**Задача 05**

**«Поиск одинаковых**

**товаров на маркетплейсе»**

**Команда “Cats DS”**

Репозиторий git: <https://github.com/DimirSDV/DimirSDV_Cats>

Презентация: <https://drive.google.com/drive/folders/1dJAaA9Gls-1VhYN1Rpp8sp0f4L6Ft9Us?usp=sharing>

Документация: <https://drive.google.com/drive/folders/1dJAaA9Gls-1VhYN1Rpp8sp0f4L6Ft9Us?usp=sharing>

Сроки проекта: 18.05.2023 - 30.05.2023

|  |  |
| --- | --- |
| **Имя и фамилия** | **Роль (функции)** |
| Иван Черных | DS / проджект |
| Толстова Ольга | Аналитик / DS |
| Степан Кутькин | Аналитик / DS |
| Егор Геращенко | Аналитик / документация |
| Дмитрий Саханенко | Капитан / презентация |

Куратор команды от ЛЦТ: @j\_rovenskaya

## 

Оглавление

[Постановка задачи 3](#_Toc136375456)

[Описание набора данных 3](#_Toc136375457)

[Описание колонок из таблиц \*\_data.parquet: 3](#_Toc136375458)

[Файлы для скачивания: 4](#_Toc136375459)

[Формирование гипотез 4](#_Toc136375460)

[Анализ baseline 4](#_Toc136375461)

[Гипотезы улучшений: 5](#_Toc136375462)

[Предлагаемое решение 6](#_Toc136375463)

[Предобработка данных 6](#_Toc136375464)

[Словарь цветов 6](#_Toc136375465)

[Пропуски 6](#_Toc136375466)

[7](#_Toc136375467)

[Проверка дистанций на эмбеддингах картинок 7](#_Toc136375468)

[Генерация дополнительных параметров 7](#_Toc136375469)

[Машинное обучение 10](#_Toc136375470)

[Анализ точности модели 10](#_Toc136375471)

[Анализ влияния признаков 11](#_Toc136375472)

[11](#_Toc136375473)

[Описание используемых моделей 13](#_Toc136375474)

[Архитектура решения 17](#_Toc136375475)

[17](#_Toc136375476)

# Постановка задачи

*Разработать ML-модель, способную определить идентичность товаров по названиям, атрибутам и изображениям. Модель должна находить среди пар-кандидатов как можно больше одинаковых товаров с точностью >95%*

Ozon — ведущая мультикатегорийная платформа электронной коммерции и одна из крупнейших интернет-компаний в России. На площадке представлено более 150 млн товарных наименований в 20 категориях: от книг и одежды до продуктов питания и товаров для здоровья.

Сейчас более 90% ассортимента площадки формируют партнеры маркетплейса, в некоторых случаях предлагающие одинаковые товары по разной стоимости и с разными сроками доставки.

В этой связи Ozon нужно постоянно совершенствовать алгоритм определения одинаковых товаров, чтобы клиенты лучше ориентировались в предложениях продавцов

# Описание набора данных

* Тренировочная выборка: пары одинаковых и различных товаров
* Тестовая выборка: пары товаров без разметки (выборка для формирования лидерборда);
* Дополнительные данные: названия, атрибуты, векторные представления картинок (эмбединги) товаров

## Описание колонок из таблиц \*\_data.parquet:

* variantid (int) уникальный id товара
* name (string) название товара
* categories (string) категории товара. Превратить в dict можно с помощью json.loads`
* color\_parsed (array`) цвета, которые удалось получить из атрибутов товара и названия простым способом
* *embeddings* - векторные представления товаров. Получены с помощью resnet50 и Bert
* characteristic\_attributes\_mapping (string) атрибуты товаров. Превратить в dict можно с помощью json.loads

## Файлы для скачивания:

* train\_pairs.parquet - тренировочная выборка с таргетом
* train\_data.parquet - дополнительные данные для тренировочной выборки
* test\_pairs\_wo\_target.parquet - тестовая выборка для участников без таргета
* test\_data.parquet - дополнительные данные для тестовой выборки
* submission\_example.csv - сабмит на бейзлайн

# Формирование гипотез

## Анализ baseline

1. Делается попарный поиск расстояния каждого с каждым изображения, т.е. получается список из 40 расстояний (10\*4).

2. Из 40 расстояний считается процентиль. Т.е. то значение которое задает порог для отделения процента элементов.

3. Расчет делается для трех процентилей [0, 25, 50] и делаются три признака с этими расстояниями, эти признаки-фичи говорят о схожести двух наборов картинок (даже если кол-во разное в наборе)

В качестве развития идеи предлагается тестирование разных методик расчета расстояний (например, стандартные из scikit-learn и из scipy.spatial.distance). Также возникла идея добавить коэффициент сходства Жаккара Н(IoU) т.к. там принципиально идея другая. Вообще IoU для этой задачи по идеи должен хорошо себя показать.

Также стоит учесть, что в изначальной функции, если у одного товара нет дополнительных картинок, то указывается «-1» как расстояние между товарами по картинкам — в таком случае можно брать основную картинку

Кол-во уникальных категорий всего в товарах:

В категории 1: 1

В категории 2: 13

В категории 3: 127

В категории 4: 357

Кол-во уникальных категорий в парах товар1-товар2:

В категории 1: 1

В категории 2: 3

В категории 3: 78

В категории 4: 271

Организаторы предлагают использовать только важные категории. Редкие категории покрывают 10к из 300к товаров, что является нормальным допущением. Такие редкие категории помечаются одной общей — создается категория «прочее» и все редкие товару переходят туда. Таким образом у организаторов получилось из 78 категории 3 уровня сделать всего 40 категорий, что лучше, но следует углубиться в это направление

Всего похожих товаров (таргет=1): 135013

Схожих товаров с одинаковыми категориями:

С одинаковыми 1 категориями: 135013

С одинаковыми 2 категориями: 134800 — т.е. есть товары которые одинаковые, но у них 2ая категория различная таких товаров 213 и их надо изучить!

С одинаковыми 3 категориями: 132228 — т.е. есть одинаковые товары, но у них 3ая категория различная — 2785

С одинаковыми 3 категориями: 127350 — т.е. есть одинаковые товары, но у них 3ая категория различная — 7663

Туда попадаются такие товары как Батарейка «AIRLINE» = Моторное масло Airline — т.е. явное противоречие, но есть и Электронный конструктор «Солнцеход», который там и там.

## Гипотезы улучшений:

— Учитывать схожесть двух товаров по 4ой категории (1 столбец). Но товары могут находится в рядом расположенных категориях, например мобильные телефоны и смартфоны. Для этого можно вручную сделать словарь соответствия категорий и в виде фичей будет добавлено только 1 поле указывающее на схожесть категорий. Т.е. будет словарь который говорит что категория 18 похожа на 18, 34 и 53. и когда будет сравнить два товара из категорий 18 и 53, то в фиче-поле «схожесть 4ой категории» ставим 1.

— Аналогично для 3ей категории тоже можно сделать «похожесть» разных категорий, если требуется — нужен анализ как это сделать

— Словесные названий категорий (NLP) нужно понять как оно сочетается с названиями товаров. Если эмбеддинги текста это и есть названия товаров, тогда можно просто их использовать

— Изучить тему уменьшения категорий, какие еще есть методы

— Чистить неверную разметку: Надо придумать правило, для которого можно заложить предварительную обработку таких данных, чтобы почистить неверные данные по классам. Например, разные категории и большое расстояние названий категорий, тогда будет считаться ошибочным назначение схожести в обучении.

— Чистить неверную разметку, если таргет указан 0, но товары одинаковые, при этом категории идентичны и эмбединги названий очень похожи

# Предлагаемое решение

## Предобработка данных

* Объединение атрибутов связанных товаров
* Расчет расстяоний между эмбеддингами изображений и названий
* Кодирование категории цвет
* Сравнение схожести категорий для одинаковых товаров
* Разбивка товаров по третьей категории, для метрики
* Характеристика товаров

## Словарь цветов

У некоторых товаров может быть несколько цветов, поэтому эта категория представлена в виде массива массивов. У товара может быть черный и зеленый цвет. Сделана кодировка в OHE. Уникальных названий цветов 247 шт. Итоговый словарь цветов выглядит так: {1: «„красный“», 2:""оранжевый«".....} и т.д.

Также сделан словарь соответствий названий цветов каждому уникальному слову обозначающего цвет. Например, цвет «грушевый» или «lime» — можно отнести и к желтому, и к зеленому, и к салатовому, а ’красновато-коричневый’ - к коричневому и красному, «молочный» и «snow» — к белому и т.д.

Вид словаря:

{«„lime“»: [2,5,7],

«„грушевый“»: [2,5,7],

«„snow“»: [1],

«„красновато-коричневый“»: [3,6],

...

}

Будет проведен тест разных версий словарей цветов: лояльный, средний, строгий.

## Пропуски

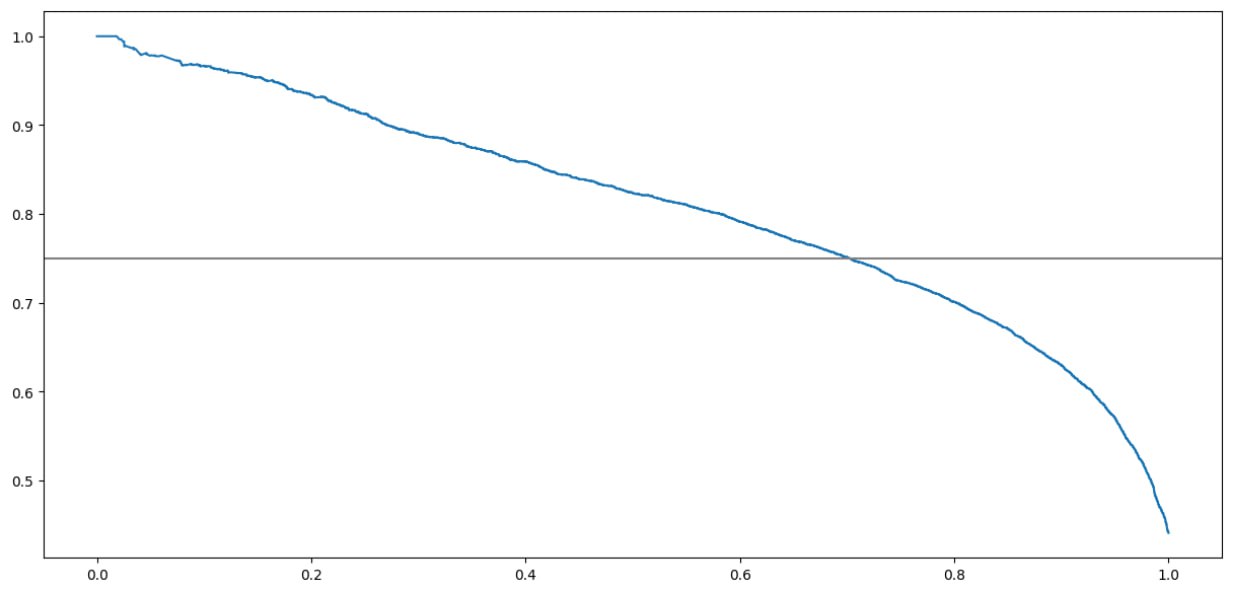
Пропуски картинок в pics заполнены главной картинкой. Размерность у изображений одинаковая - 128.

Выполнена проверка метрик классификации. Оптимальный результат показала метрика PRAUC

**Распределение PR-ROC-кривой** в baseline

## 

Распределение **PR-ROC-кривой** после предобработки данных: работа с цветами, расстояниями и схожестью названий



## Проверка дистанций на эмбеддингах картинок

Для эмбеддингов подошли [’braycurtis’, ’canberra’, ’chebyshev’, ’correlation’, ’hamming’, ’minkowski’, ’sqeuclidean’]

## Генерация дополнительных параметров

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Название** | **code** | **Описание** | **Результат применения** |
| 1 | OHE цвета товара 1 и 2 | color1\_n, color2\_n (n: 1-11) | Цвета разложенные на простые и закодированные в OHE | Есть прирост 3% (0.03) |
| 2 | Коэффициент схожести наименований | iou\_names | Наименования часто содержат коды моделей и основные атрибуты товаров. Предлагается сделать простую разбивку наименований и сравнить IoU | Есть прирост 3% (0.03) стала самым важным признаком |
| 3 | Коэффициент схожести наименований, с убранными спецсимволами | iou\_names\_2 | Наименования часто содержат коды моделей и основные атрибуты товаров. Предлагается сделать простую разбивку наименований и сравнить IoU | Есть прирост 2% (0.02), |
| 4 | Совпадение характеристик и Совпадение словарей характеристик  и  Коэффициент совпадения и самих словарей и характеристик внутри словарей | iou\_attr  iou\_dict  iou\_attr\_mat\_dict | Совпадение списка похожих словарей немного больше влияет, чем совпадения самих значений. Возможно это из-за совпадений товаров из разных категорий  iou\_attr\_mat\_dict равен перемножение iou\_attr на iou\_dict | Есть прирост 4,8% |
| 5 | Взвешенное совпадение характеристик | iou\_attr\_weight | Если две характеристики не совпадают это еще не факт что точно не равны может быть это описательная характеристика и они равны не точь в точь а например на 70%. Поэтому применяется IoU в качестве сравнения для всех характеристик. Получается такое IoU над коэффициентами подсчитывании через IoU | Есть прирост 1,7% |
| 6 | Сходство TOP характеристик | TOP\_ch\_n (n: 1- 20) | У каждой категории свои характеристики товаров, но можно выделить топ N-характеристик для каждой категории и в обучение подать OHE из топ-10 характеристик для каждой категории, а также признак сходства и IoU  TODO:  - признак сходства и IoU  - Требуется также отдельно посчитать для rest  - расширить на 4ую категорию | Есть прирост 4,8% |
| 7 | Определение важности категорий внутри каждой группы | df\_idf | Если сама характеристика встречается у большинства товаров, значит это важная характеристика, которая может разделить множество товаров. Но при этом, если значение характеристики у большинства товаров одинаковое (например, почти у всех товаров "тип вилки" будет "евровилка" ), то это менее значимая характеристика.  Таким образом из набора характеристик будут использоваться только важные |  |
| 8 | Коэффициент различий между названиями товаров с учетом нахождения в различающихся слов в списке «АнтиСлов» | anti\_words\_values | Коэффициент различий между названиями товаров с учетом нахождения в различающихся слов в списке «АнтиСлов».  Например, видеокарта GeForce ABC и GeForce ABC Ti и эти товары помечены как разные (таргет = 0). Так вот проанализировав все пары с таргетом «0» были сформированы наиболее частые слова которые различаются в названиях различных товаров. | Есть прирост 1,4% |
| 9 | Сравнение персечений N-gram в названиях товаров | iou\_2gram\_name, iou\_3gram\_name | iou (жаккара) сравнить количество одинаковых биграм (триграм) в двух названиях с общим количеством биграмм (триграм). И аналогично сделать и для характеристик | Есть прирост 0,9% |

# 

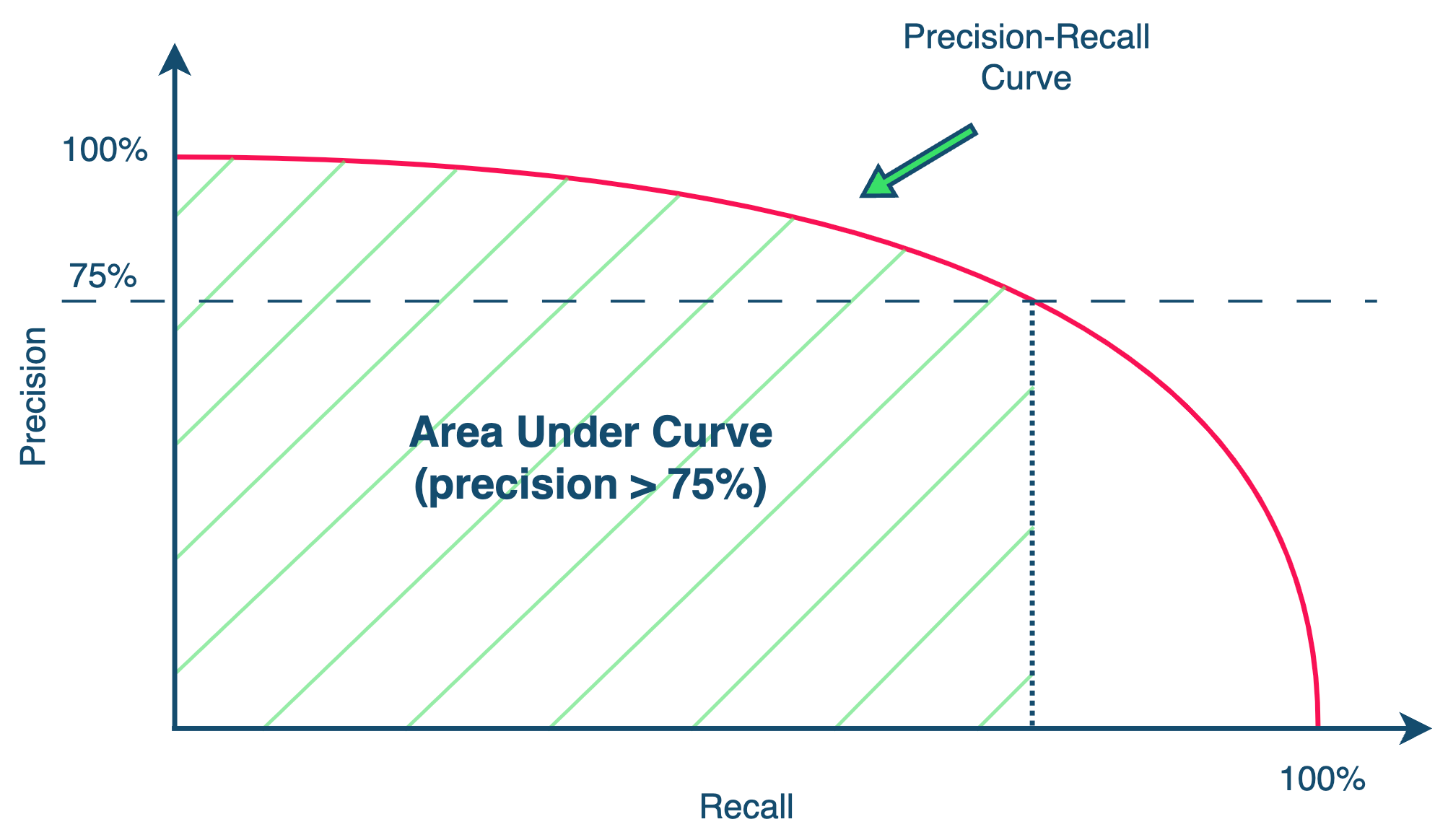
# Машинное обучение

## Анализ точности модели

Метрики

Задача

Разработайте ML-модель, способную определить идентичность товаров по названиям, атрибутам и изображениям. Модель должна находить среди пар-кандидатов как можно больше одинаковых товаров с точностью >75%



Метрика

Precision-Recall-AUC (Precision = 75%) усредненный по крупным категориям.

PRAUC (Precision = 75%) – это площадь под графиком Precision-Recall Curve в зоне, где Precision > 0.75.

Всего у нас есть 4 уровня категорий. 1 уровень — самый общий (EPG, Apparel и другие). 2 — более глубокий (Автотовары, Одежда, Электроника и другие). В рамках соревнования рассматриваем только категорию Электроника и её подкатегории (категории 3 и 4 уровня). Примеры категорий 3 уровня в Электронике: Компьютер, Смарт-часы, Смартфоны и другие. Так как рассматриваем только одну категорию 2 уровня, усреднять метрику будем по категориям 3 уровня. Крупные категории 3 уровня для обучающей выборки можно получить следующим образом:

train["cat3"] = train["categories"].apply(lambda x: json.loads(x)["3"])

train["cat3\_grouped"] = train["cat3"].apply(lambda x: x if cat3\_counts[x] > 1000

else "rest")

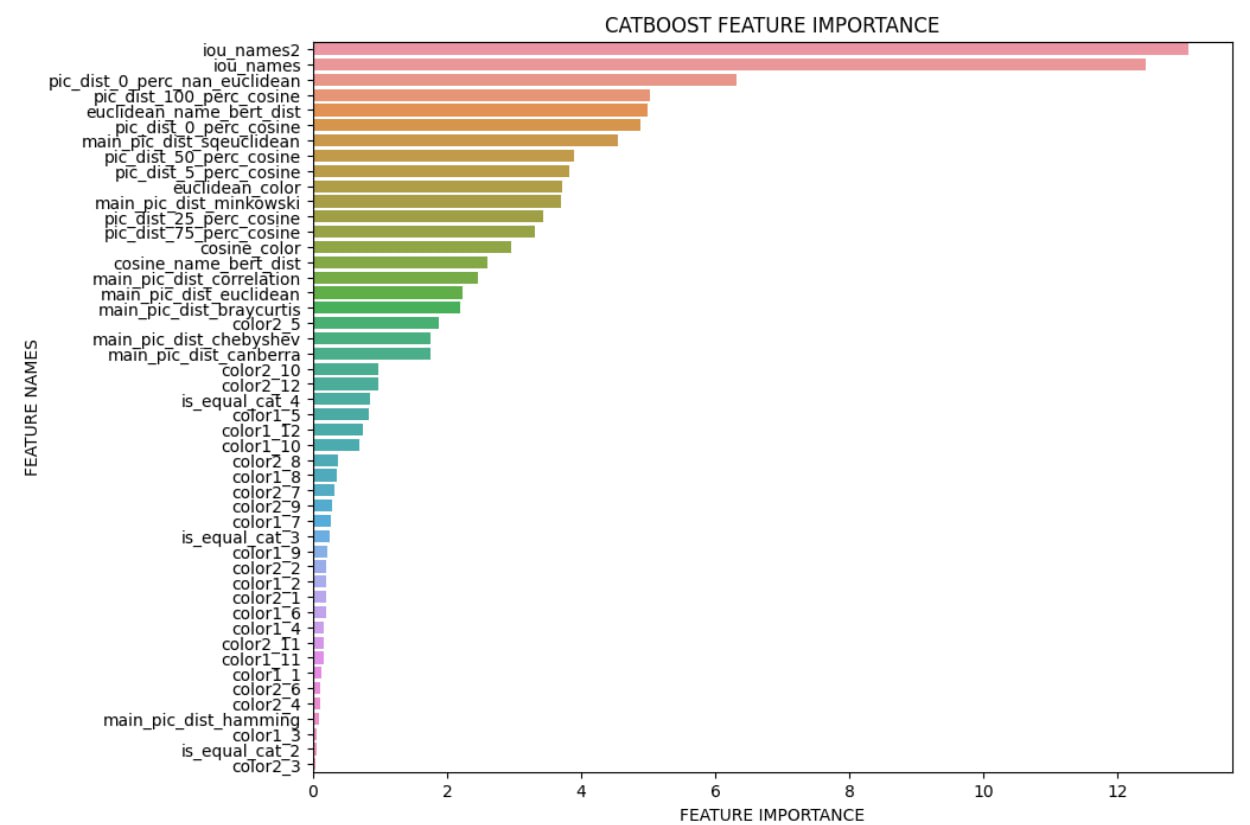
categories — можно достать из train\_data.parquet и test\_data.parquet. cat3\_grouped — большие категории 3 уровня, по которым будет усредняться метрика (по сути, сливаем все категории с менее чем 1к примерами в обучающей выборке в одну категорию). Тестовую выборку с категориями можно скачать с платформы DS Works, на которую предстоит загружать решения.

## Анализ влияния признаков

Важен выбор методов расчета дистанции. Евклидово расстояние показало хороший результат для эмбеддинов, но еще лучше результат получен при добавлении squclidean или minkowski

## 

IoU это метрика на основе расстояния Жаккара называется IoU (Intersection over Union – Пересечение над Объединением).



*import re*

*def text\_title\_iou(name1, name2):*

*words1 = set(name1.split())*

*words2 = set(name2.split())*

*iou = len(words1 & words2)/ len(words1 | words2)*

*return iou*

*def text\_title\_iou\_filter(name1, name2):*

*words1 = set(re.findall( r'([\d\w]+)', name1))*

*words2 = set(re.findall( r'([\d\w]+)', name2))*

*iou = len(words1 & words2)/ len(words1 | words2)*

*return iou*

Во второй версии была применена разбивка не по словам, а с помощью вытаскивания подряд идущих символов и цифр через регулярные выражения.

Финальная версия с учетом всех наработок выглядит таким образом:

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Автоматически созданное описание

# Описание используемых моделей

|  |  |
| --- | --- |
| Название | Описание |
| Базовая модель с заполненными пропусками изображений | У товаров которых нет доп картинок, используются их главные картинки |
| Модель №2 + добавлена категоризация цветов | Основа модель №2. Добавлена категоризация цветов. С учетом разложения сложных цветов на более простые (выбрано 11 простых цветов) |
| Модель №3 + добавлены признаки равенства категорий 2,3,4 | Модель №3 + добавлены бинарные признаки равенства категорий 2,3,4 между товарами товаров |
| Модель №4 + поправлен расчет группировки 3 категории и добавлена группировка по 4ой категории | Модель №4 + поправлен расчет группировки 3 категории и добавлена группировка по 4ой категории |
| Модель №5 + Коэффициент схожести наименований | Наименования часто содержат коды моделей и основные атрибуты товаров. Предлагается сделать простой сплит наименований и сравнить IoU |
| Модель №6 + Исправленный коэффициент схожести наименований | Изменен коэффициент IoU, теперь деление слов не по сплиту а через регулярные выражения. Учитываются только цифры и символы |
| Модель №7 + Измененный словарь цветов | Модель №7 + Измененный словарь цветов. Более строгое цветовое соответствие |
| Модель №7 + сравнение векторов цветов через евклидово и cos расстояния | Модель №7 + сравнение векторов цветов через евклидово и cos расстояния |
| Модель №7 + исправленное сравнение векторов цветов через евклидово и cos расстояния | Модель №7 + сравнение векторов цветов через евклидово и cos расстояния |
| Модель №7 + исправленное сравнение векторов цветов через евклидово и cos расстояния | Модель №7 + сравнение векторов цветов через евклидово и cos расстояния. **Строгий** словарь цветов |
| Модель №7 + исправленное сравнение векторов цветов через евклидово и cos расстояния | Модель №7 + сравнение векторов цветов через евклидово и cos расстояния. **Лояльный** словарь цветов |
| Модель №11 + добавление процентилей 75 и 95 | Модель №11 + добавление процентилей 75 и 95 для дополнительных картинок |
| Модель №13 + добавление дополнительные расстояние для проценталей | Модель №13 + добавление дополнительные расстояние для проценталей  cosine + manhattan + nan\_euclidean |
| Модель №13 + чистка лишних дистанций | Много одинаковых параметров ухудшают модель, надо оставить только важные дистанции |
| Модель №15 + добавление расстояний между главными картинками | Проверялись три: косинусное, евклидово, неевклидово и манхеттенское .  Лучше сработало евклидово расстояние, остальные надо убрать |
| Модель №16 + проверка расстояния между главными картинками другими методами | Модель №16 + проверка других расстояний из ['braycurtis', 'canberra', 'chebyshev', 'correlation', 'hamming', 'minkowski', 'sqeuclidean']: |
| Модель №17 + проверка расстояния между доп картинками другими методами | Применение методов расчета расстояния ["sqeuclidean", "minkowski"] не только на главные но и на доп картинки |
| Модель №18 + чистка лишних дистанций | Много одинаковых параметров ухудшают модель, надо оставить только важные дистанции |
| Модель №19 + проверка новых методов расчета расстояния на текстовых эмбедингах | Применение методов расчета расстояния ["sqeuclidean", "minkowski"] не только на картинки но и на эмбеддинги текста |
| NLP Модель №21 на основе сравнения названий товаров от Степана | NLP Модель №21 на основе сравнения названий товаров |
| Модель №20 + IoU атрибутов | Сделан первичный разбор характеристик товаров  "characteristic\_attributes\_mapping1" и первым делом подсчитан iou |
| Модель №22 + расширинеы метрики аттрибутов "iou\_attr", "iou\_dict", "iou\_attr\_mat\_dict" | "iou\_attr" - совпадение характеристик  "iou\_dict" - совпадение словарей характеристик  "iou\_attr\_mat\_dict" - перемножение iou\_attr на iou\_dict |
| Модель №23 + взвешенный iou\_attr | Модель №23 + взвешенный iou\_attr\_weight |
| Модель №24 + чистка лишних дистанций | Проверялся threshold: apply(lambda x: 1 if x > 0.5 else x)  0.00771 |
| Модель №25 + 10 TOP характеристик | Идея такая: у каждой категории свои характеристики товаров, но можно выделить топ N характеристики для каждой категории и в обучение подать OHE из топа 10 характеристик для каждой категории |
| Аналог модели №26 но 5 TOP характеристик | OHE из топа 5 характеристик для каждой категории |
| Аналог модели №26 но 15 TOP характеристик | OHE из топа 15 характеристик для каждой категории |
| Модель №28 + Развернуть расстояние в близость добавить расстояние цветов исключить отдельные цвета как признаки | Добавлено расстояние цветов, чтобы избавится от отдельных параметров colors  Близость = (1 - расстояние) |
| Модель №28 + Скорректированы расстояния цветов | Стабильная версия |
| Модель №30 + еще один заход схлопнуть цвета хотя бы оставить бинарное сравнение вместо полных двух списков | Все-таки простые цвета как признак дают некоторый скор, может позже получится агрегацию оставить только |
| Модель №30 + iou\_color | Для группы данных которые по сути не являются вектором, т.е. не задают направление для них плохо работает расстояние.  И для них по сути надо делать сверку множеств. Еще один заход на IoU цвета |
| Модель №32 + 15/20 TOP характеристик с взвешенной разницей | ТОП характеристик не просто совпадает или нет 0/1, а если не совпадает то на сколько через IoU |
| Для каждой категории уровня 3 своя модель | Это сделано с идеей что у каждой категории свои особенности "схожести" товаров |
| Модель №35 + выделение ТОП 30/10/20/40 важных признаков DF IDF | Идея что выбираем не просто какие то ТОП признаков по которым смотрим совпадение а выбираем важные признаки с помощью DF IDF. |
| Модель №39 добавлен перевод в нижний регистр | Добавлен перевод в нижний регистр |
| Модель №40 + Сэмплирование данных | Сэмплирование данных для устранения дисбаланса производилось с помощью ADASYN |
| Модель №40 + Взвешивание классов через параметры Катбуста | Вместо сэмплирования, которое надо очень аккуратно делать, была попытка взвесить классы при обучении |
| Модель №40 + RandomOverSampler + TomekLinks + ручные веса + подбор гиперпараметров + сэмплирование только подвырки трейн без вала и без теста | RandomOverSampler + TomekLinks + ручные веса + подбор гиперпараметров + сэмплирование только подвырки трейн без вала и без теста |
| Модель №40 + TomekLinks + Взвешивание классов через параметры Катбуста + Over sampler | Модель №40 + TomekLinks + Взвешивание классов через параметры Катбуста + SMOTE |
| Модель №50 + anti\_words\_values ТОП 100 Антислов | Коэффициент различий между названиями товаров с учетом нахождения в различающихся слов в списке "АнтиСлов".  Есть например видеокарта GeForce ABC и GeForce ABC Ti и эти товары помечены как разные (таргет = 0). Так вот проанализировав все пары с таргетом =0 были сформированы наиболее частые слова, которые различаются в названиях различных товаров |
| Модель №50 + anti\_words\_values ТОП 200 Антислов | Сравнивается с ТОП 200 Антислов |
| Модель №52 + Ансамбль | Ансамбль:  BalancedBaggingClassifier  EasyEnsembleClassifier  LGBMClassifier  CatBoostClassifier  GradientBoostingClassifier  HistGradientBoostingClassifier  RandomForestClassifier  В качестве стекинг модели VotingClassifier |
| Модель №52 + Ансамбль | Ансамбль:  BalancedBaggingClassifier  EasyEnsembleClassifier  LGBMClassifier  CatBoostClassifier  GradientBoostingClassifier  HistGradientBoostingClassifier  RandomForestClassifier  В качестве стекинг модели VotingClassifier |
| Модель №54 + N-gram (биграммы и триграммы) | Сравнение пересечений (iou / жаккара) количество одинаковых биграмм (триграмм) в двух названиях с общим количеством биграмм (триграмм) |
| Модель №55 + изменен Ансамбль | Изменен ансамбль |
| Модель №56 + новые данные | Генерация данных должна быть аккуратной, валидацонную выборку надо определять до генерации  Надо верхнее ограничение поставить. Подтюнить и разобраться с генерацией Увеличить число групп  Попробовать уменьшить данные в группах с низкими минорными классами  ундер-самплинг  В метрике вес изменить, сейчас считается не очень верно, из-за того что кол-во данных было изменнно |
| Модель №56 + новые данные c верхним ограничением | 1. Валидацонную выборку надо определять до генерации 2. Заменить генерацию отрицательных примеров |
| Модель №56 + новые данные только для класса =1 | Надо:  Негативные примеры будут генрирроваться по более простому правилу, будут браться товары из разных 4х категорий |
| Модель №56 + новые данные только для класса =1 с контролем дублирования пар товаров | Надо разбираться почему падает скор Времени нет, отказываемся пока от этой идеи возвращаемся к модели 56 |
| Модель №61 + N-gram для характеристик | Была ошибка, значения N-gram обнулялись |
| Модель №62 + Уменьшение цветов | см. в названии модели |
| Модель №62 + новые N-грамы + корректировка обраотчиков текста | см. в названии модели |
| Модель №64 + Вернули признакми по цветам | см. в названии модели |
| Модель №65 + предикт от общей модели | см. в названии модели |
| Модель №66 + N-gram (2,3,4,5) в характеристиках товаров | см. в названии модели |
| Модель №67 + больше групп с 24 до 30 + времено исключена общая модель Catboost | см. в названии модели |
| Модель №68 + добавлены еще расстояния в названия и в характеристики с отдельным учетом не слов а кодов обозначений | см. в названии модели |
| Модель №69 + сравнение вектора важных числовых признаков | Скорее всего уже влияет кол-во признаков, надо уменьшать |
| Модель №70 + оставлена только две дистанции | Исключаем признаки |
| Модель №69 + чистка | Чистка модели |
| Модель №72 + включена общая модель Catboost | Вывод: общая модель Catboost все портит |

## Архитектура решения

## 