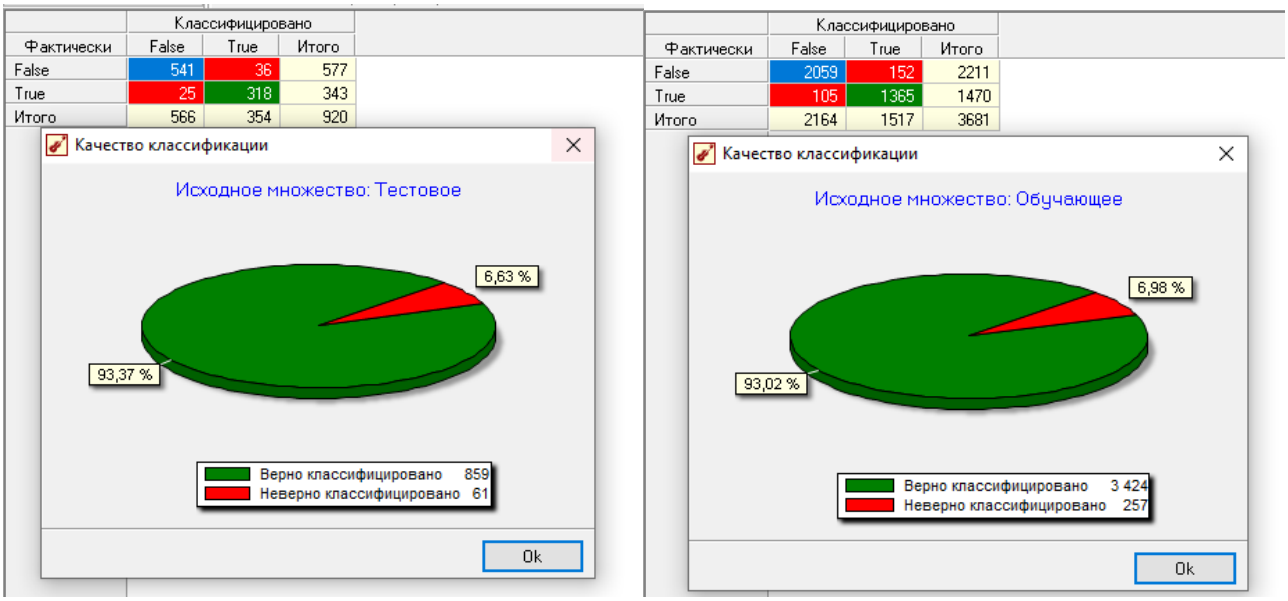

Рисунок 3 – Результаты логистической регрессии на тестовом и обучающем множестве

В таблице сопряженности мы можем увидеть ошибки классификации, представим два результата на тестовом множестве и на обучающем множестве. В таблице сопряженности та тестовом множестве зафиксировано 36 случаев ложного срабатывания спам-фильтра. Письмо было признано спамом, хотя оно таковым не являлось. И 25 случаев ложного пропуска, когда спам определился как обычное письмо. Доля верно классифицированных случаев составила 93,37%, соответственно ошибка 6,63%, что согласуется с результатами, достигнутыми на обучающем множестве.

В зависимости от преследуемой цели исследователь может настраивать специфичность (доля истинноположительных наблюдений) и чувствительность (1 - доля ложноположительных наблюдений) логистической регрессии. Если мы зададимся целью уменьшить долю пропускаемого фильтром спама, то мы можем уменьшит порог отсечения, но при этом мы увеличиваем долю ложных срабатываний фильтра, и вероятно важные для пользователя письма будут отмечены как спам. На рисунке 4 представлены результаты классификации при различных порогах отсечения.

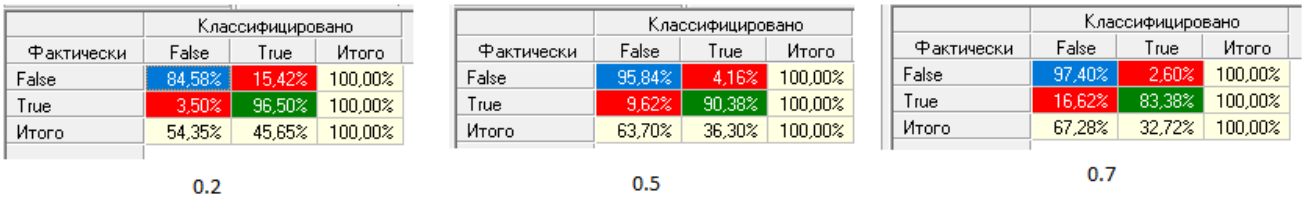

Рисунок 4 – Результаты логистической регрессии при различных значениях порога отсечения 0.2, 0.5, 0.7

Увеличение порога отсечения приведет к уменьшению доли ложных срабатываний (ложная тревога или ошибка II рода), но при этом увеличится количество ложноотрицательных результатов (ошибка I рода). Модель, которая способна идеально точно классифицировать как положительные, так и отрицательные примеры, будет иметь 100-процентную чувствительность и специфичность. Чтобы минимизировать ошибки I рода, нужно использовать модель с высокой чувствительностью. Чтобы минимизировать ошибки II рода, нужно использовать модель с высокой специфичностью.

5. Деревья решений

Деревья решений являются одним из наиболее популярных подходов к решению задач добычи данных. Они создают иерархическую структуру классифицирующих правил типа «ЕСЛИ…ТО…», имеющую вид дерева. Каждый узел дерева осуществляет проверку одной независимой переменной (вероятно, также сравнение их друг с другом, или вычисление некоторой функции одной или ряда переменных). Конечные узлы дерева (листья) соответствуют классам (градациям зависимой переменной). В результате построения дерева решений формируются правила, по которым можно определить принадлежность объекта к заранее заданным классам, если значения его независимых переменных будут удовлетворять условиям. Записанным в узлах дерева на пути от корня к листу.

В Мастер обработки выбираем Дерево решений. Настройку полей оставим, как и в прошлом методе. В окнах Разбиение исходного множества на подмножества, Настройка параметров обучения дерева решений и Способ построения оставляем настройки по умолчанию. Далее в следующем окне запускаем процесс построения дерева решений

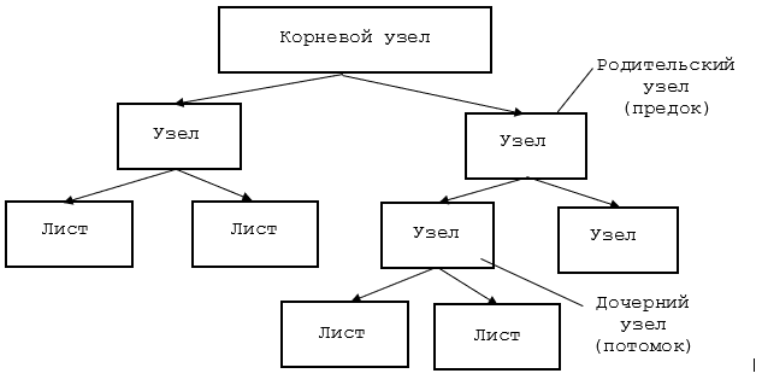

Рисунок 5 Структура дерева решений

Пример структуры дерева решений показана на рисунке 5. При построении дерева решений формируются решающие правила и для каждого из них создается узел. Для каждого узла необходимо выбрать атрибут ветвления. Существуют различные методы построения дерева решений, чаще всего применяются так называемые жадные алгоритмы. При выборе атрибута ветвления должны обеспечиваться максимальная чистота. Большая чистота означает, что в узле будут представлены в основном объекты, относящиеся к одному классу. В идеальном случае в узле должны быть представлены объекты одного класса (не должно быть "примесей"). Кроме того, разбиение не должно создавать узлы, содержащие мало объектов (это значит, что правило применимо для малого числа объектов).

Добавим в сценарий обработчик *Дерево решений*. На первом шаге мастера зададим входные и выходные значения. Аналогично предыдущему методу выбираем параметры разбиения на обучающее и тестовое множества. На 4 шаге выберем параметры обучения дерева решений: минимальное количество примеров в узле, при котором будет создаваться новый узел, примем равным 36. Это примерно 1% от объема обучающей выборки. Большее значение может привести к полному отсутствию правил, меньшее к построению недостоверных правил.

Визуализатор *Дерево решений* строит графическое представление дерева решений

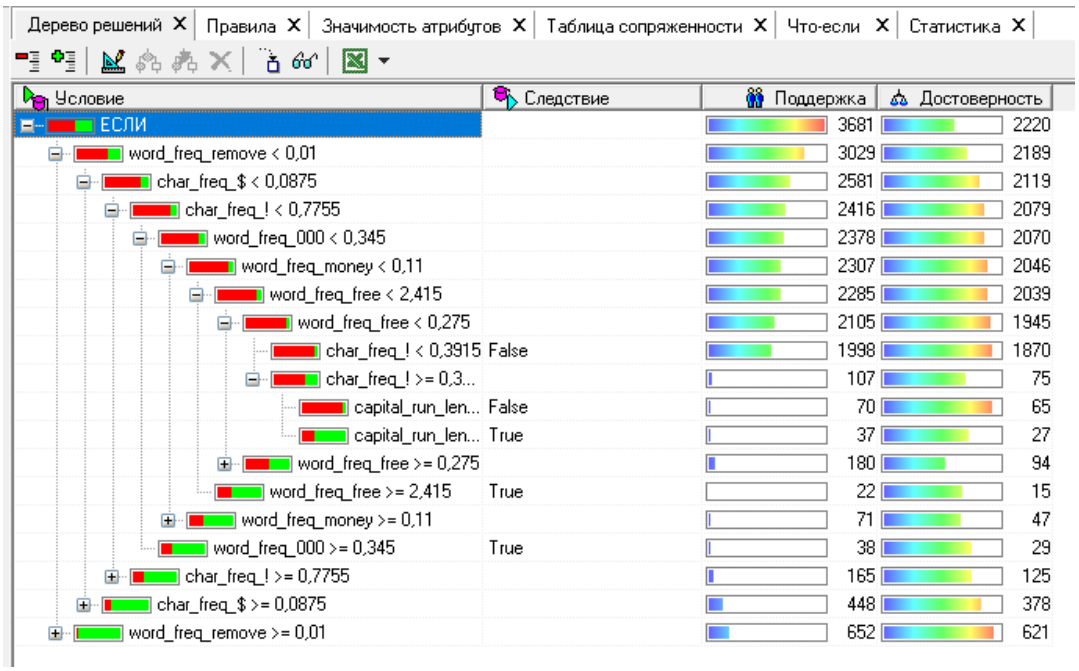

Рисунок 6 Визуализатор *Дерево решений*

Визуальное представление дерева решений сходно с принятым в операционной системе Windows отображением структуры папок: узлы отображаются как папки, слева от которых находится значок "+", если эта ветвь имеет свернутые подветви, и "-", если ветвь полностью развернута.

Кроме того, для каждого узла решений указываются: следствие, поддержка и достоверность:

- *Следствие* — метка класса к которому отнесено данное правило.
- *Поддержка* — количество примеров, попавших в узел, от общего количества примеров выборки. Чем выше это значение, тем выше статистическая обоснованность результатов, поскольку классификация в данном узле проводится на большем количестве примеров.
- *Достоверность* — число распознанных примеров от общего числа примеров в данном узле. Чем выше данный показатель, тем достовернее результаты квалификации.

Визуализатор *Правила* показывает правила по каждой ветви дерева решений. На визуализаторе *Значимость атрибутов* мы можем увидеть, на сколько сильно выходное поле зависит от каждого из входных значений.

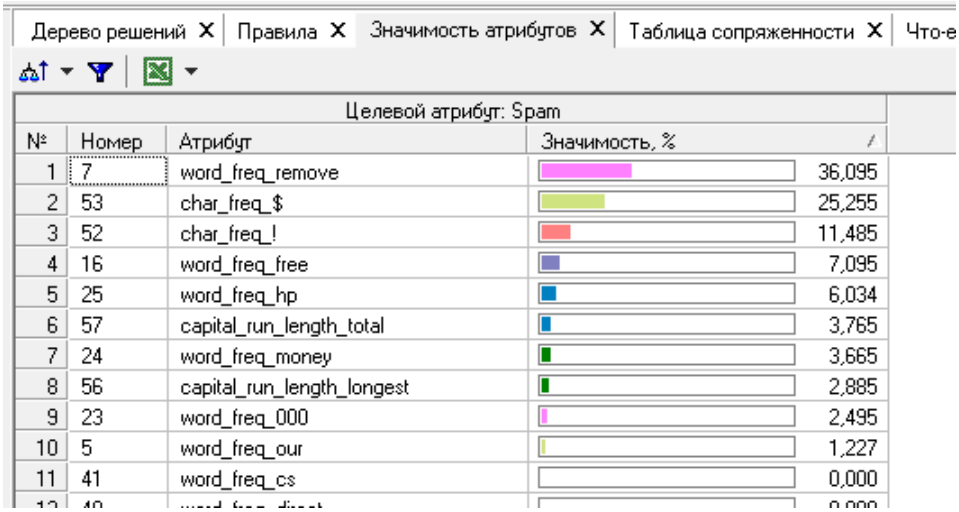

Рисунок 7 Визуализатор *Значимость атрибутов*

На рисунке 8 изображены таблицы сопряженности на обучающем и тестовом множествах, по ней мы можем сказать, что ошибка классификации на обучающем множестве составила 7,63%, на тестовом 9,02%.

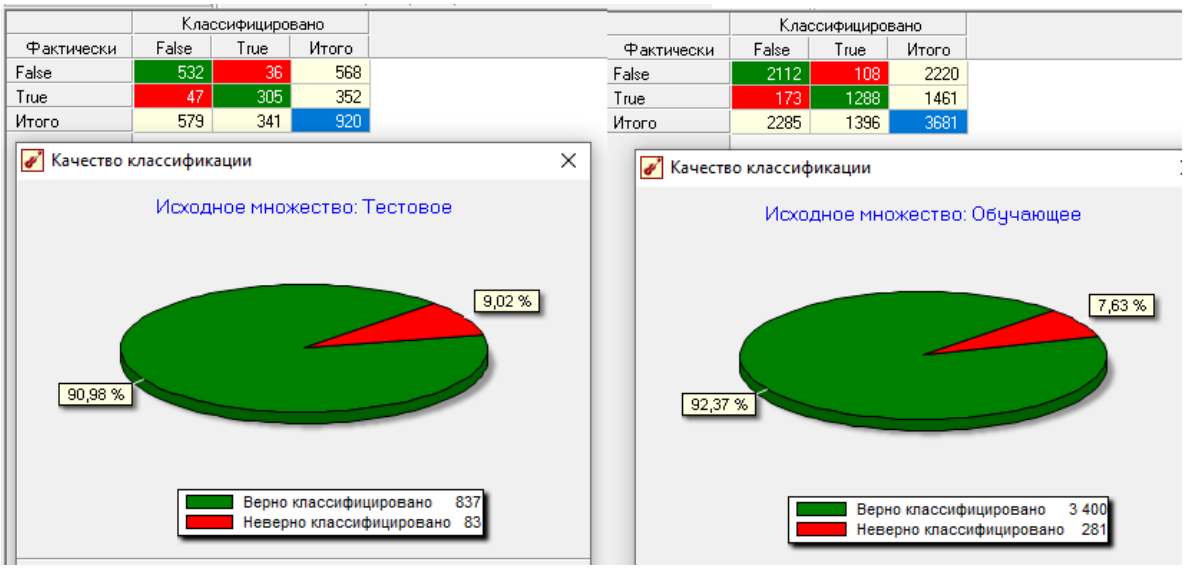

Рисунок 8 Таблица сопряженности для дерева решений

Деревья решений позволяют хранить информацию о данных в компактной форме. Вместо громоздких массивов данных можно хранить дерево решений, которое содержит точное описание объектов. Деревья решений отлично справляются с задачами классификации, т.е. отнесения объектов к одному из заранее известных классов.

6. Нейронная сеть

Деревья решений и регрессия, широко используемые для решения задач классификации и прогнозирования, имеют ряд ограничений. Эти алгоритмы не всегда позволяют разделить объекты на классы или построить регрессию с приемлемой ошибкой. От этих недостатков свободны нейронные сети, которые могут моделировать любые зависимости. Нейронные сети в Data Mining чаще всего используются для классификации и прогнозирования.

Нейронные сети – самообучающиеся системы, имитирующие деятельность человеческого мозга, попытка смоделировать низкоуровневую структуру мозга. Мозг человека состоит из белого и серого вещества: белое – это тела нейронов, а серое – соединяющие их нервные волокна. Каждый нейрон состоит из трех частей: тела клетки, дендритов и аксона (рис. 9).

Нейрон получает информацию через свои *дендриты*, а передает ее дальше через *аксон*, разветвляющийся на конце на тысячи *синапсов* — нервных нитей, соединяющих нейроны между собой. Простейший нейрон может иметь до 10 000 дендритов, принимающих сигналы от других клеток. В человеческом мозге содержится приблизительно 10^{11} нейронов. Каждый нейрон связан с $10^3 \ldots 10^4$ другими нейронами. Каждый нейрон может существовать в двух состояниях – возбужденном и невозбужденном. В возбужденное состояние нейрон переходит под воздействием электрохимических сигналов, поступающих к нему от других нейронов, когда эти воздействия становятся достаточно большими. В возбужденном состоянии нейрон сам посылает электрический сигнал другим соединенным с ним нейронам. Интенсивность сигнала, получаемого нейроном (а, следовательно, и возможность его активации), сильно зависит от

активности синапсов. Важно отметить, что веса синапсов могут изменяться со временем, а значит, меняется и поведение соответствующего нейрона.

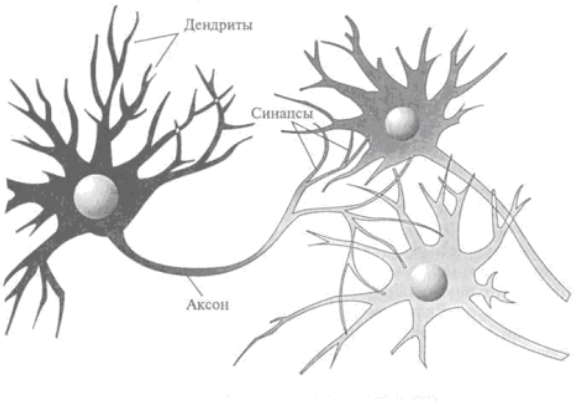

Рисунок 9 Строение биологического нейрона

Таким образом, простейшая математическая модель нейрона должна включать суммирование входных сигналов, умноженных на синаптические веса, и обработку полученной суммы пороговым элементом. Первая модель нейрона была предложена в 1943 году (рис. 10).

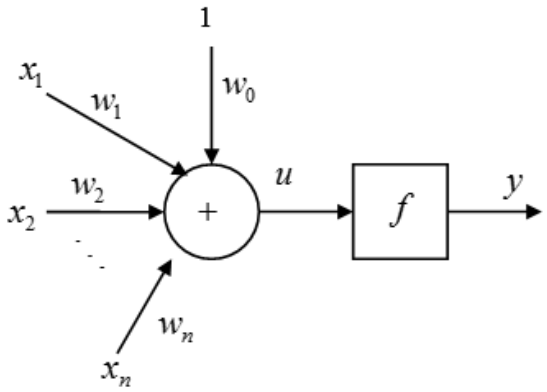

Рисунок. 10. Модель нейрона Мак-Каллока-Питса

Нейрон описывается зависимостью

$$y = f\left(\sum_{i=1}^{n} w_i x_i + w_0\right),$$

где y – выход нейрона, x_i – входные сигналы, w_i – синаптические веса, w_0 – порог (смещение), $f(u)$ – пороговая функция активации:

$$f(u) = \begin{cases} 1, & \text{если } u > 0, \\ 0, & \text{если } u \le 0. \end{cases}$$

Многослойный персептрон — многослойная сеть, состоящая из нейронов, расположенных на разных уровнях, причем, помимо входного и выходного слоев, имеется еще, как минимум один внутренний, т.е. скрытый слой. Изображенная на рис. 11 сеть содержит M слоев, среди которых выделяют первый (входной) слой, промежуточные (скрытые) слои и выходной слой.

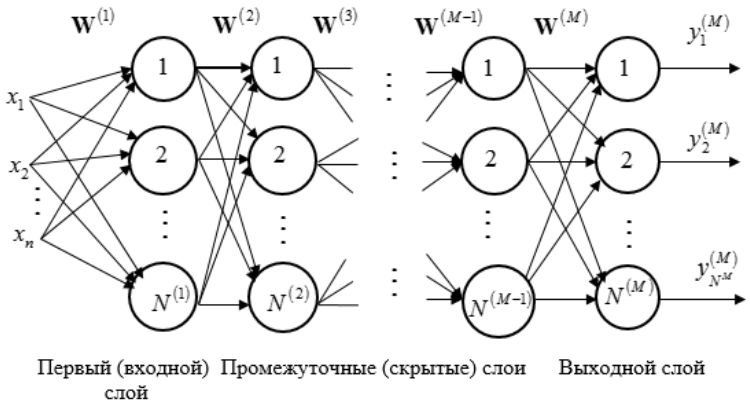

Рисунок. 11. Структура многослойной сети

Для того чтобы нейронная сеть приобрела способность решать конкретную задачу необходимо ее обучить. У нас имеется некоторый набор данных. Предъявляя примеры обучающей выборки на вход сети, мы получаем от нее некоторый ответ, не обязательно верный. Нам известен и верный (желаемый) ответ – в данном случае нам хотелось бы, чтобы на выходе нейронной сети при предъявлении примера обучающей выборки был получен ответ: 1 или 0 (спам или обычное письмо). Вычисляя разность между желаемым ответом и реальным ответом сети, мы получаем вектор *ошибки*. Один и тот же пример мы можем предъявлять сети много раз. Такой процесс называют «обучение с учителем» (см. рис. 12).

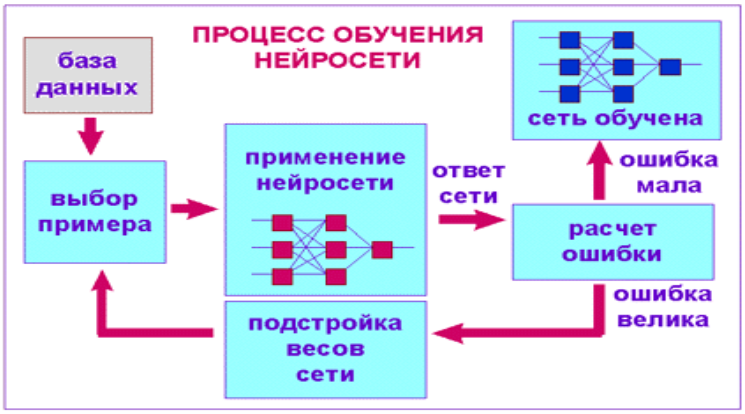

Рисунок 12. Процесс обучения с учителем

В используемых на практике нейросетях количество весов может составлять несколько десятков тысяч, поэтому обучение – сложный процесс. Для многих архитектур разработаны специальные алгоритмы обучения, которые позволяют настроить веса сети определенным образом. Наиболее популярный из этих алгоритмов – алгоритм градиентного (скорейшего) спуска, использующий метод обратного распространения ошибки [3-5]. *Алгоритм обратного распространения ошибки*, используемый для обучения многослойной сети, это набор формул, который позволяет по вектору ошибки вычислить требуемые поправки для весов сети.

В Deductor откроем *Мастер обработки* и выберем в нем *Нейросеть*. Установим в качестве входных все поля, кроме поля Spam Настройку нормализации оставим по умолчанию. Аналогично предыдущим методам, указываем процентное соотношение обучающего и тестового множеств, а также случайный характер выбора элементов этих множеств. На 4 шаге необходимо задать структуру нейронной сети для этого указываем количество скрытых слоев и нейронов в них, функцию активации. Изменяя число скрытых слоев и нейронов в скрытых слоях, исследуем различные структуры нейронной сети. Выберем лучшую сеть по точности обучения (таблица 2)

В ходе экспериментов было выявлено (таблица 2), что лучше всего подходит нейросеть с 2 слоями по 6 нейронов в каждом.

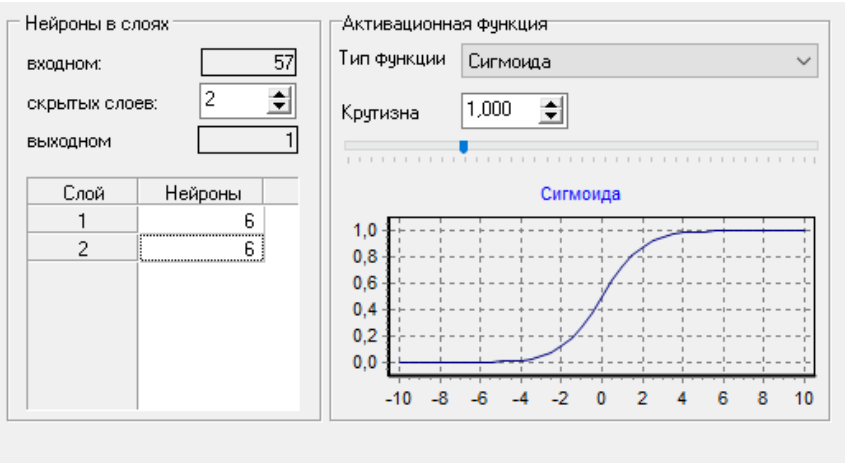

Рисунок 13. Настройка архитектуры сети

Таблица 2

Количество нейронов в слоях	Ошибка, %	
	на тестовом множестве	на обучающем множестве
57×10×1	6,09	5,99
57×30×1	6,09	4,53
57×6×6×1	**5,11**	**4,7**
57×25×30×1	6,85	4,73
57×12×12×1	5,65	5,08

Сравнивая полученные результаты с имеющимися в работе [7,8], можно сделать вывод что нам удалось получить лучший результат используя нейронные сети с различной архитектурой. Нейронные сети позволяют эффективно строить нелинейные зависимости, более точно описывающие наборы данных.

Заключение

В ходе теоретических исследований были изучены основные методы Data Mining, в частности логистическая регрессия; деревья решений; нейронные сети.

Для моделирования работы указанных алгоритмов была выбрана платформа Deductor, разработанная компанией BaseGroup Labs. Познакомились с возможностями платформы Deductor.

Для создания интеллектуального спам-фильтра исследовались возможность применения логической регрессии, деревьев решений, нейронных сетей. Был определен наиболее точный метод решения – нейронные сети. Экспериментально выбрана лучшая структура сети: многослойного персептрона с двумя скрытыми слоями по 6 нейронов в каждом из слоев, ошибка сети составила 5%.

В процессе работы были выполнены все поставленные задачи и достигнута цель.

Следует отметить, что данная сеть будет нуждаться в переобучении с течением времени, на новых, накопившихся при использовании спам-фильтра данных. Что сделает сеть адаптируемой к изменяющимся параметрам внешней среды.

Список литературы и источников

1. Паклин Н. Б., Орешков В. И. Бизнес-аналитика: от данных к знаниям.— СПб.: Питер, 2013. — 704 с.
2. Татьяна Куликова. Спам и фишинг в 2020 году (15.02.2021) // https://securelist.ru/spam-and-phishing-in-2020/100408/ Дата обращения: 19 сентября 2021.
3. Искусственная нейронная сеть [Электронный ресурс]. URL: http://ru.wikipedia.org/wiki/Искусственная_нейронная_сеть
4. Элементарное введение в технологию нейронных сетей с примерами программ / Р. Тадеусевич, Б. Боровик, Т. Гончаж, Б. Леппер. — М: Горячая линия-Телеком, 2011. — С. 408.
5. Бурков Андрей. Машинное обучение без лишних слов.— СПб.: Питер, 2020. — 192 с.
6. http://www.machinelearning.ru/wiki/index.php?title=UCI Дата обращения: 22 октября 2021.
7. Spambase Data Set http://archive.ics.uci.edu/ml/datasets/Spambase Дата обращения: 21 октября 2021.
8. Задача распознавания спама http://www.statproject.ru/publ/analiz_i_obrabotka_dannykh_osnovnye_zadachi/dobycha_dannykh_data_mining/zadacha_raspoznavanija_spama/26-1-0-47 Дата обращения: 22 декабря 2021.

Приложение 1

№	Исходное название	Перевод
1	word_freq_make	Частота слова “сделать”
2	word_freq_address	Частота слова “адрес”
3	word_freq_all:	Частота слова “все”
4	word_freq_3d	Частота слова “3d”
5	word_freq_our	Частота слова “наш”
6	word_freq_over	Частота слова “над”
7	word_freq_remove	Частота слова “снять”
8	word_freq_internet	Частота слова “интернет”
9	word_freq_order	Частота слова “порядок”
10	word_freq_mail	Частота слова “почта”
11	word_freq_receive	Частота слова “получить”
12	word_freq_will	Частота слова “буду”
13	word_freq_people	Частота слова “люди”
14	word_freq_report	Частота слова “отчет”
15	word_freq_addresses	Частота слова “адреса”
16	word_freq_free	Частота слова “свободный”
17	word_freq_business	Частота слова “бизнес”
18	word_freq_email	Частота слова “email”
19	word_freq_you	Частота слова “ты”
20	word_freq_credit	Частота слова “кредит”
21	word_freq_your	Частота слова “ваш”
22	word_freq_font	Частота слова “шрифт”
23	word_freq_000	Частота сочетания символов “000”
24	word_freq_money	Частота слова “деньги”
25	word_freq_hp	Частота слова “hp”
26	word_freq_hpl	Частота слова “hpl”
27	word_freq_george	Частота слова “Джордж”
28	word_freq_650	Частота сочетания символов “650”
29	word_freq_lab	Частота слова “лаборатория”
30	word_freq_labs	Частота слова “лаборатории”
31	word_freq_telnet	Частота слова “телнет”
32	word_freq_857	Частота сочетания символов “857”
33	word_freq_data	Частота слова “данные”
34	word_freq_415	Частота сочетания символов “415”
35	word_freq_85	Частота сочетания символов “85”
36	word_freq_technology	Частота слова “технологии”
37	word_freq_1999	Частота сочетания символов “1999”
38	word_freq_parts	Частота слова “части”
39	word_freq_pm	Частота слова “pm”
40	word_freq_direct	Частота слова “непосредственный”
41	word_freq_cs	Частота слова “cs”
42	word_freq_meeting	Частота слова “встреча”

43	word_freq_original	Частота слова “оригинал”
44	word_freq_project	Частота слова “проект”
45	word_freq_re	Частота слова “ повторно ”
46	word_freq_edu	Частота слова “edu”
47	word_freq_table	Частота слова “стол”
48	word_freq_conference	Частота слова “конференция”
49	char_freq ;	Частота символа ;
50	char_freq (	Частота символа (
51	char_freq_[	Частота символа [
52	char_freq_!	Частота символа !
53	char_freq_$	Частота символа $
54	char_freq_#	Частота символа #
55	capital_run_length_average	самая длинная последовательность прописных букв
56	capital_run_length_longest	средняя длина всех последовательностей прописных букв
57	capital_run_length_total	общая длина всех последовательностей прописных букв
58	spam	определение спам или нет {0, 1}