ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 8](#_Toc514513810)

[1. БИЗНЕС-АНАЛИЗ 9](#_Toc514513811)

[Вывод 11](#_Toc514513812)

[2. СБОР ДАННЫХ 12](#_Toc514513813)

[Вывод 15](#_Toc514513814)

[3. АНАЛИЗ И ПОДГОТОВКА СОБРАННЫХ ТОВАРОВ 16](#_Toc514513815)

[3.1. Поиск выбросов в наименовании товара 18](#_Toc514513816)

[3.2. Поиск выбросов в тексте категории товара 20](#_Toc514513817)

[3.3. Поиск выбросов в тексте характеристик 22](#_Toc514513818)

[3.4. Анализ каталога 26](#_Toc514513819)

[Вывод 30](#_Toc514513820)

[4. ПОСТРОЕНИЕ МОДЕЛЕЙ МАШИННОГО ОБУЧЕНИЯ 31](#_Toc514513821)

[4.1. Постановка задачи машинного обучения 31](#_Toc514513822)

[4.2. Метрики качества классификации 33](#_Toc514513823)

[4.3. Кросс-валидация 36](#_Toc514513824)

[4.4. Выбор метода построения модели машинного обучения 39](#_Toc514513825)

[4.5. Выбор способа преобразования признаков 41](#_Toc514513826)

[4.5.1. Преобразования TFIDF 41](#_Toc514513827)

[4.5.2. Добавление биграмм 42](#_Toc514513828)

[4.5.3. Нормализация текста 43](#_Toc514513829)

[4.5.4. Выбор оптимальных параметром для стохастического градиентного спуска 44](#_Toc514513830)

[4.6. Очистка выборки с помощью обученной модели 46](#_Toc514513831)

[Вывод 47](#_Toc514513832)

[5. ВВОД КЛАССИФИКАТОРА В ЭКСПЛУАТАЦИЮ 48](#_Toc514513833)

[6. ИСПОЛЬЗОВАННЫЕ ТЕХНОЛОГИИ 50](#_Toc514513834)

[6.1. Jupyter notebook 50](#_Toc514513835)

[6.2. Pandas 52](#_Toc514513836)

[6.3. Docker 53](#_Toc514513837)

[6.4. Другие технологии 54](#_Toc514513838)

[7. ИДЕИ ДЛЯ ДАЛЬНЕЙШЕГО УЛУЧШЕНИЯ МОДЕЛИ 56](#_Toc514513839)

[ЗАКЛЮЧЕНИЕ 58](#_Toc514513840)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 59](#_Toc514513841)

ВВЕДЕНИЕ

Компания Proanalytics занимается мониторингом цен на товары в интернет-магазинах, и предоставляет различные отчеты для магазинов конкурентов по собранным данным о товарах с различных сайтов. Для компании важно иметь собственный классификатор товаров, который на данный момент составляется вручную. Это представляет огромные затраты и большой объем рутинной работы. На данный момент существуют технологии, которые позволяют автоматизировать эту работу.

Данная выпускная квалификационная работа решает реальную проблему бизнеса. Цель этой работы — разработать систему, позволяющую гораздо быстрее классифицировать товары с минимальными затратами ресурсов.

Для достижения этой цели решено применить методы интеллектуального анализа, и были поставлены следующие задачи:

1) Изучить теоретические основы интеллектуального анализа данных, основные технологии и библиотеки для практического применения методов

2) Собрать и проанализировать данные

3) Выбрать метрику и способ оценки качества моделей

4) Построить и оценить различные модели

5) Внедрить модель в промышленное применение

В ходе разработке будут исследованы различные способы подготовки данных и методы машинного обучения.

Более того, данная работа может оказаться полезной не только для конкретной компании, поскольку в данной работе будет проведено исследование различных методов. Это может дать пользу последующим исследованиям в области обработки естественного языка.

# 1. БИЗНЕС-АНАЛИЗ

Компания ProAnalytics ежедневно собирает информацию о миллионах цен в день с различных сайтов — интернет-магазинов. В нее входит наименование цен, текст категории, изображение товара, характеристики, цена, описание, наличие. Для отдельных заказчиков может собираться дополнительно другая информация.

Клиентами компании являются вендоры либо интернет-магазины. Как правило, они заказывают отчеты по ценам на товары конкурентов. Для многих отчетов требуется информация по товарам в разрезе определенных категорий. Проблема заключается в том, что у каждого интернет-магазина своя структура каталога товаров. Поскольку информация собирается с различных интернет-магазинов, и у каждого своя структура каталога, то собранные товары требуется классифицировать в единый каталог.

На данный момент эта классификация производится вручную работниками компании. Т.е. работники вручную привязывают, по определенному списку, некоторые товары к категориям к которым эти товары относятся.

Целью является автоматизация этого процесса. Необходимо разработать программный продукт, который снизит затраты, и позволит более оперативно классифицировать товары, повышая довольство клиентов.

За период существования компании было привязано порядка миллиона товаров. Накопленные данные можно исследовать, и на их основе создать классификатор, который смог бы автоматически классифицировать товар.

Процесс ручной классификации выглядит следующим образом. Работник открывает специальную панель, на который выпадает список товаров, для которых необходимо определить категорию. Для них он выбирает нужную категорию. Для некоторых товаров автоматически предлагаются категории по ключевым словам, составленным вручную, и работник только выбирает «Да» или «Нет».

На данный момент автоматическое определение категории работает с плохим качеством, и не применяет методов машинного обучения. Оно работает следующим образом. Имеется некий словарь ключевых слов для некоторых категорий. Эти слова могут быть положительными, то есть их присутствие в наименовании и категории товара говорит о том, что товар относится к категории. Есть слова отрицательные, наличие которых говорит о том что, товар не должен быть в этой категории. Слова в словаре составлены вручную.

В процессе сбора данных было предложено дополнительно собирать нажатие кнопки «Нет». Т.е. при нажатии «Нет» в базе должна сохраняться запись, которая говорит о том, что конкретный товар не принадлежит конкретной категории. Эти данные позволят улучшать модель машинного обучения.

## **Вывод**

В данной главе была кратко описана деятельность компании. К тому же изучен процесс, связанные классификации товаров, который на данный момент осуществляется вручную.

Данный анализ позволит осуществить сбор нужных данных, и построить классификатор товаров.

# 2. СБОР ДАННЫХ

Первый этап разработки это просмотр и анализ имеющейся информации. А так же выбор нужной информации для использования в построении модели машинного обучения.

Вся информация по товарам хранится в единой базе данных. В качестве СУБД используется MySQL.

Был проведен опрос стейкхолдеров, для выяснения того, как и какие данные собираются, в каких таблицах можно найти ценную информацию. Так же было необходимо выяснить насколько зашумленными могут быть данные. В итоге удалось выяснить следующее.

Всего накоплено 1770580 товаров привязанных к товару.

Данные, которые собираются с интернет-магазинов по товарам:

* Наименование
* Текст категории
* Статус (В наличии / Нет в наличии)
* Цена
* Ссылка на изображение
* Характеристики товаров
* Описание
* Артикул товара на сайте
* Дата сбора
* Ссылка на товар
* Компания (интернет-магазин), с которой был собран товар
* Бренд

У всех товаров собираются:

* наименование
* текст категории
* статус
* цена, ссылка на изображение
* артикул товара на сайте
* дата сбора
* ссылка на товар
* ссылка на изображение

И только для некоторых компаний собираются:

* Характеристики товаров
* Описание
* Бренд

Среди товаров у который есть привязка к категории

* Товаров с характеристиками: 382087
* С описанием: 209651
* С брендами: 1284510

Данные о привязках верны, и без ошибок. Так утверждал стейкхолдер, но, как оказалось, этому утверждению нельзя доверять.

Для построения первой версии модели машинного обучения было решено использовать текстовую информацию, поскольку она наиболее полно характеризует категорию товара:

1. Наименование
2. Текст категории
3. Характеристики

Поскольку многие товары в обучающей выборе не имеют описания, эту информацию решено не включать в обучающую выборку.

Бренды также были исключены, поскольку они содержат много ошибок, и определяются автоматически через наименование товара.

Картинки так же можно использовать, но хорошее качество в классификации они способны показать лишь при использовании глубокого обучения, на многослойных нейронных сетях, которые способны генерировать высокоуровневые признаки из изображений. Но для их обучения требуется большое количество изображений и вычислительных ресурсов. Поэтому для начала было решено их не использовать.

После отсеивания пропусков и лишних данных было получено 342012 товаров.

## **Вывод**

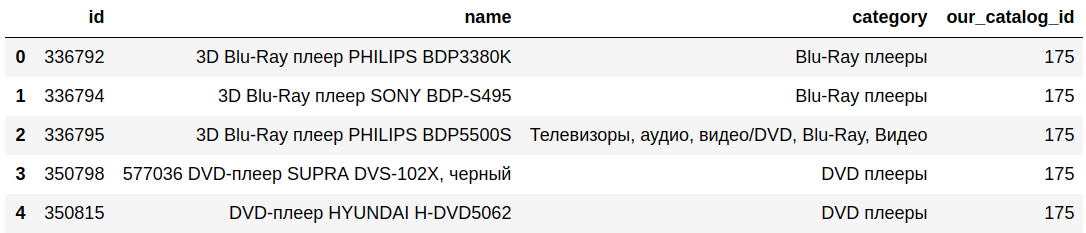
В этой главе была исследована база данных компании, изучен объем имеющихся данных. К тому же был проведен опрос стейкхолдеров о качестве данных, о том, как эти данные были получены.

Среди имеющихся данных были выбраны подходящие для построения модели машинного обучения, и исключены лишние.

# 3. АНАЛИЗ И ПОДГОТОВКА СОБРАННЫХ ТОВАРОВ

Прежде чем как либо обрабатывать данные, нужно их проанализировать. Нужно понять какие ошибки встречаются в данных. Возможно где-то может не хватать данных. Для начала найдем выбросы.

Для начала посмотрим как выглядят наименования и категории товаров. Для этого выведем 5 товаров в виде таблицы (Рисунок 3.1).

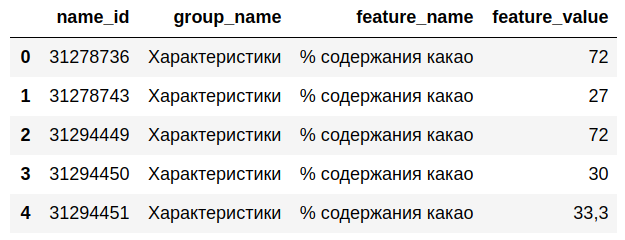
Рисунок 3.1 - Примеры наименования и категории товаров

В таблице id — это идентификатор товара в базе, name — наименование, category — текст категории, our\_catalog\_id — идентификатор каталога.

Как мы видим наименования и текст категории содержат различный регистр символов, символы пунктуации, и прочие символы, которые необходимо заменить на пробел. Символы верхнего регистра необходимо перевести в нижний. Необходимо оставить только буквы, числа и пробелы. К тому же в тексте встречаются слова, состоящие только из чисел, такие слова тоже удалим.

Структура каталога имеет 4 уровня вложенности. Но товары относятся только к последнему 4-му уровню. Т.е. к первым трем уровням привязок товаров нет.

К тому же были собраны характеристики товаров. Эти характеристики иметь привязку один товар ко многим характеристикам. Т.е. несколько записей характеристик могут относиться к одному товару. Посмотрим записи собранных из базы характеристик.

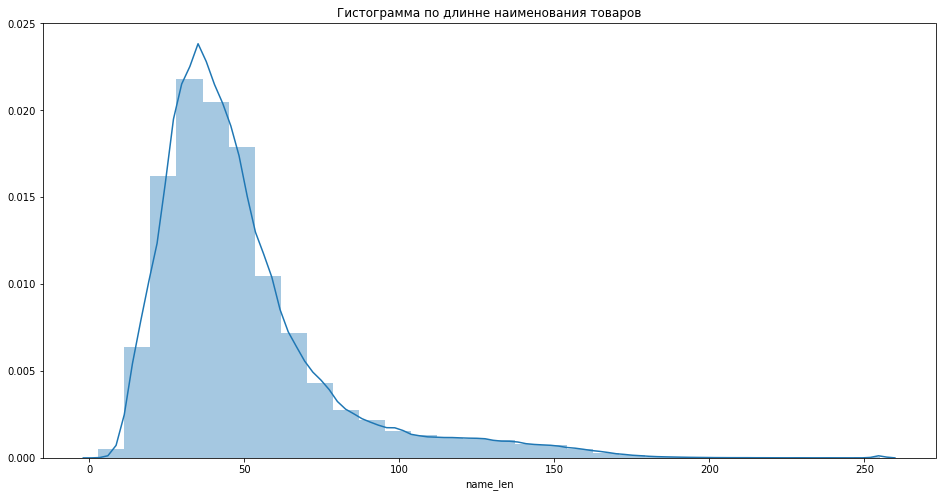
Рисунок 3.2 - Примеры характеристик 

Здесь group\_name — это наименование группы характеристик. Характеристики могут быть сгруппированы. feature\_name — название характеристики, feature\_value — значение характеристики.

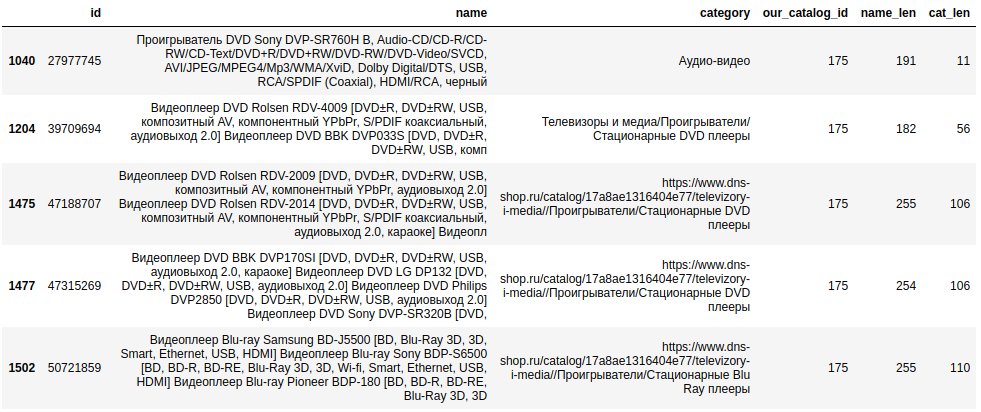
## **3.1. Поиск выбросов в наименовании товара**

В статистике выбросы – это значения, резко отличающиеся от других значений в собранном наборе данных. Выброс может указывать на аномалии в распределении данных или на ошибки при измерениях, поэтому зачастую выбросы исключаются из набора данных. Исключив выбросы из набора данных, вы можете прийти к неожиданным или более точным выводам. Поэтому необходимо уметь вычислять и оценивать выбросы, чтобы обеспечить надлежащее понимание статистических данных.

Для того чтобы найти выбросы, посмотрим на наименования с слишком малой длинной наименования и слишком большой. Для этого построим гистограмму, по которой можно увидеть распределение количества товаров из выборки по длине наименования.

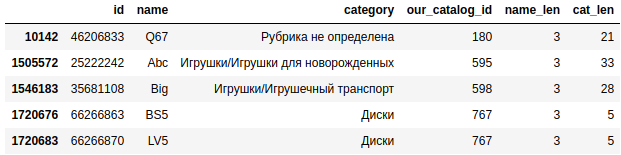
Рисунок 3.3 – Гистограмма по длине наименования товаров 

По гистограмме видно, что количество товаров, у которых длинна наименования меньше 180, резко уменьшается, практически до 0. Но в конце после 250 увеличивается. Это подозрительно. Посмотрим на товары у которых длинна наименования больше 180. Для примера возьмем 5 товаров.

Рисунок 3.4 – Товары с длинным наименованием 

Видно, что у этих товаров в наименовании присутствуют характеристики. К тому же у них присутствуют ошибки в тексте категории. Поскольку таких товаров немного, удалим их из выборки.

Далее посмотрим на слишком короткие наименования. Возьмем товары, с длинной наименования менее 4-х символов.

Рисунок 3.5 – Товары с коротким наименованием 

У этих товаров в наименовании только текст модели. Также удалим их, поскольку их не много, а текст модели никак не характеризует категорию.

Так же мы видим, что у товаров может быть категория «Рубрика не определена». Для товаров где в тексте категории содержится «Рубрика не определена», будем оставлять пустой текст.

## **3.2. Поиск выбросов в тексте категории товара**

Для поиска выбросов в тексте категории построим такую же гистограмму, как в пункте выше, только для текста категории.

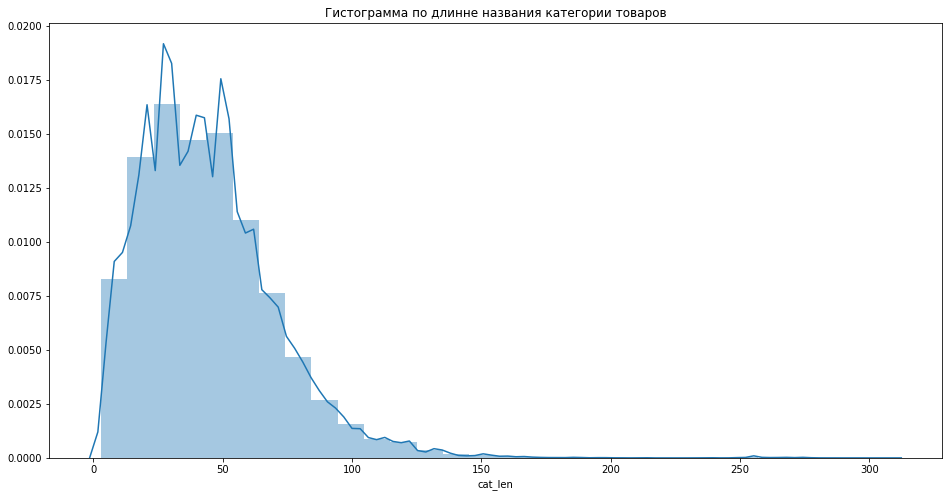


Рисунок 3.6 – Гистограмма по длине текста категории

Видим что есть аномальные, по длине категории, товары, при длине больше 150 символов. Посмотрим на эти товары.

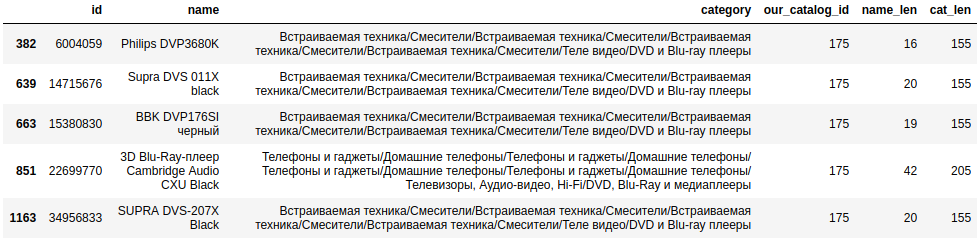


Рисунок 3.7 – Товары с длинным текстом категории

В этих товарах дубли категории. Удалим их, поскольку их также немного.

Посмотрим на слишком короткие категории с длинной текста менее 4-х символов.

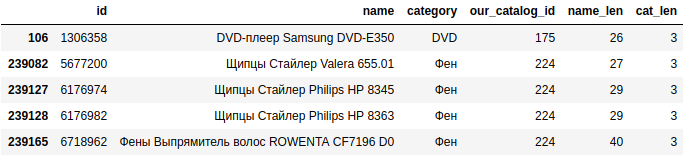


Рисунок 3.8 – Товары с длинным текстом категории

У этих товаров все нормально, поэтому их можно оставить.

## **3.3. Поиск выбросов в тексте характеристик**

Прежде чем искать выбросы, сразу удалим из характеристик различные спецсимволы, такие как «%», «-», «@», знаки пунктуации, и другие. Переведем все в нижний регистр. Также удалим слова которые состоят только из чисел.

Построим гистограммы по названию групп характеристик.

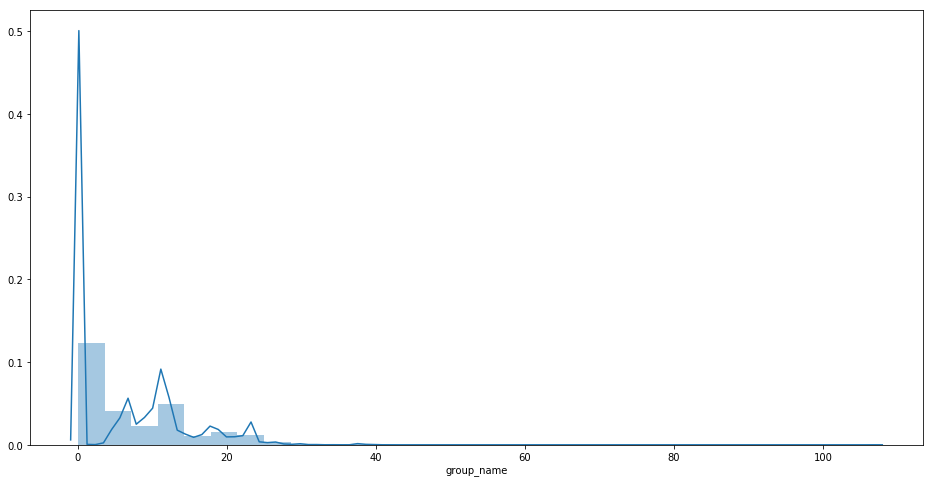


Рисунок 3.9 – Гистограмма по длине наименования группы характеристик

Для наименования групп аномальными являются характеристики с длинной более 40 символов.

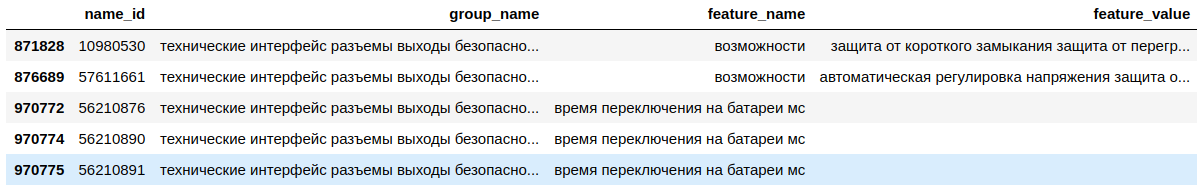


Рисунок 3.10 – Характеристики с длинным названием группы

Среди этих товаров мы видим, что названия групп собраны неправильно, эти характеристики с ошибками, удалим их.

Построим гистограммы по названию характеристик.



Рисунок 3.11 – Гистограмма по длине наименования характеристик

Видим, что при длине более 50-ти символов начинаются аномалии. Посмотрим на эти характеристики.

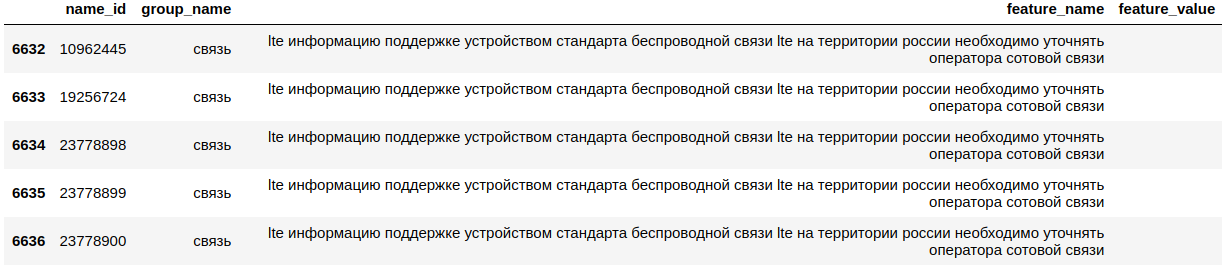


Рисунок 3.12 – Характеристики с длинным названием

Видно, что у этих характеристик, название ошибочное. Эти характеристики были неправильно собранны с интернет-магазинов. Удалим эти характеристики.

Теперь посмотрим гистограмму по длине значения характеристик (поле feature\_value).

Аномальные характеристики, начинаются при длине более 40 символов. Посмотрим на них.



Рисунок 3.13 – Гистограмма по длине значения характеристик

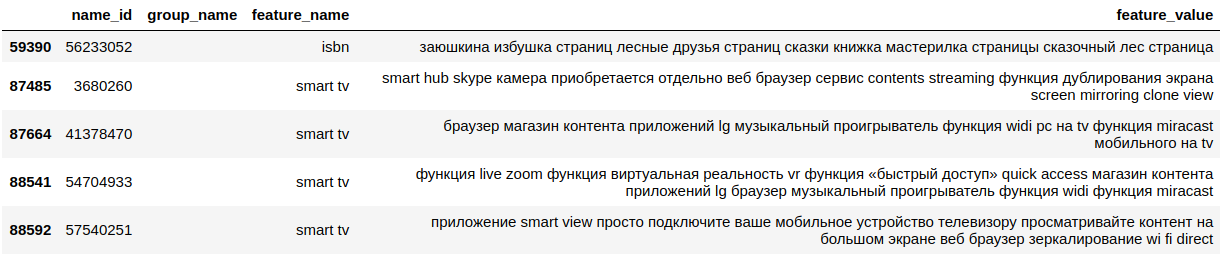


Рисунок 3.14 – Характеристики с длинным текстом значения

У этих товаров так же ошибки. Эти характеристики были собраны неправильно с интернет-магазинов.

Далее преобразуем все характеристики в мешок слов для каждого товара. Т.е. возьмем группу характеристик склеим её со всеми наименованиями и значениями характеристик.

Затем объединим мешки характеристик с наименованиями и посмотрим на полученную выборку.



Рисунок 3.15 – Товары с характеристиками в виде мешка слов

Видно, что в тексте категорий остались лишние слова. На примере последний товар имеет слова «главная», «каталог», который нужно удалить из текста категории.

## **3.4. Анализ каталога**

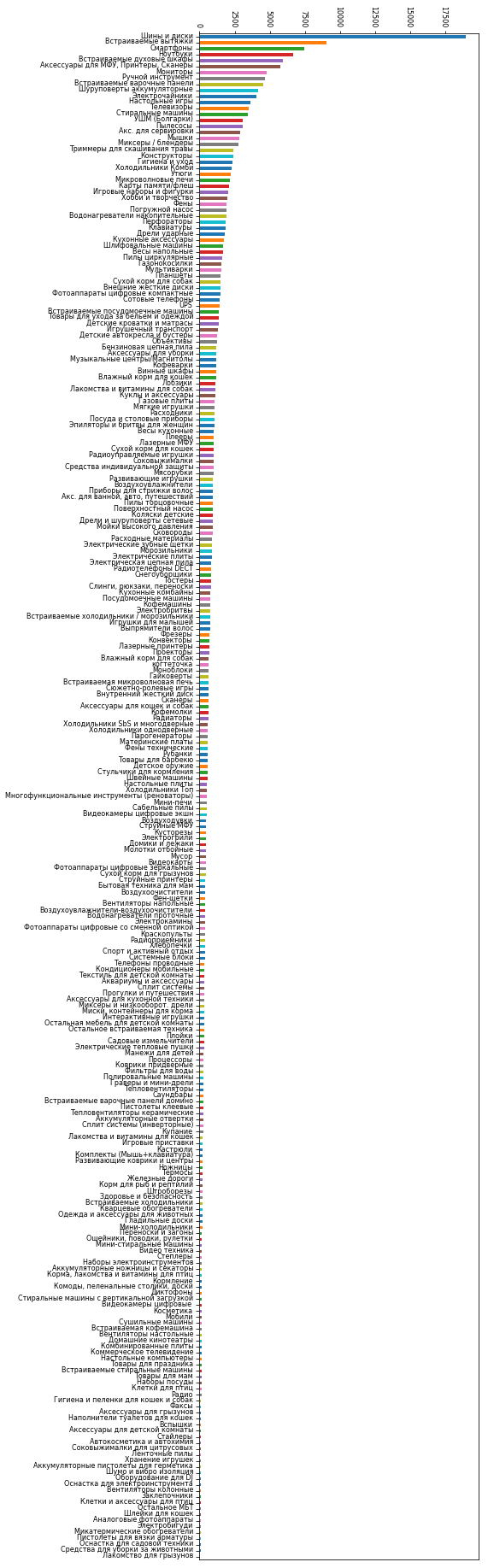
После визуального просмотра каталогов, было выявлено, что некоторые категории товаров сильно похожи друг на друга. Например категория 3-го уровня «Телевизоры» содержит подкатегории: «LCD/LED-телевизоры 15"-30"», «LCD/LED-телевизоры 32"-38"», «LCD/LED-телевизоры 39"-42"» и т. д. Т.е. эти категории содержат телевизоры с различной диагональю. Было выявлено несколько аналогичных категорий. Эти категории трудно классифицировать методами машинного обучения. Поэтому было решено товары этих категорий классифицировать к 3-му уровню каталога. Соответственно, в обучающей выборке эти категории были заменены на категорию родителя.

После преобразования получилось всего 255 категорий в обучающей выборке.

Ниже представлен график количества категорий (рис 3.16).

Из графика видно, что выборка несбалансированная. Товаров в категории «Шины и диск» гораздо больше чем остальных. К тому же, во многих категориях очень мало товаров, что может плохо сказаться на качестве классификации товаров. Для обучения таких категорий может просто не хватить данных.

Следующей проблемой являются пустые категории. Другими словами, имеются категории, к которым не привязан не один товар. Когда будет нужно классифицировать реальные товары, если попадутся товары из категорий, которых не было в обучающей выборке, то эти товары будут классифицироваться в категории, которые не были пустыми. Это проблема нецелостных данных. Например, в обучающей выборке нет категории наушники. Но когда будет нужно классифицировать товары некоторого интернет-магазина, то там могут оказаться наушники, и тогда модель машинного обучения не сможет правильно определить его категорию.

Рисунок 3.16 – Количество товаров в категориях

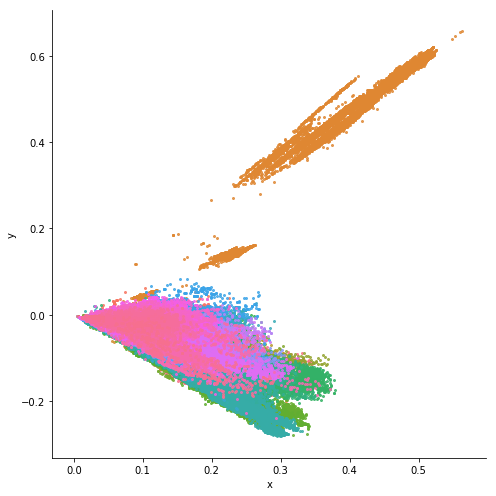
Хотелось бы посмотреть на все объекты, в данном случае объекты — это товары, в двухмерном пространстве, и понять, есть ли какая либо закономерность в классах. Можно ли их вообще отделить друг от друга с неким приемлемым качеством. Для этого воспользуемся методом t-SNE (t-distributed stochastic neighbor embedding) и преобразуем выборку в двухмерное пространство.

Алгоритм t-SNE, был опубликован в 2008 году[6] голландским исследователем Лоуренсом ван дер Маатеном и Джеффри Хинтоном. Классический SNE был предложен Хинтоном и Ровейсом в 2002[7]. В статье 2008 года описывается несколько «трюков», которые позволили упростить процесс поиска глобальных минимумов, и повысить качество визуализации. Кроме того, была сделана удачная реализация алгоритма, которая потом портировалась в другие популярные среды.

Для того чтобы перевести товары в числовые векторы склеим наименование, текст категории и характеристики в одну строку, и преобразуем все признаковое пространство с помощью меры TF-IDF.

На рисунке представлен график, где точками отмечены товары (рис 3.17). Для каждой категории используется свой цвет. Из графика видно, что некоторые категории хорошо отделяются от других, некоторые плохо, но в целом есть некая закономерность.

Этот график дает понять, что данные можно использовать для обучения моделей и наши объекты можно будет классифицировать на различные категории с помощью машинного обучения.

Рисунок 3.17 – Выборка в двумерном пространстве

## **Вывод**

В этой главе были изучены собранные данные. В первую очередь рассмотрена информация о товарах, и их характеристиках. Был описан процесс поиска выбросов в выборке товаров и характеристик. Были рассмотрены возможные ошибки в тексте товаров.

Так же был проанализирован каталог товаров, была построена диаграмма катерий по количеству товаров. Были рассмотрены проблемы в текущей структуре каталога.

После очистки и обработки выборки была построена диаграмма распределения товаров в двухмерном пространстве, которая горовит о том, что в данных есть некая закономерность, и с помощью выборки можно построить модель машинного обучения, которая сможет выделять с некоторой точностью категории товаров.

# 4. ПОСТРОЕНИЕ МОДЕЛЕЙ МАШИННОГО ОБУЧЕНИЯ

## **4.1. Постановка задачи машинного обучения**

Требуется решать задачу многоклассовой классификации текстов. В качестве обучающей выборки используется множество товаров с информацией о наименовании товара, тексте категории и мешка слов из характеристик товара. В качестве целевой метки будут использованы идентификаторы категории товара.

Для того чтобы преобразовать текстовую информацию в признаковое пространство, т.е. для того, чтобы каждый товар был представлен числовым вектором, будет применена мера TFIDF.

TFIDF - статистическая мера, используемая для оценки важности слова в контексте документа. Вес каждого слова пропорционален количеству употребления этого слова в документе, и обратно пропорционален частоте употребления слова в других документах коллекции. В качестве документа в данном случае представлена текстовая информация о товаре. В виде формулы она выглядит следующим образом:

здесь

где ni - число вхождений слова в документ. А в знаменателе общее число слов в документе

где |D| - количество документов в выборке, - количество документов, в которых встречается ti (когда ni≠0)

Большой вес в TFIDF получат слова с высокой частотой в пределах конкретного документа и с низкой частотой употреблений в других документах.

В качестве меры был взят TFIDF, из-за того, что эта мера позволяет сильно уменьшить вклад в ответ классификатора для распространенных слов, таких как союзы предлоги, частицы и другие слова не характеризующие категорию.

Мера TF-IDF часто используется для представлении документов коллекции в виде числовых векторов, отражающих важность использования каждого слова из некоторого набора слов (количество слов набора определяет размерность вектора) в каждом документе. Подобная модель называется векторной моделью (VSM) и даёт возможность сравнивать тексты, сравнивая представляющие их вектора в какой либо метрике (евклидово расстояние, косинусная мера, манхэттенское расстояние, расстояние Чебышева и др.), т. е. производя кластерный анализ.

Формальная постановка задачи будет выглядеть следующим образом.

Пусть X — множество описаний объектов, Y — конечное множество номеров (имён, меток) классов. Существует неизвестная целевая зависимость — отображение , значения которой известны только на объектах конечной обучающей выборки

Требуется построить алгоритм , способный классифицировать произвольный объект .

Признаком называется отображение , где Df — множество допустимых значений признака. Если заданы признаки f1,…,fn, то вектор называется признаковым описанием объекта . В данном случае f1,…,fn - меры TFIDFдля каждого уникального слова в выборке по тексту товаров.

## **4.2. Метрики качества классификации**

В качестве метрик классификации были рассмотрены следующие:

* Доля верных ответов (Accuracy)
* Точность (Precision)
* Полнота (Recall)
* F – мера (F1)

Каждый из рассматриваемых метрик предназначен для бинарной классификации. Но в этой задаче 255 классов, и к тому же классы несбалансированны, поэтому будет рассматриваться взвешенная (по количеству объектов в классе) усредненная сумма оценок по каждому классу.

Доля верных ответов –самая простая метрика, представляет собой сумму верных ответов классификатора на общее число объектов в тестовой выборке. Она не подойдет для классификации из-за того, что плохо оценивает несбалансированную выборку.

Точность - это доля документов действительно принадлежащих данному классу относительно всех документов, которые система отнесла к этому классу.

Полнота - это доля найденных классификатором документов принадлежащих классу относительно всех документов этого класса в тестовой выборке.

Если представить ответы классификатора в виде матрицы, то матрица будет выглядеть следующим образом (рис 4.1).



Рисунок 4.1 – Матрица оценок

В таблице содержится информация о том, сколько раз система приняла верное и сколько раз неверное решение по документам заданного класса. А именно:

TP — истинно-положительное решение;

TN — истинно-отрицательное решение;

FP — ложно-положительное решение;

FN — ложно-отрицательное решение.

Тогда, точность и полнота определяются следующим образом:

Точность и полнота отражают разные качества классификатора. Высокая точность говорит о том, что классификатор редко дает ошибочные положительные решения, но может пропускать верные ответы, и это никак не скажется на оценке. Высокая полнота наоборот, говорит о том, что классификатор пытается найти как можно больше верных положительных ответов, не никак не учитывает число ошибочных верных ответов. Т.е. максимальная полнота будет в случае положительного решения ко всем объектам.

Для верной оценки нужно как то обобщить эти 2 метрики. Что балансировать между этими двумя метриками существует F–мера. F-мера представляет собой гармоническое среднее между точностью и полнотой.

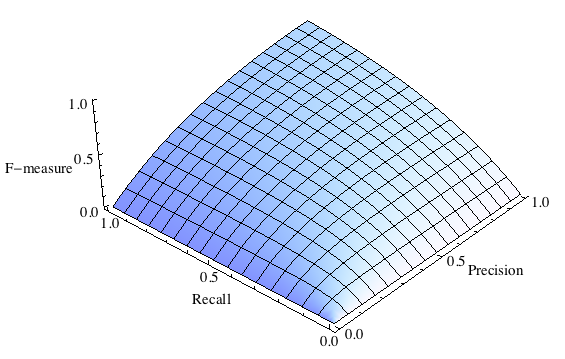


Рисунок 4.2 – График F-меры

На рисунке 4.2 представлен график зависимости F-меры от точности и полноты. Видно, что она стремится к нулю, если точность или полнота стремятся к нулю.

Для оценки классификации была выбрана Fмера, поскольку она позволяет адекватно оценивать несбалансированную выборку.

## **4.3. Кросс-валидация**

Для оценивания модели была применена кросс-валидация по блокам из компаний.

Кросс-валидация (Cross-validation) — она же «перекрёстная проверка», метод оценки аналитической модели и её поведения на независимых данных. При оценке модели имеющиеся в наличии данные разбиваются на k частей. Затем на k−1 частях данных производится обучение модели, а оставшаяся часть данных используется для тестирования. Процедура повторяется k раз, в итоге каждая из k частей данных используется для тестирования. В результате получается оценка качества выбранной модели с наиболее равномерным использованием имеющихся данных.

Обычно кросс-валидация используется в ситуациях, где целью является предсказание, и хотелось бы оценить, насколько предсказывающая модель способна работать на практике. Один цикл кросс-валидации включает разбиение набора данных на части, затем построение модели на одной части (называемой тренировочным набором), и валидация модели на другой части (называемой тестовым набором). Чтобы уменьшить разброс результатов, разные циклы кросс-валидации проводятся на разных разбиениях, а результаты валидации усредняются по всем циклам.

Кросс-валидация важна для защиты от гипотез, навязанных данными («ошибки третьего рода»), особенно когда получение дополнительных данных затруднительно или невозможно.

Цель кросс-валидации в том, чтобы оценить ожидаемый уровень соответствия модели данным, независимым от тех данных, на которых модель тренировалась. Она может использоваться для оценки любой количественной меры соответствия, подходящей для данных и модели. Например, для задачи бинарной классификации, каждый случай в тестовом наборе будет предсказан правильно или неправильно. В этой ситуации коэффициент ошибки может быть использован в качестве оценки соответствия, хотя могут использоваться и другие оценки. Если предсказываемое значение распределено непрерывно, для оценки соответствия может использоваться среднеквадратичная ошибка, корень из среднеквадратичной ошибки или медианное абсолютное отклонение.

В каждый блок были выбраны компании таким образом, чтобы в блок попадало как можно больше различных категорий, и чтобы блоки были приблизительно равного размера.

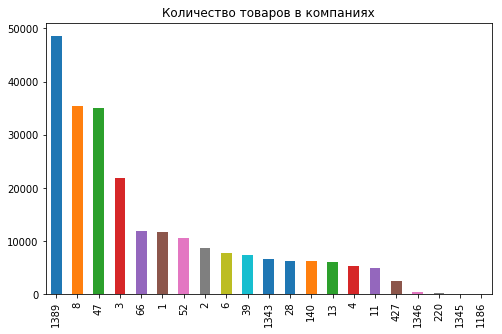


Рисунок 4.3 – Количество товаров в разных компаниях

Если не проводить разбиение по блокам из компаний, а брать блоки из случайных объектов, то оценка может быть неправильной. Это может произойти из-за того, что если на обучение и тест попадут товары из одной компании и одной категории, то они могут иметь абсолютно одинаковый текст категории. А это не правильно, поскольку из-за этого модели могут обучиться некорректно, и переобучиться на тексте категорий.

На рисунке 4.3 представлено количество товаров в компаниях. Для того чтобы не нарушать коммерческую тайну, компании заменены их идентификаторами. На рисунке видно, что есть компании с очень маленьким числом товаров в выборке, а есть наоборот.

Было проведено следующее разбиение по блокам:

* Блок 1: 1389
* Блок 2: 8, 47
* Блок 3: 3, 66, 52
* Блок 4: 1, 2, 6, 39
* Блок 5: 1343, 28, 140, 13, 4, 11, 427, 1346, 220, 1345, 1186

При таком разбиении блоки имеют почти одинаковое количество товаров и наибольшее число категорий в каждом блоке.

## **4.4. Выбор метода построения модели машинного обучения**

В данной работе было рассмотрено 2 типа моделей машинного обучения: линейные модели и модели, основанные на решающих деревьях.

Линейные модели имею преимущество над решающими деревьями в том плане, что они гораздо быстрее работают, за счет своей простоты и лучше работают с данными большой размерности. С другой стороны, модели, представляющие собой ансамбли из решающих деревьев, могут выявлять скрытые закономерности в данных.

Были рассмотрены следующие методы построения моделей машинного обучения:

1. Стохастический градиентный спуск
2. Метод опорных векторов
3. Градиентныйбустинг над решающими деревьями

Для моделей были выставлены следующие параметры:

Для стохастического градиентного спуска:

* Кусочно-линейная функция потерь
* L2 – нормализация,
* Коэффициент регуляризации: 0.00001,
* максимальное число итераций: 5

Для метода опорных векторов:

* Ядро: Радиальная базисная функция
* Параметр регуляризации: 1

Для градиентногобустинга над решающими деревьями:

* Число деревьев: 50
* Максимальная глубина деревьев: 3

Из выборки были взята часть случайных данных, а конкретно 20000 товаров, и на них проведена кросс-валидация по этим трем методам. Это сделано для того, чтобы меньше ждать результат оценки. Результаты кросс-валидации представлены в таблице.

Таблица 4.1 – Качество классификации различных методов

|  |  |
| --- | --- |
| Метод | F - мера |
| Стохастический градиентный спуск | 0.84 |
| Метод опорных векторов | 0.1 |
| Градиентногобустинг над решающими деревьями | 0.46 |

По результатам оценки можно видеть, что стохастический градиентный спуск справляется заметно лучше остальных алгоритмов.

Это говорит о том, что, скорее всего признаки имеют линейную зависимость от целевой переменной. К тому же после преобразования текста в векторы, число признаков получилось примерно равным 10000. Это очень разряженная выборка, поскольку в ней много нулей, и только несколько признаков имеют ненулевое значение в одной строке, а именно столько, сколько уникальных слов в тексте товара. Но именно линейные модели хорошо работаю с разряженными выборками большой размерности.

Именно для метода стохастического градиентного спуска и было решено подбирать оптимальные параметры.

## **4.5. Выбор способа преобразования признаков**

### **4.5.1. Преобразования TFIDF**

В качестве признаков были взяты TFIDF по наименованию товара и тексту категории склеенных в одну строку, и TFIDF мешка слов по характеристикам. Два разных преобразований TFIDF были объединены. Это было сделано из следующих соображений. Я предположил, что слова в наименовании и категории могут иметь другой смысл, чем в характеристиках. В характеристиках может попасться слово, например, разъем, и оно может выражать тип разъема, который имеет музыкальный плеер, в то время как в наименованиях оно вряд ли будет иметь такой смысл, скорее всего в наименовании слово разъем уже будет характеризовать, что этот товар сам является неким разъемом.

Моя теория о том, что наименования с категориями нужно преобразовывать отдельно от характеристик, была проверена. В таблице представлены оценки классификации различных способов преобразования текста.

Таблица 4.2 – Качество классификации при различных склейках

|  |  |
| --- | --- |
| Варианты склеек | F - мера |
| (Наименование + категория), (Характеристики) | 0.84 |
| (Наименование + категория + Характеристики) | 0.82 |
| (Наименование), (категория), (Характеристики) | 0.81 |

Как видно из таблицы, моя теория подтверждается, склейка «(Наименование + категория), (Характеристики)» показывает лучшую оценку.

### **4.5.2. Добавление биграмм**

N-грамма—последовательностьиз n элементов. Биграммы – это последовательность из 2-х элементов. В данном случае элементами являются слова.

Для каждого товара в тексте дополнительно были добавлены пары слов, расположенных по соседству, склеенных символом «\_». Это позволило учитывать последовательность пар слов в тексте.

В таблице представлены результаты оценки различных вариантов использования биграмм методом стохастического градиентного спуска.

Таблица 4.3 – Качество классификации с биграммами

|  |  |
| --- | --- |
| Вариант | F - мера |
| С биграммами в тексте категории | 0.84 |
| С биграммами в тексте характеристик | 0.83 |
| С биграммами в тексте наименования | 0.86 |

Биграммы в тексте наименования дают небольшое улучшения качества. Его будем использовать в дальнейшем.

### **4.5.3. Нормализация текста**

Нормализацией текста называется приведение всех слов текста к словарной форме: к именительному падежу, единственному числу (если таковое есть) или инфинитиву для глаголов. [4]

Нормализация нужна, например, для быстрого поиска слова в словарях, синтаксического и семантического разбора текста. Эта процедура особенно актуальна для языков грамматической группы как, например, русский или финский, у которых богатая морфология (сильные словоизменения в следствии грамматической вариативности).

В данной работе нормализация слов не дала результатов, а даже немного ухудшила результат.

F - мера для нормализованных слов: 0.83.

Возможно, это связанно с тем, что склонения слов говорят многое о товаре, и как то характеризуют категорию товара. К примеру, содержание текста «для монитора» уже может говорить о том, что данный товар не монитор, а скорее всего аксессуар для монитора, в то время как содержание слова «монитор» говорит, что это товар относится к категории мониторы. Если же нормализовать текст «для мониторов», то он преобразуется в текст «длямонитор», и модели будет тяжелее классифицировать товар верно.

Поэтому решено не применять нормализацию текстов.

### **4.5.4. Выбор оптимальных параметром для стохастического градиентного спуска**

Суть градиентного спуска – минимизировать функцию, делая небольшие шаги в сторону наискорейшего убывания функции. Название методу подарил тот факт из математического анализа, что вектор частных производных функции задает направление наискорейшего возрастания этой функции. Значит, двигаясь в сторону антиградиента функции, можно уменьшать значения этой функции быстрее всего.

Метод приспособлен для динамического обучения, когда обучающие объекты поступают потоком, и надо быстро обновлять вектор.

Алгоритм способен обучаться на избыточно больших выборках за счёт того, что случайной подвыборки может хватить для обучения.

Возможны различные стратегии обучения. Если выборка избыточно большая, или обучение происходит динамически, то допустимо не сохранять обучающие объекты. Если выборка маленькая, то можно повторно предъявлять для обучения одни и те же объекты.

Для того чтобы подобрать наилучшие параметры модели была использована сетка поиска параметров[18]. Поиск заключается в переборе всех сочетаний параметровсреди заданных.

Для поиска были заданы следующие параметры:

1. Функция потерь, только одна – логистическая. Логистическая функция потерь позволяет получать вероятность отнесения товара к каждой категории. Благодаря вероятности, можно отличать более надежную классификациюот менее надежной. Это позволит в некоторой мере решить проблему отсутствия некоторых категорий, которая была описана в разделе «Анализ каталога», путем отбрасывая результаты ненадежной классификации.
2. Регуляризация: L1 и L2
3. Максимальное число итераций: от 4 до 40 с шагом 4
4. Коэффициент регуляризации: 9 \* 10-6, 9 \* 10-5, 9 \* 10-3
5. Каждый класс имеет вес ошибки, это позволяет бороться с несбалансированностью выборки

После перебора всех сочетаний, самую лучшую оценку показал следующий набор параметров:

* Коэффициент регуляризации: 9 \* 10-6
* Регуляризация: L2
* Функция потерь: логистическая
* Максимальное число итераций: 8
* Классы взвешены по размеру в выборке

## **4.6. Очистка выборки с помощью обученной модели**

После обучения модели на всех размеченных данных было решено посмотреть, как модель будет классифицировать товары на неразмеченных данных. Были просмотрены ответы классификатора с уверенной классификацией, где одной категории давался ответ с вероятностью более 60%.

При просмотре ответов были выявлены серьезные ошибки, некоторые товары попадали в категории, которые не имеют ничего общего с этим товаром.

Например, наушники попадали в видеорегистраторы.

Я предположил, что могут быть ошибки в выборке. Для этого было решено посмотреть слова, с наибольшим коэффициентом в линейной модели, по категориям, в которых были обнаружены ошибки.

И действительно, оказалось, что для категории видеорегистраторы, наибольший вес был у слова наушники. Это подтверждало теорию.

Оставалось только найти все товары в категории видеорегистраторы содержащие слово наушники и удалить их, а затем заново обучить модель.

Таким образом, в несколько итераций была очищена выборка от серьезных ошибок, что дало несколько плюсов.

Во-первых, был очищен от ошибок каталог товаров, что могло привести к недовольству клиентов.

Во-вторых, это позволило улучшить качество классификации.

В результате качество было улучшено до F1 = 0.9.

## **Вывод**

В главе был выбран и описан способ преобразования выборки в признаковое пространство, и каждый товар был описан с помощью числового вектора.

Затем рассмотрены различные метрики оценки модели машинного обучения. Среди рассматриваемых метрик был выбран наиболее подходящий. Для оценки был использован метод кросс-валидации по блокам из компаний.

Были исследованы различные алгоритмы для построения модели машинного обучения на имеющихся данных. В качестве наиболее подходящего был выбран метод стохастического градиентного спуска и подобраны для него оптимальные параметры.

Были исследованы такие способы преобразования как нормализация текста и добавление биграмм. Добавление биграмм дало небольшое повышение качества классификации.

К тому же, благодаря построенной модели удалось избавиться от серьезных ошибок в выборке.

На данном этапе была получена оптимальная модель машинного обучения и метод подготовки данных для этой модели.

# 5. ВВОД КЛАССИФИКАТОРА В ЭКСПЛУАТАЦИЮ

Для создания конечной программы был написан код для загрузки новых товаров. Все этапы обработки информации о товаре и применения модели машинного обучения были собраны в единую библиотеку.

Разработанная система функционирует в автоматическом режиме, ежедневно классифицируя все новые товары вечером после 17:00.

Классификация проходит по 2-м типам товаров: по товарам с характеристиками и без характеристик.

Для того, чтобы была возможность классифицировать товары, не имея характеристики, была построена новая модель, обученная без характеристик, но с теми же параметрами. Она показывает более низкое качество – F1 = 0.74, но в результате её классификации берутся товары с более уверенной классификацией.

Результаты работы классификатора записываются в единую таблицу, дополнительно сохраняя информацию о дате и уверенности классификации.

Так же была разработана панель проверки результатов работы классификатора.

На изображении (рис. 5.1) представлен скриншот веб панели, предназначенной для проверки результатов работы классификатора.

В поле «Категория» указана категория товара на сайте, откуда товар был собран. В поле наша категория – категория товара в классификаторе Proanalytics, которую определила разработанная модель классификации.

При нажатии кнопки «Не соответствует», данные о несоответствии сохранятся в базу. Они могут пригодиться для улучшения качества классификации, для того чтобы исправить ошибки в классификации в текущей версии.

Панель для проверки представляет собой веб – приложение.

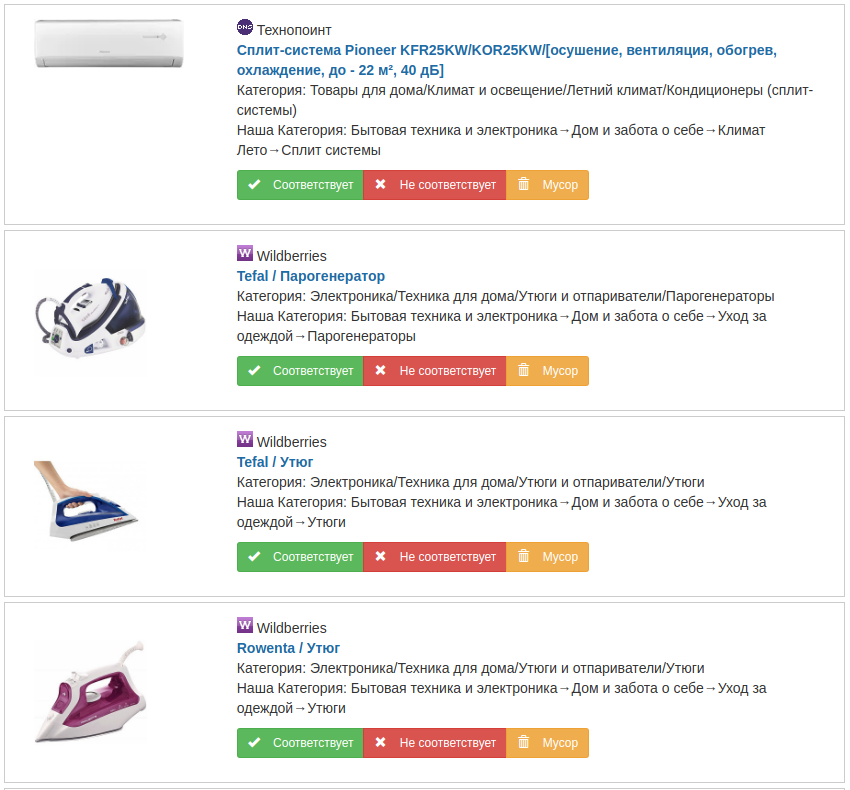


Рисунок 5.1 – Панель проверки классификации

# 6. ИСПОЛЬЗОВАННЫЕ ТЕХНОЛОГИИ

В качестве языка программирования использовался Python версии 2.7.

## **6.1. Jupyter notebook**

Для проведения анализа, и моделирования использовалось веб-приложение Jupyter notebook, которое позволяет производить работу с данными используя язык программирования Python, отображать данные сразу в окне браузера, и сохранять результаты работы в удобном виде.

На изобрабжении (рис 6.1) представлен скриншот веб-приложения Jupyter notebook.

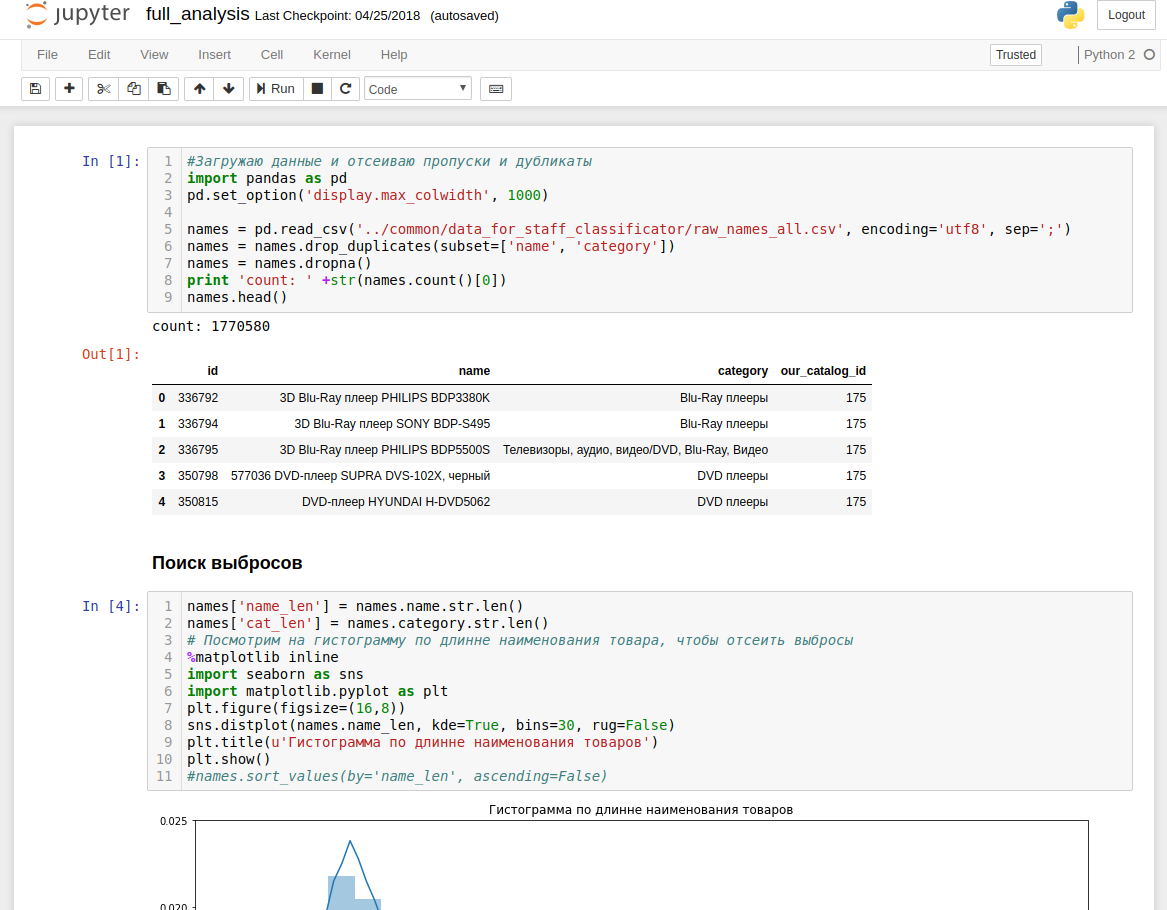


Рисунок 6.1 – Jupyter Notebook

Jupyter Notebook – это удобный инструмент для создания красивых аналитических отчетов, так как он позволяет хранить вместе код, изображения, комментарии, формулы и графики.

Раньше она называлась IPython Notebook, но название сменили, чтобы подчеркнуть совместимость не только с Python, но и другими языками программирования. Jupyter поддерживает множество языков программирования и может быть легко запущен на любом сервере, необходим только доступ по ssh или http. К тому же это свободное ПО.

## **6.2. Pandas**

Библиотека pandas позволила удобно загружать данные, преобразовывать, группировать, объединять и рассматривать в различных разрезах с помощью класса DataFrame.

Pandas — программная библиотека на языке Python для обработки и анализа данных. Работа pandas с данными строится поверх библиотеки NumPy, являющейся инструментом более низкого уровня pandas предоставляет специальные структуры данных и операции для манипулирования числовыми таблицами и временными рядами. Название библиотеки происходит от эконометрического термина «панельные данные», используемого для описания многомерных структурированных наборов информации. pandas распространяется под новой лицензией BSD.

Данный пакет делает Python мощным инструментом для анализа данных. Пакет дает возможность строить сводные таблицы, выполнять группировки, предоставляет удобный доступ к табличным данным, а при наличии пакета matplotlib дает возможность рисовать графики на полученных наборах данных.

## **6.3. Docker**

Разработанный программный продукт был изолирован в Docker контейнер для того, чтобы его можно было быстро разворачивать на любом сервере, без настройки окружения.

Docker — программное обеспечение для автоматизации развёртывания и управления приложениями в среде виртуализации на уровне операционной системы. Позволяет «упаковать» приложение со всем его окружением и зависимостями в контейнер, который может быть перенесён на любую Linux-систему с поддержкой cgroups в ядре, а также предоставляет среду по управлению контейнерами.

Приложения работают в изолированной среде (построенной с помощью пространств имён, namespaces, и групп процессов, cgroups), с одной стороны изолируя процессы друг от друга, с другой стороны не прибегая при этом к таким избыточным средствам как виртуализация или эмуляция.

Система написана на Go. Работает в различных UNIX/Linux-системах, ожидается, что в недалёком будущем будет поддерживаться и Windows.

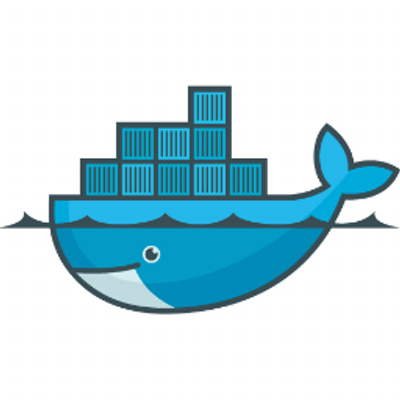


Рисунок 6.2 – Логотип Docker

## **6.4. Другие технологии**

Библиотека seaborn использовалась для визуализации данных.

Библиотека sklearn содержит множество алгоритмов обучения моделей машинного обучения. Она позволила быстро строить модели методами стохастического градиентного спуска в классе SGDClassifier и метода опорных векторов в классе SVC, а так же применить сингулярное разложение в классе TruncatedSVD и преобразовать текст с помощью меры TFIDF в классе TfidfVectorizer .

Для построения градиентного бустинга использовалась библиотека xgboost, в которой реализация метода лучше, чем в библиотеке sklearn. В библиотеке xgboost градиентный бустинг обучается заметно быстрее.

Для нормализации текста применялась библиотека pymorphy2.

Pymorphy2 написан на языке Python (работает под 2.7 и 3.3+). Он умеет:

* приводить слово к нормальной форме (например, “люди -> человек”, или “гулял -> гулять”).
* ставить слово в нужную форму. Например, ставить слово во множественное число, менять падеж слова и т.д.
* возвращать грамматическую информацию о слове (число, род, падеж, часть речи и т.д.)

Для составления биграмм использовал библиотеку nltk.

Библиотека NLTK, или NLTK, — пакет библиотек и программ для символьной и статистической обработки естественного языка, написанных на языке программирования Python. Содержит графические представления и примеры данных. Сопровождается обширной документацией, включая книгу с объяснением основных концепций, стоящих за теми задачами обработки естественного языка, которые можно выполнять с помощью данного пакета.

NLTK хорошо подходит для студентов, изучающих компьютерную лингвистику или близкие предметы, такие как эмпирическая лингвистика, когнитивистика, искусственный интеллект, информационный поиск и машинное обучение.NLTK с успехом используется в качестве учебного пособия, в качестве инструмента индивидуального обучения и в качестве платформы для прототипирования и создания научно-исследовательских систем. NLTK является свободным программным обеспечением. Проект возглавляет Стивен Бёрд.

# 7. ИДЕИ ДЛЯ ДАЛЬНЕЙШЕГО УЛУЧШЕНИЯ МОДЕЛИ

Данную модель возможно можно ещё улучшить за счет имеющихся данных. В данной работе не были рассмотрены картинки товаров. Их так же можно использовать для классификации товаров.

С классификацией по изображению лучше все справляются нейронные сети. Но у них есть огромный недостаток. Для обучения модели нужно очень большое число картинок товаров с различных ракурсов, под разным освещением для каждой категории. У нас же всего примерно около 2-х миллионов картинок, чего в разы недостаточно.

Но есть другое решение. Это использование предобученных нейронных сетей. Только крупные компании, такие как Google и Microsoft имеют огромное количество ресурсов и данных, и они создали такие модели, и предоставили их в общий доступ.

Существуют следующие предобученные нейронные сети:

* VGG16 - сеть VisualGeometryGroup из университета Оксфорда для распознавания объектов на изображениях, состоит из 16 слоев.
* VGG19 - еще одна сеть Visual Geometry Group для распознавания объектов, но содержит 19 слоев.
* Inception v3 - нейронная сеть компании Google для распознавания объектов на изображениях.
* ResNet50 - нейронная сеть компании Microsoft, использующая остаточное обучение (residual learning). Применяется для распознавании объектов на изображениях.
* Xception - модификация сети Inception от создателя KerasFrançoisChollet.
* CRNN for music tagging - сверточная рекуррентная нейронная сеть для классификации музыки.

Эти предобученые нейронные сети выложены в открытый доступ и их можно использовать в своей работе. Они обучены на общедоступном наборе данных ImageNet который содержит 14 миллионов изображений.

Один из вариантов использования этих сетей заключается в следующем.

У предобученной нейронной сети снимается последний слой. Затем все изображения, предварительно обработав, прогоняются через нейросеть. Полученные значения на предпоследнем слое сети, записываются в таблицу, и используются в качестве признаков. Эти признаки характеризуют различные высокоуровневые свойства объекта на изображении (это могут быть углы, формы, и т.д.). Затем на полученныхпризнаках можно обучить модель с помощью какого либо метода, например - градиентного бустинга.

На сайте kaggle проходило соревнование под названием «Cdiscount’s Image Classification Challenge». Оно заключалось в классификации товаров по изображению. Участники, занявшие призовые места использовали предобученную сеть Resnet50, с последующим обучение признаков с помощью градиентного бустинга над решающими деревьями.

Можно было бы использовать этот подход на существующем наборе данных и классифицировать товары по изображениям. Затем объединить 2 подхода, используя ответы двух классификатор, увеличив, таким образом, качество классификации.

# ЗАКЛЮЧЕНИЕ

В данной работе были рассмотрены все этапы разработки классификатора товаров.

Из данной работы я сделал вывод, что нельзя слепо верить утверждениям стейкхолдеров, а нужно проверять их, если есть возможность.

Большую часть работы составляет сбор и подготовка данных. На само моделирование требуется значительно меньше времени. Это связанно с тем, что реальные данные кроме ошибок содержат ещё множество аномалий и выбросов.

В данной работе были приведены результаты исследования различных способов подготовки признаков для моделей машинного обучения, и результаты моделирования с оценкой качества модели с помощью метрики F1.

В качестве методов машинного обучения были рассмотрены: стохастический градиентный спуск, метод опорных векторов, градиентный бустинг над решающими деревьями, и выбран наиболее подходящий.

В результате работы был разработан программный продукт, который работает на реальном предприятии и позволяет классифицировать товары в автоматическом режиме.

Так же была рассмотрена возможность дальнейшего развития данного продукта, и рассмотрен способ применения изображений товаров для улучшения качества классификации.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Гудфеллоу Я., Бенджио И. Глубокое обучение / пер. с анг. А. А. Слинкина. – 2-е изд., испр. – М.: ДМК Пресс, 2018. – 652 с.: цв. ил.
2. Блог компании ГК ЛАНИТ - CRISP-DM: проверенная методология для DataScientist-ов.– 2017 - <https://habr.com/company/lanit/blog/328858/>
3. Введение в анализ текстовой информации с помощью Python и методов машинного обучения.– 2013 -<https://habr.com/post/205360/>
4. Морфологический анализатор pymorphy2, Руководство - http://pymorphy2.readthedocs.io/en/0.1/user/index.html
5. Предварительно обученные нейронные сети в Keras.–2017 - <https://www.asozykin.ru/deep_learning/2017/06/06/keras-pretrained-networks>
6. L.J.P. vander Maatenand G.E. Hinton. Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research 9 (Nov): 2579 -2605, 2008. - http://lvdmaaten.github.io/publications/papers/JMLR\_2008.pdf
7. G.E. Hintonand S.T. Roweis. StochasticNeighborEmbedding. In Advancesin Neural Information. ProcessingSystems, volume 15, pages 833–840, Cambridge, MA, USA, 2002. The MIT Press. - http://www.cs.toronto.edu/~fritz/absps/sne.pdf
8. Pandas 0.22.0 documentation.- <http://pandas.pydata.org/pandas-docs/stable/index.html>
9. Seaborn documentation. - <http://seaborn.pydata.org>
10. Documentation of scikit-learn 0.19.1 - <http://scikit-learn.org/stable/documentation.html>
11. Model evaluation: quantifying the quality of predictions. - http://scikit-learn.org/stable/modules/model\_evaluation.html#precision-recall-f-measure-metrics
12. Stochastic Gradient Descent. - <http://scikit-learn.org/stable/modules/sgd.html>
13. Support Vector Machines. - <http://scikit-learn.org/stable/modules/svm.html>
14. Introduction to Boosted Trees. - <http://xgboost.readthedocs.io/en/latest/model.html>
15. Pavel Ostyakov - Oursolution [5th place].–2017 - <https://www.kaggle.com/c/cdiscount-image-classification-challenge/discussion/45733>
16. Cdiscount’s Image Classification Challenge. - 2017 - <https://www.kaggle.com/c/cdiscount-image-classification-challenge>
17. Francois Chollet - Xception: Deep Learning with Depth wise Separable Convolutions. - 2017 -<https://arxiv.org/pdf/1610.02357.pdf>
18. Sklearn GridSearchCV - <http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>
19. Геометрия машинного обучения. Разделяющие гиперплоскости или в чём геометрический смысл линейной комбинации? - <https://habr.com/post/324736/>
20. Векторная модель - http://www.machinelearning.ru/wiki/index.php?title=Векторная\_модель