**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.

Оглавление

[1. Бизнес-анализ 3](#_Toc513214642)

[2. Сбор данных 5](#_Toc513214643)

[3. Анализ и подготовка собранных данных 8](#_Toc513214644)

[3.1. Поиск выбросов в наименовании товара 10](#_Toc513214645)

[3.2. Поиск выбросов в тексте категории товара 12](#_Toc513214646)

[3.3. Поиск выбросов в тексте характеристик 14](#_Toc513214647)

[3.4. Анализ каталога 18](#_Toc513214648)

[4. Построение моделей машинного обучения 21](#_Toc513214649)

[4.1. Постановка задачи машинного обучения 21](#_Toc513214650)

[4.3. Метрики качества классификации 23](#_Toc513214651)

[4.4. Кросс-валидация 26](#_Toc513214652)

[4.5. Выбор метода построения модели машинного обучения 28](#_Toc513214653)

[4.6. Выбор способа преобразования признаков- 30](#_Toc513214654)

[4.6.1. Преобразования TFIDF 30](#_Toc513214655)

[4.6.2. Добавление биграмм 31](#_Toc513214656)

[4.6.3. Нормализация текста 32](#_Toc513214657)

[4.6.4. Выбор оптимальных параметром для стохастического градиентного спуска 33](#_Toc513214658)

[4.6. Очистка выборки с помощью обученной модели 34](#_Toc513214659)

[5. Ввод классификатора в эксплуатацию 35](#_Toc513214660)

[6. Использованные технологии 37](#_Toc513214661)

[7. Идеи для дальнейшего улучшения модели 39](#_Toc513214662)

[Заключение 41](#_Toc513214663)

[Библиографический список 42](#_Toc513214664)

# 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 собирается информация с различных интернет-магазинов, 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 которой 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».

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

# 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 собираются данные, 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:

1. Наименование
2. Текст категории
3. Статус (В наличии / Нет в наличии)
4. Цена
5. Ссылка на изображение
6. Характеристики товаров
7. Описание
8. Артикул товара на сайте
9. Дата сбора
10. Ссылка на товар
11. Компания (интернет-магазин), с которой был собран товар
12. Бренд

У всех товаров собираются:

1. наименование
2. текст категории
3. статус
4. цена, ссылка на изображение
5. артикул товара на сайте
6. дата сбора
7. ссылка на товар
8. ссылка на изображение

И только для некоторых компаний собираются:

1. Характеристики товаров
2. Описание
3. Бренд

????

Здесь привести количество товаров с хар-ми

С описанием

С брендами

????

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 характеризует категорию товара:

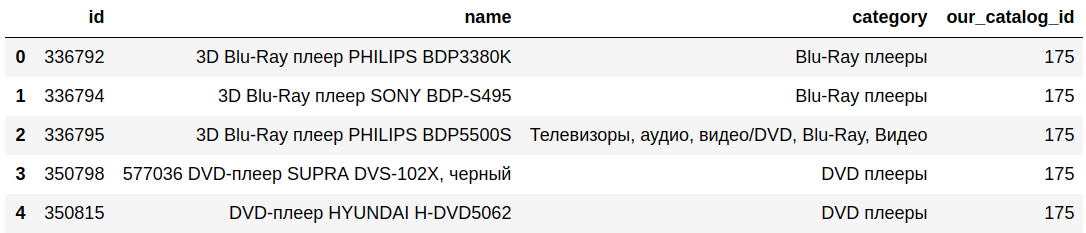
1. Наименование
2. Текст категории
3. Характеристики

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

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.

# Анализ и подготовка собранных данных

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

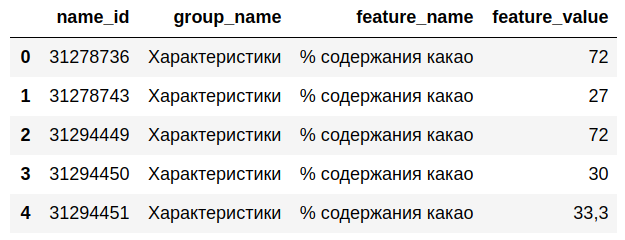


В таблице id — это идентификатор товара в базе, name — наименование, category — текст категории, our\_catalog\_id — идентификатор каталога.

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.

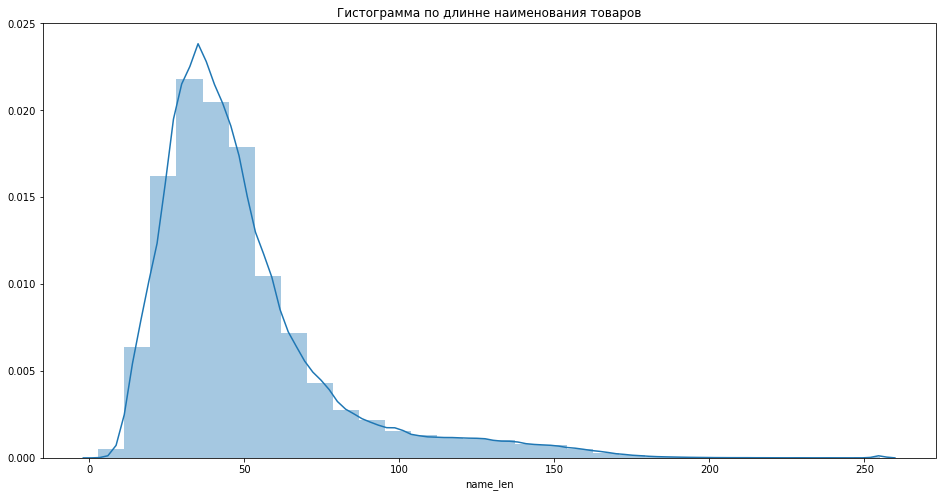
К тому же были собранны характеристики товаров. Эти характеристики иметь привязку один товар ко многим характеристикам. Т.е. несколько записей характеристик могут относиться к одному товару. Посмотрим записи собранных из базы характеристик.



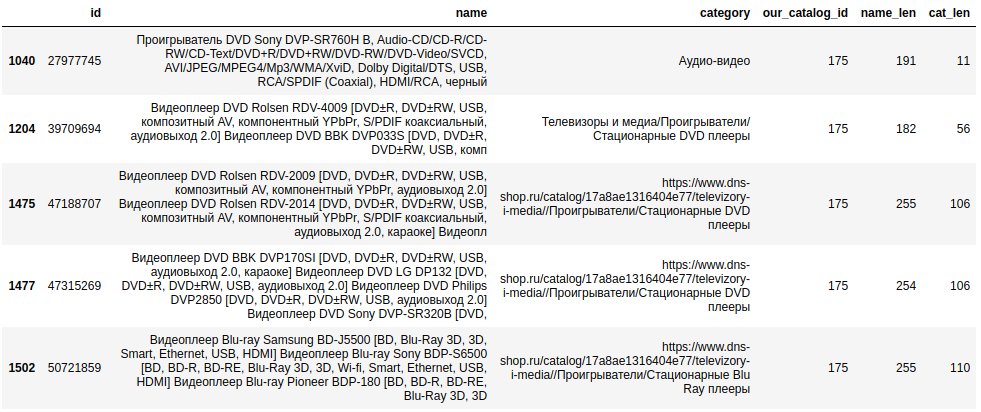
Здесь group\_name — это наименование группы характеристик. Характеристики могут быть сгруппированы. feature\_name — название характеристики, feature\_value — значение характеристики.

## 3.1. Поиск выбросов в наименовании товара

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

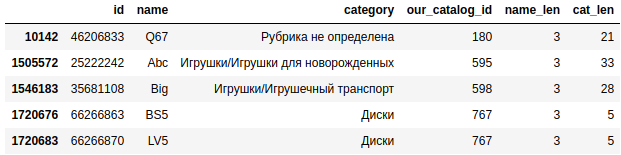


По гистограмме видно, что количество товаров, у которых длинна наименования меньше 180, резко уменьшается, практически до 0. Но в конце после 250 увеличивается. Это подозрительно. Посмотрим на товары у которых длинна наименования больше 180. Для примера возьмем 5 товаров.



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.

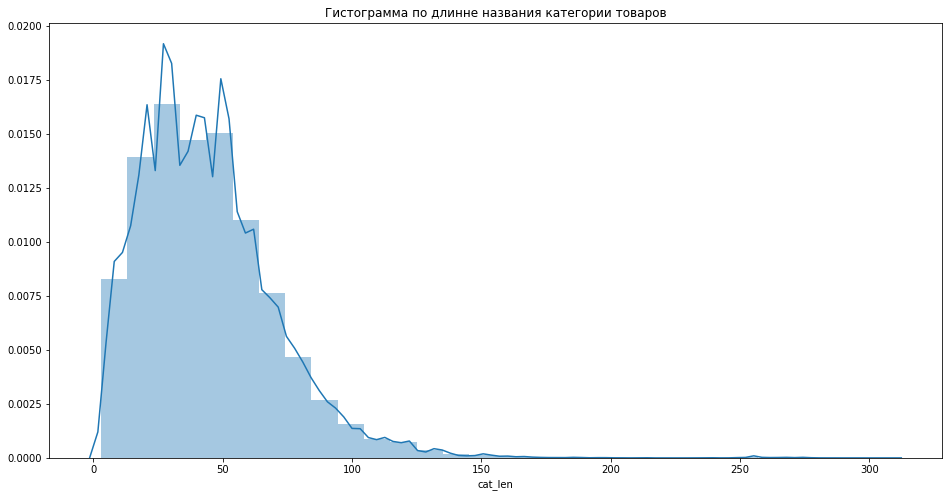


У этих товаров в наименовании только текст модели. Также удалим их, поскольку их не много, а текст модели никак не характеризует категорию.

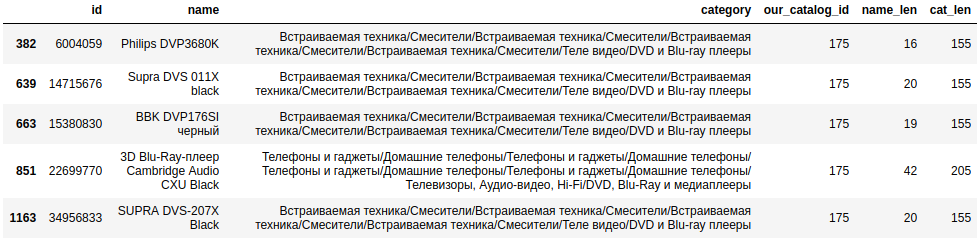
Так же мы видим, что у товаров может быть категория «Рубрика не определена». Для товаров где в тексте категории содержится «Рубрика не определена», будем оставлять пустой текст.

## 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.

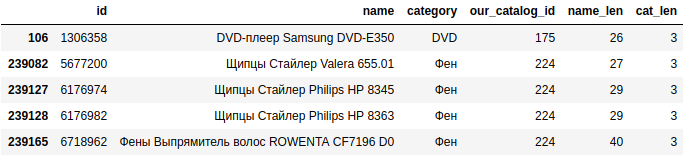


Видим что есть аномальные, по длине категории, товары, при длине больше 150 символов. Посмотрим на эти товары.



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

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

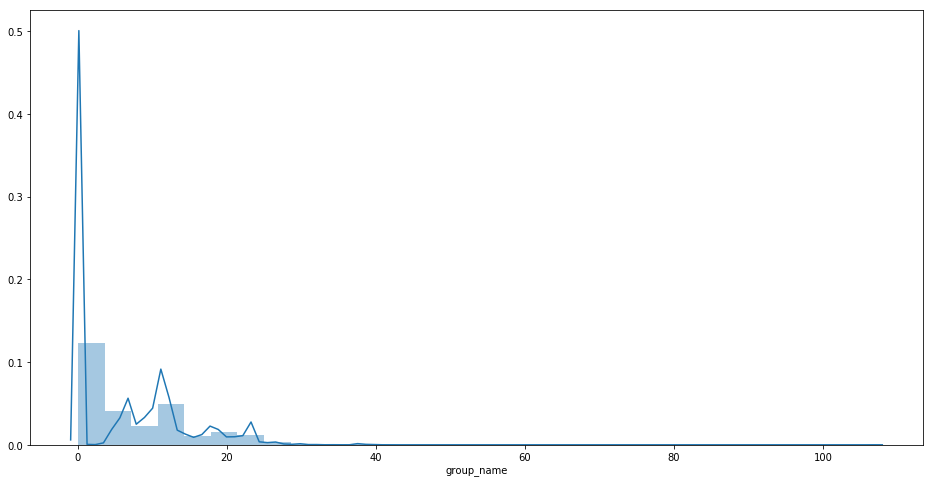


У этих товаров все нормально, поэтому их можно оставить.

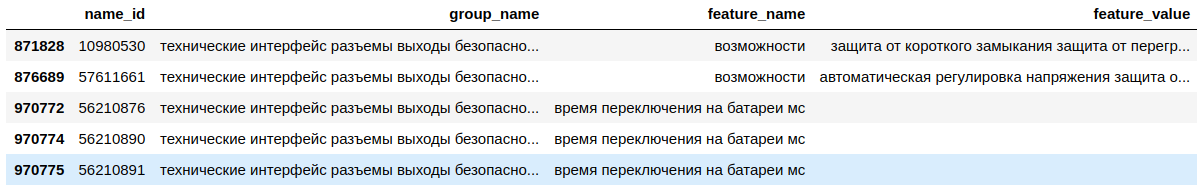
## 3.3. Поиск выбросов в тексте характеристик

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.



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

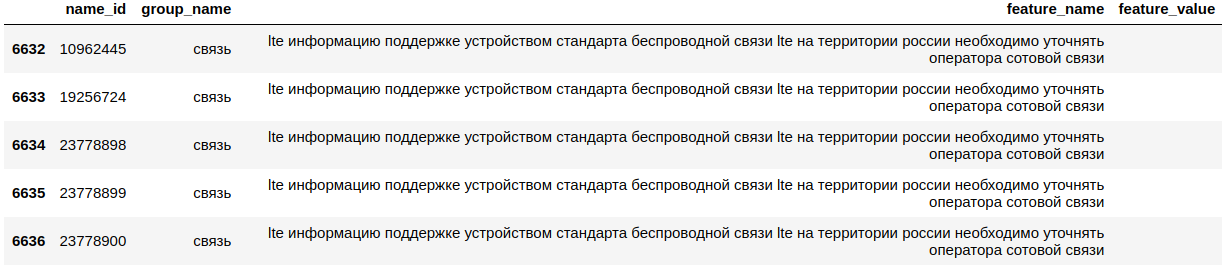


Среди этих товаров мы видим, что названия групп собраны неправильно, эти характеристики с ошибками, удалим их.

Построим гистограммы по названию характеристик.



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

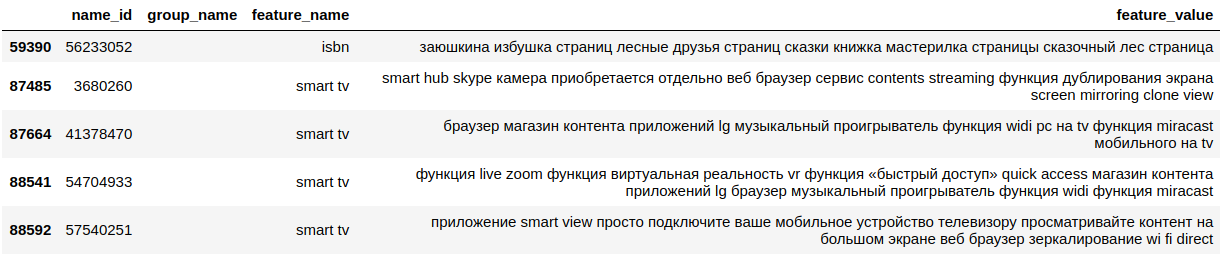


Видно, что у этих характеристик, название ошибочное. Эти характеристики были неправильно собранны с интернет-магазинов. Удалим эти характеристики.

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

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





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.



## 3.4. Анализ каталога

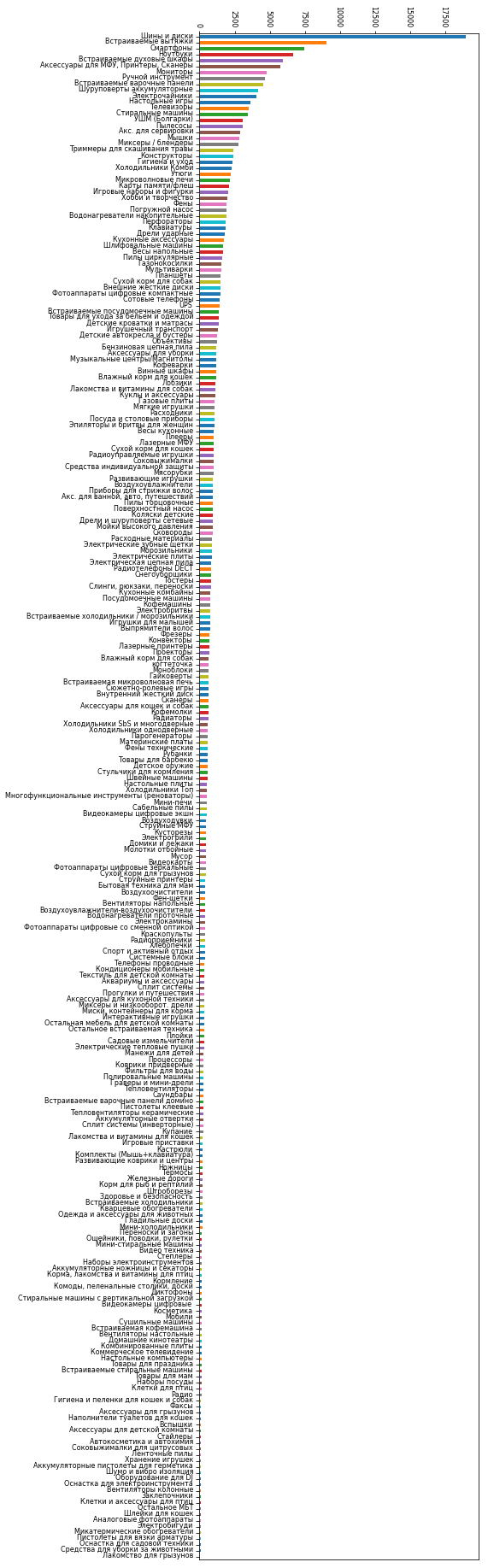
После визуального просмотра каталогов, было выявлено, что некоторые категории товаров сильно похожи друг на друга. Например категория 3-го уровня «Телевизоры» содержит подкатегории: «LCD/LED-телевизоры 15"-30"», «LCD/LED-телевизоры 32"-38"», «LCD/LED-телевизоры 39"-42"» и т. д. Т.е. эти категории содержат 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.

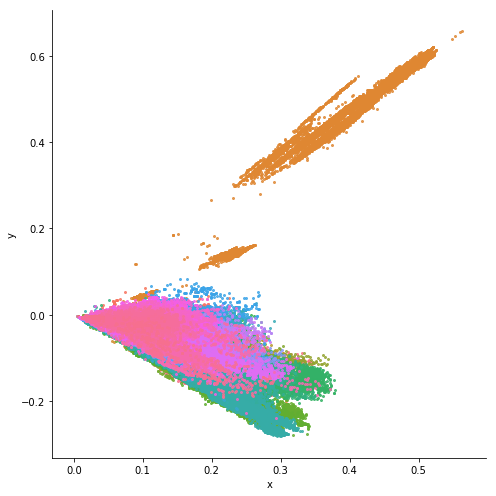
Из графика видно, что выборка несбалансированная. Товаров в категории «Шины и диск» гораздо больше чем остальных. К тому же, во многих категориях очень мало товаров, что может плохо сказаться на качестве классификации товаров. Для обучения таких категорий может просто не хватить данных.

Следующей проблемой являются пустые категории. 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.



Далее склеим наименование, текст категории и характеристики в одну строку, и преобразуем все признаковое пространство с помощью метрики TF-IDF.

Хотелось бы посмотреть на все объекты, в данном случае объекты — это товары, в двухмерном пространстве, и понять, есть ли какая лидо закономерность в классах. Можно ли их вообще отделить друг от друга с неким приемлемым качеством. Для этого воспользуемся сингулярным разложением и преобразуем выборку в двухмерное пространство.



На рисунке представлен график, где точками отмечены объекты. Для каждого категории используется свой цвет. Из графика видно, что некоторые категории хорошо отделяюся от других, некоторые плохо, но в целом есть некая закономерность. Этот график дает понять, что наши объекты можно классифицировать на различные категории с помощью машинного обучения.

# 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.

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 TFIDF для каждого уникального слова в выборке по тексту товаров.

## 4.3. Метрики качества классификации

В качестве метрик классификиции 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:



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

TP — истинно-положительное решение;

TN — истинно-отрицательное решение;

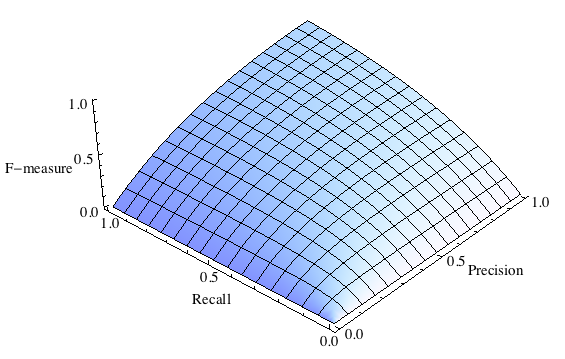
FP — ложно-положительное решение;

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.



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

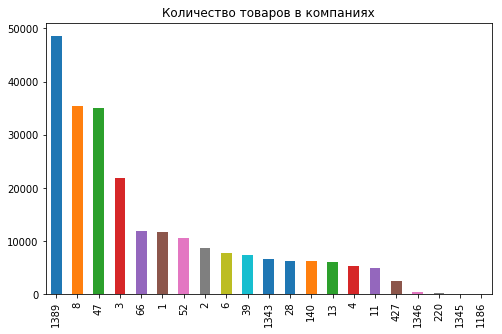
Для оценки классификации была выбрана F мера, поскольку она позволяет адекватно оценивать несбалансированную выборку.

## 4.4. 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.

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



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 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

## 4.5. Выбор метода построения модели машинного обучения

Были рассмотрены следующие методы построения моделей машинного обучения:

1. Стохастический градиентный спуск
2. Метод опорных векторов
3. Градиентный бустинг 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 градиентного бустинга над решающими деревьями:

* Число деревьев: 50
* Максимальная глубина деревьев: 3

Из выборки были взяты 20000 случайных товаров, и на них проведена кросс-валидация по этим трем методам. Результаты кросс-валидации представлены в таблице.

|  |  |
| --- | --- |
| Метод | F - мера |
| Стохастический градиентный спуск | 0.84 |
| Метод опорных векторов | 0.1 |
| Градиентного бустинг 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.

## 4.6. Выбор способа преобразования признаков-

4.6.1. 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.

|  |  |
| --- | --- |
| 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.

4.6.2. Добавление биграмм

N-грамма — последовательность из 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.

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

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

4.6.3. Нормализация текста

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

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 «для монитор», и модели будет тяжелее классифицировать товар верно.

Поэтому решено не применять нормализацию текстов.

4.6.4. Выбор оптимальных параметром для стохастического градиентного спуска

Для того чтобы подобрать наилучшие параметры модели была использована сетка поиска параметров. Поиск заключается в переборе всех сочетаний параметров среди 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 классификацию от 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.

# 5. Ввод классификатора в эксплуатацию

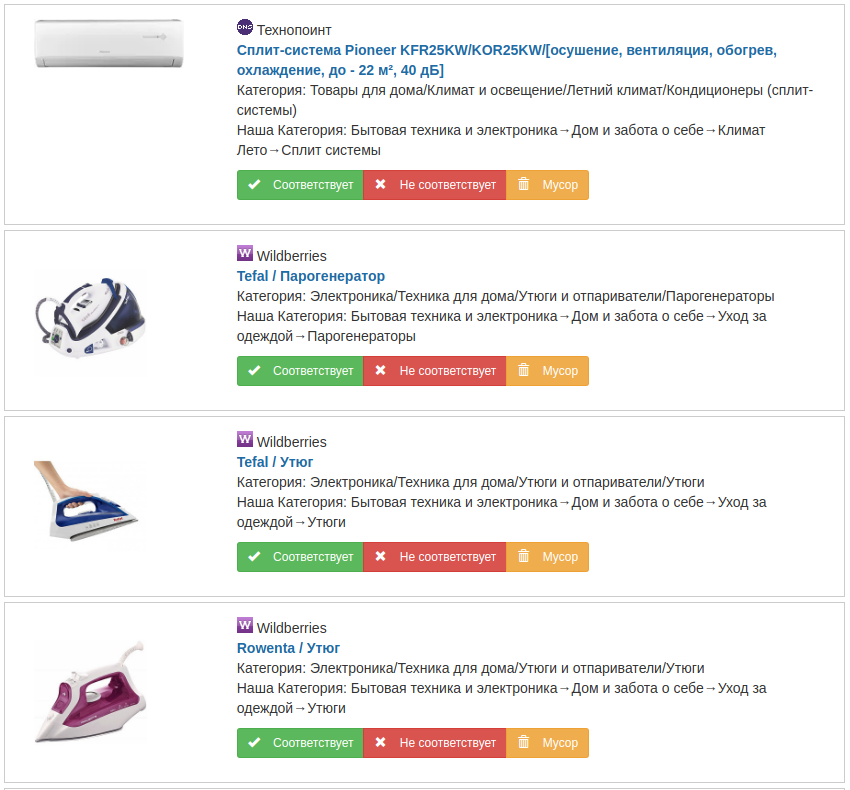
Для создания конечной программы был написан код для загрузки новых товаров. Все этапы обработки информации о товаре и применения модели машинного обучения были собраны в единую библиотеку.

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

Так же была разработана панель проверки результатов работы классификатора.

На изображении представлен скриншот веб панели, предназначенной для проверки результатов работы классификатора.

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.

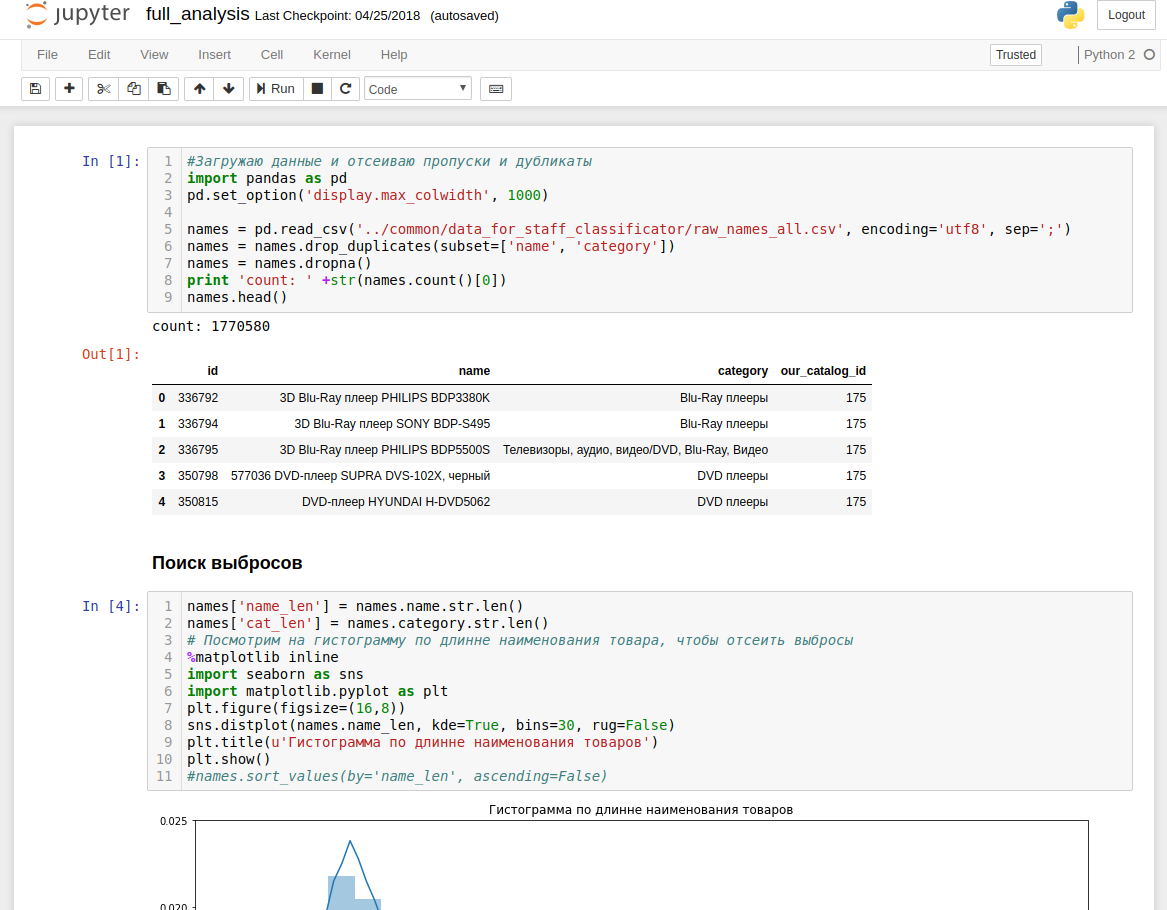


# 6. Использованные технологии

В качестве языка программирования использовался Python версии 2.7.

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 картинке представлен скриншот веб-приложения Jupyter notebook.



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

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.

Для составления биграмм использовал библиотеку nltk.

# 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 Visual Geometry Group 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 Keras François Chollet.
* 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 полученных признаках 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 предложен способ применения изображений товаров для их классификации.

# Библиографический список

1. Константин Коточигов - CRISP-DM: проверенная методология для Data Scientist-ов - <https://habr.com/company/lanit/blog/328858/>
2. Pandas 0.22.0 documentation - <http://pandas.pydata.org/pandas-docs/stable/index.html>
3. Seaborn documentation - <http://seaborn.pydata.org>
4. Documentation of scikit-learn 0.19.1 - <http://scikit-learn.org/stable/documentation.html>
5. Stochastic Gradient Descent - <http://scikit-learn.org/stable/modules/sgd.html>
6. Support Vector Machines - <http://scikit-learn.org/stable/modules/svm.html>
7. Introduction to Boosted Trees - <http://xgboost.readthedocs.io/en/latest/model.html>
8. Pavel Ostyakov - Our solution [5th place] - <https://www.kaggle.com/c/cdiscount-image-classification-challenge/discussion/45733>
9. Cdiscount’s Image Classification Challenge - <https://www.kaggle.com/c/cdiscount-image-classification-challenge>
10. Franc¸ois Chollet - Xception: Deep Learning with Depthwise Separable Convolutions - <https://arxiv.org/pdf/1610.02357.pdf>
11. Предварительно обученные нейронные сети в Keras - <https://www.asozykin.ru/deep_learning/2017/06/06/keras-pretrained-networks>