



Контентные рекомендации

Андрей Зимовнов (Яндекс, ВШЭ)



Зачем content-based?

- **Похожести товаров**
 - Поступил новый товар → пока не знаем кому он нравится, но знаем его описание → можем считать похожести описания → решаем проблему холодного старта для товара
- **Похожести юзер-товар**
 - Хотим учитывать предпочтения юзера к режиссеру, словам описания, картинке, ...
- **Похожести юзеров**
 - Например: любители вестернов и Иствуда 😊



Зачем content-based?

- **Похожести товаров**
 - Можно использовать в CF
- **Похожести юзер-товар**
 - Готовый признак
- **Похожести юзеров**
 - Можно использовать в CF

Какой бывает контент



Название	
Режиссер	
Год	
Актер	
Жанр	
Описание	

- Структурированный (характеристики)
- Текстовый (описание)
- Медиа (аудио, видео, картинки)



Структурированный

Самый простой тип контента!

Товар-товар: совпадение или близость характеристик

- одинаковый бренд
- похожая диагональ
- ...



Структурированный

Самый простой тип контента!

Юзер-товар: предпочтение характеристики юзером

- Частота жанра в истории юзера
- Средняя оценка фильмов этого жанра в истории юзера
- ...



Структурированный

Самый простой тип контента!

Юзер-юзер: совпадение профилей юзеров

- Смотрят в одинаковых пропорциях жанры
- Или любят тех же актеров
- ...



Пример

	Комедия	Боевик	Вестерн	Триллер
Юзер 1	0.3	0.1	0.2	0.4
Юзер 2	0.1	0.2	0.3	0.4
Фильм 1	1	0	0	0
Фильм 2	0	0	0	1

Как измерить похожесть юзер-товар?



Пример

	Комедия	Боевик	Вестерн	Триллер
Юзер 1	0.3	0.1	0.2	0.4
Юзер 2	0.1	0.2	0.3	0.4
Фильм 1	1	0	0	0
Фильм 2	0	0	0	1

Как измерить похожесть юзер-товар?

Косинус! Как и для других похожестей!



Пример

	Комедия	Боевик	Вестерн	Триллер
Юзер 1	0.3	0.1	0.2	0.4
Юзер 2	0.1	0.2	0.3	0.4
Фильм 1	1	0	0	0
Фильм 2	0	0	0	1

Как измерить похожесть юзер-товар?

Косинус! Как и для других похожестей!

$$a \cdot b = ||a|| \cdot ||b|| \cos \theta$$



$$\cos \theta = \frac{a \cdot b}{||a|| \cdot ||b||}$$

Пример: топики (темы)

Любит / не любит

Темы	Document	User	
политика	0	1	0
экономика	1	1	1
шоппинг	1	-1	-1
IT	0	1	0
футбол	0	1	0
романтика	1	-1	-1
семья	1	-1	-1
природа	0	0	0
юмор	0	0	0
Россия	1	1	1

$$\cos \theta = ?$$

$$\|b\| = ?$$

$$\|a\| = ?$$

$$a \cdot b = ?$$

Пример: топики (темы)

Любит / не любит

Темы	Document	User	
политика	0	1	0
экономика	1	1	1
шоппинг	1	-1	-1
IT	0	1	0
футбол	0	1	0
романтика	1	-1	-1
семья	1	-1	-1
природа	0	0	0
юмор	0	0	0
Россия	1	1	1

$$\cos \theta = -0.16$$

$$\|b\| = \sqrt{5}$$

$$\|a\| = \sqrt{8}$$

$$a \cdot b = -1$$

Пример: one-hot кодирование



бренд	Desigual, Concept Club, ...
цвет	желтое, зеленое, красное, ...
фасон	вечернее, повседневное, ...
страна	Россия, Италия, Франция, ...
сезон	демисезон, лето, зима, ...



Пример: one-hot кодирование



Desigual	0
Concept Club	1
желтое	1
зеленое	0
красное	0
вечернее	0
повседневное	1
Россия	1
Италия	0
Франция	0
демисезон	1
лето	0
зима	0



Текстовый контент



Мешок слов

[illegible]



Хэширование слов

Храним “word” → index

- Словарь должен быть общим для всех машин
- Может не поместиться в RAM

Считаем “word” → hash(“word”)

- Большое число корзинок хэш-функции ($\sim 2^{24}$), качество растёт по $\log(\text{hash bits})$
- Значительно быстрее, не занимает память и легко распараллеливается

Пример хэш-функции

$$\phi(\text{good}) = 0$$

$$\phi(\text{movie}) = 1$$

$$\phi(\text{not}) = 2$$

$$\phi(\text{a}) = 3$$

$$\phi(\text{did}) = 3$$

$$\phi(\text{like}) = 4$$

$$\text{hash}(s) = s[0] + s[1]p^1 + \dots + s[n]p^n$$

s – строка

p – простое число

$s[i]$ – код символа

коллизии!

good movie
not a good movie
did not like



0	1	2	3	4
1	1	0	0	0
1	1	1	1	0
0	0	1	1	1

*HashingTF
в sklearn!*



Чуть умнее: TF-IDF

Частые слова менее важны: они есть везде, по ним все будут похожи

Редкие слова более важны: какая-то узкая тематика

Но не слишком редкие: опечатки 😊

Term Frequency

частота термина t
в документе d

$$TF_{t,d} = \frac{f_{td}}{\max_z f_{zd}}$$

максимальная частота
по всем терминам z
в документе d

Term Frequency

Variants of TF weight

weighting scheme	TF weight
binary	$\{0,1\}$
raw frequency	$f_{t,d}$
log normalization	$\log(1 + f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \frac{f_{t,d}}{\max f_{t,d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max f_{t,d}}$

Inverse Document Frequency

$$IDF_t = \log \frac{N}{df_t}$$

число документов
в корпусе

число документов
с термином t

Inverse Document Frequency

Variants of IDF weight

weighting scheme	IDF weight
unary	1
inverse frequency	$\log \frac{N}{n_t}$
inverse frequency smooth	$\log(1 + \frac{N}{n_t})$
inverse frequency max	$\log \left(1 + \frac{\max_t n_t}{n_t} \right)$
probabilistic inverse frequency	$\log \frac{N - n_t}{n_t}$

$$TF_{t,d} * IDF_t = \frac{f_{td}}{\max_z f_{zd}} \log \frac{N}{df_t}$$

TF-IDF в sklearn

```
from sklearn.feature_extraction.text import TfidfVectorizer
import pandas as pd
texts = [
    "good movie", "not a good movie", "did not like",
    "i like it", "good one"
]
tfidf = TfidfVectorizer(min_df=2, max_df=0.5, ngram_range=(1, 2))
features = tfidf.fit_transform(texts)
pd.DataFrame(
    features.todense(),
    columns=tfidf.get_feature_names()
)
```

	good movie	like	movie	not
0	0.707107	0.000000	0.707107	0.000000
1	0.577350	0.000000	0.577350	0.577350
2	0.000000	0.707107	0.000000	0.707107
3	0.000000	1.000000	0.000000	0.000000
4	0.000000	0.000000	0.000000	0.000000

Чуть умнее: TF-IDF

TFIDF Normed Vectors	a	Accelerating	and	applications	art	behavior	Building	Consumer	CRM	customer	data	for	Handbook	Introduction	Knowledge	Management	Marketing	Mastering	mining	of	relationship	Research	science	technology	the	to	using	website	your
Building data mining applications for CRM				0.502			0.502		0.344		0.251	0.502							0.251										
Accelerating customer relationships: using CRM and relationship technologies		0.432	0.296						0.296	0.216											0.468			0.432			0.432		
Mastering Data Mining: the art and science of Customer Relationship Management			0.256		0.374					0.187	0.187					0.256		0.374	0.187	0.374	0.256		0.374		0.374				
Data Mining your website											0.316								0.316									0.632	0.632
Introduction to Marketing Consumer behavior						0.707		0.707						0.636			0.436									0.636			
Marketing Research: a Handbook	0.537												0.537				0.368					0.537							
Customer Knowledge Management										0.381					0.736	0.522													

Что с этим делать дальше?

Что с этим делать дальше?



Текстовая похожесть

- Косинусная похожесть неплохо работает на мешках слов!
- Можем находить похожие по описанию товары!



Текстовая похожесть

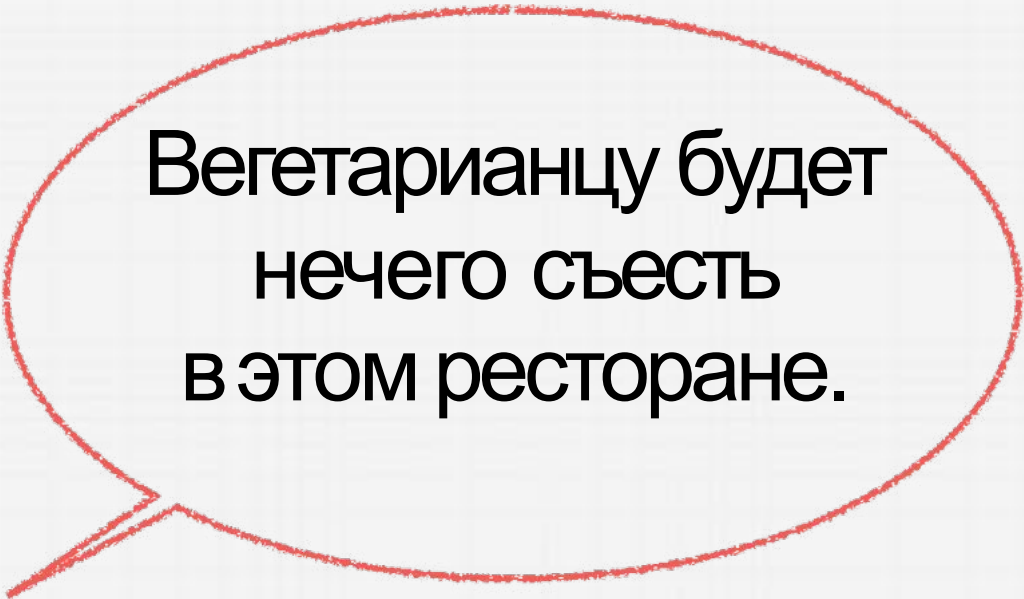
Плюсы:

- Решает проблему холодного старта для нового товара!
- Может рекомендовать даже непопулярные товары
- Интерпретируемые похожести (мешок слов)

Минус: теряется смысл



вегетарианец
есть
ресторан

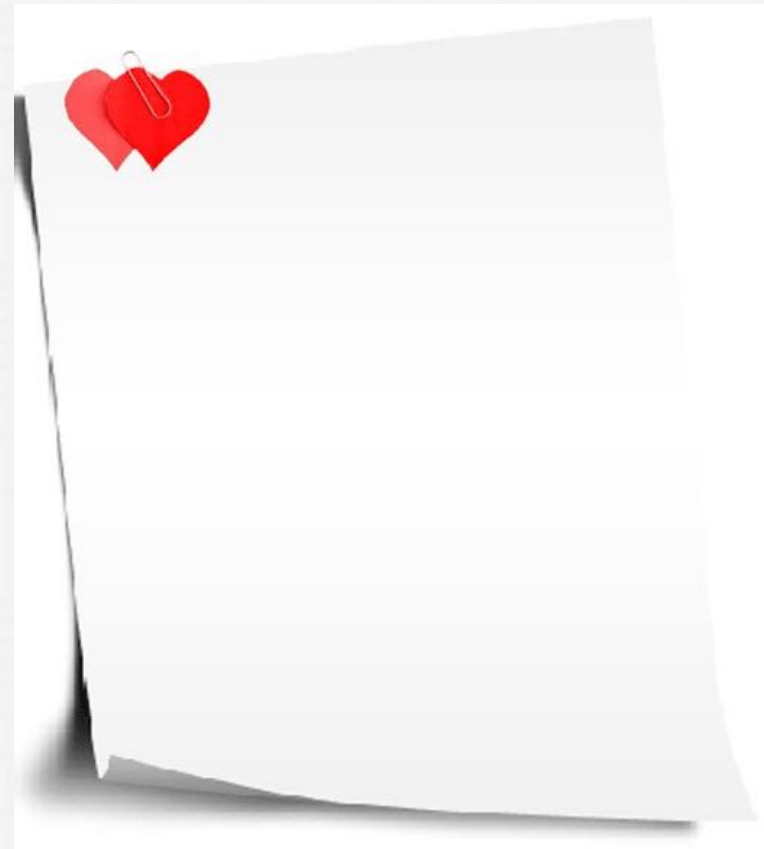


Вегетарианцу будет
нечего съесть
в этом ресторане.

Минус: не учитывает синонимы

казистый
хороший пышный видный
нарядный смазливый
щегольской изящный
пленительный благовидный
разубранный обворожительный
разукрашенный благообразный чудный
благотворный восхитительный художественный пригожий
привлекательный миловидный распрекрасный
блестящий бесподобный хорошенький
великолепный взрачный божественный
роскошный прелестный
живописный картинный
дивный

Минус: многозначные слова



Минус: многозначные слова





Минус: застреваем в пузыре

Контентной модели сложно удивить пользователя



Текстовые классификаторы

- Бывает полезно иметь текстовые классификаторы.
- Например, по тексту понимать его тематику (например, политика).
- Так можем обобщить предпочтения юзера от слов до тем!

$$c = \arg \max_c P(C | O)$$

строка текста

классы

Наивный Байес

вероятность
такого текста
в классе C

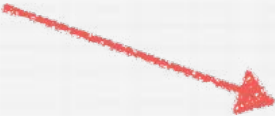
вероятность
класса C

$$P(C | O) = \frac{P(O | C)P(C)}{P(O)}$$

вероятность
текста O

Наивный Байес

вероятность
такого текста
в классе C

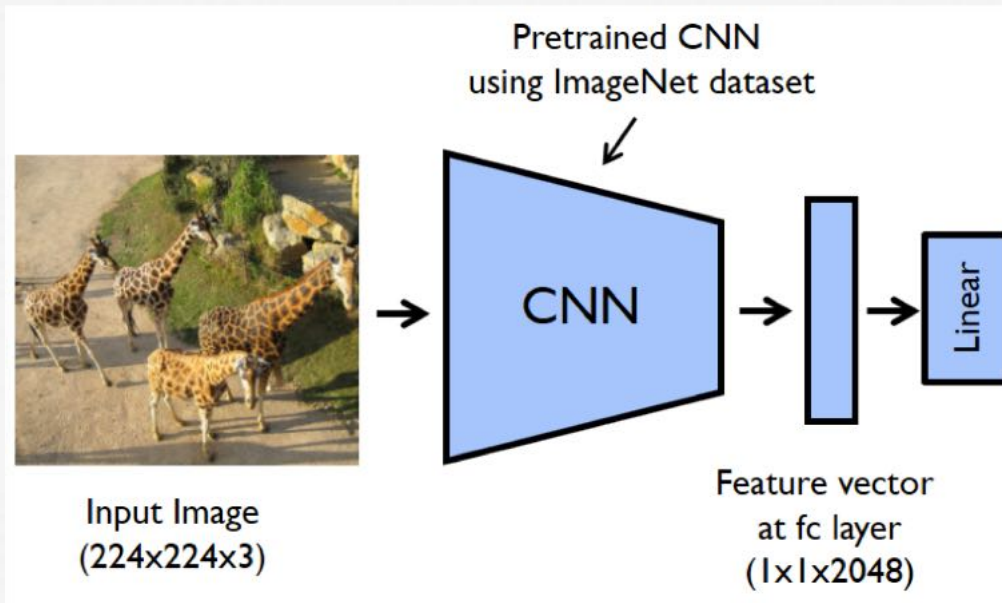

$$P(O | C) = P(o_1 o_2 \dots o_n | C)$$

Наивное предположение о независимости наличия слов

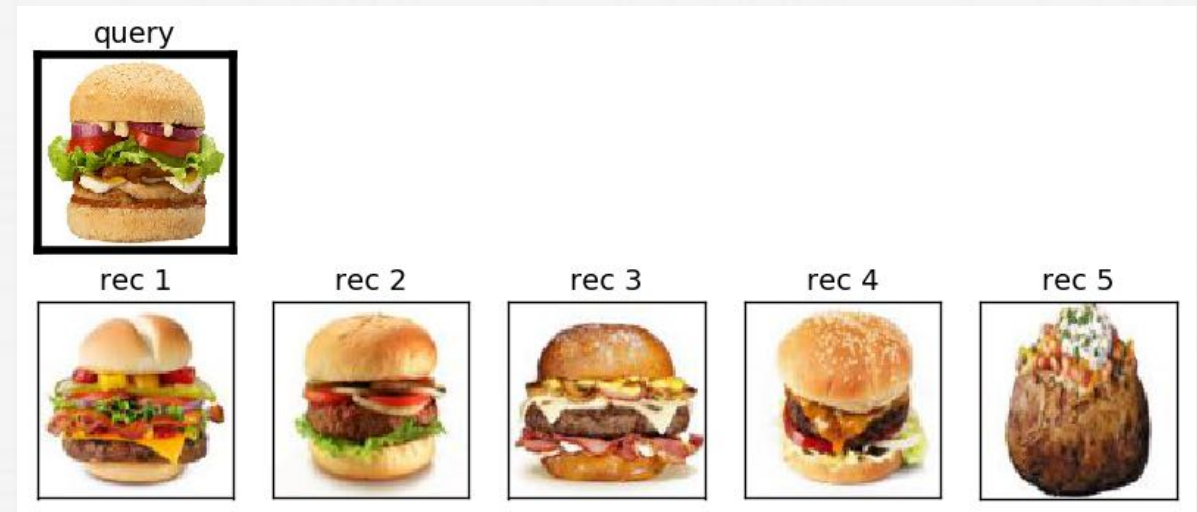
Наивный Байес



$$P(o_1 | C), P(o_2 | C), \dots, P(o_n | C)$$



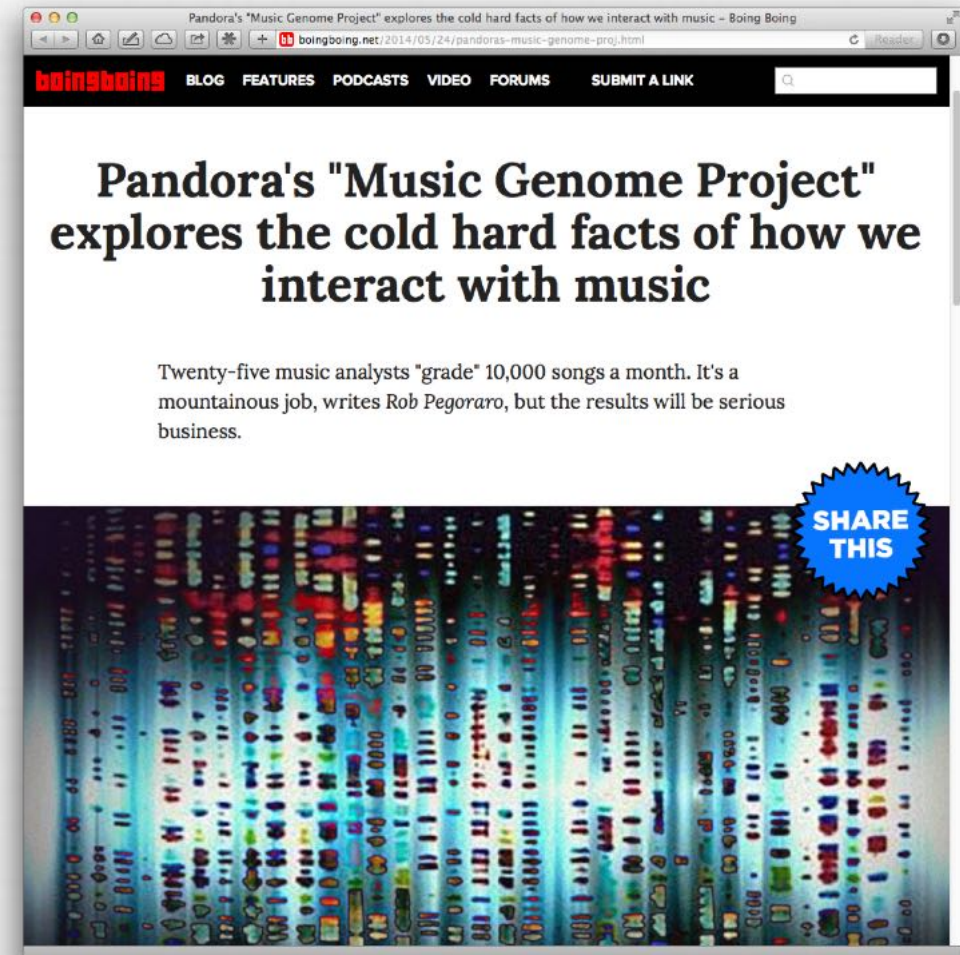
Нейросеть превращает картинку в вектор (embedding) из 300 чисел!



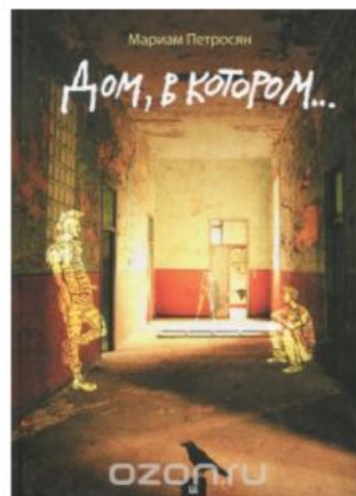
Похожие по косинусу вектора

Пример: Pandora Radio

- Поставили много на контентные признаки
- 480 характеристик у каждой песни!
- Размечают эксперты (25 экспертов размечают 10000 песен в месяц)!



Контентные признаки
в книгах особенно
полезны!



Дом, в котором...

ID 24277965

Новинка

Бестселлер

★★★★★ (155 отзывов)

👍 566

👎 189

У меня это есть

Автор: Мариам Петросян

Издательство: Гаятри/Livebook

ISBN 978-5-904584-69-6; 2015 г.

Язык: Русский

[Дополнительные характеристики](#)

Рекомендуем также



Дом, в котором... В
3 томах (комплект)
509,60 ₽

[В корзину](#)



Тринадцатая
сказка
332 ₽

[В корзину](#)



Дом странных
детей
326,40 ₽

[В корзину](#)



Дом, в котором...
164,90 ₽

[Скачать](#)



Убить
пересмешника...
287,20 ₽

[В корзину](#)

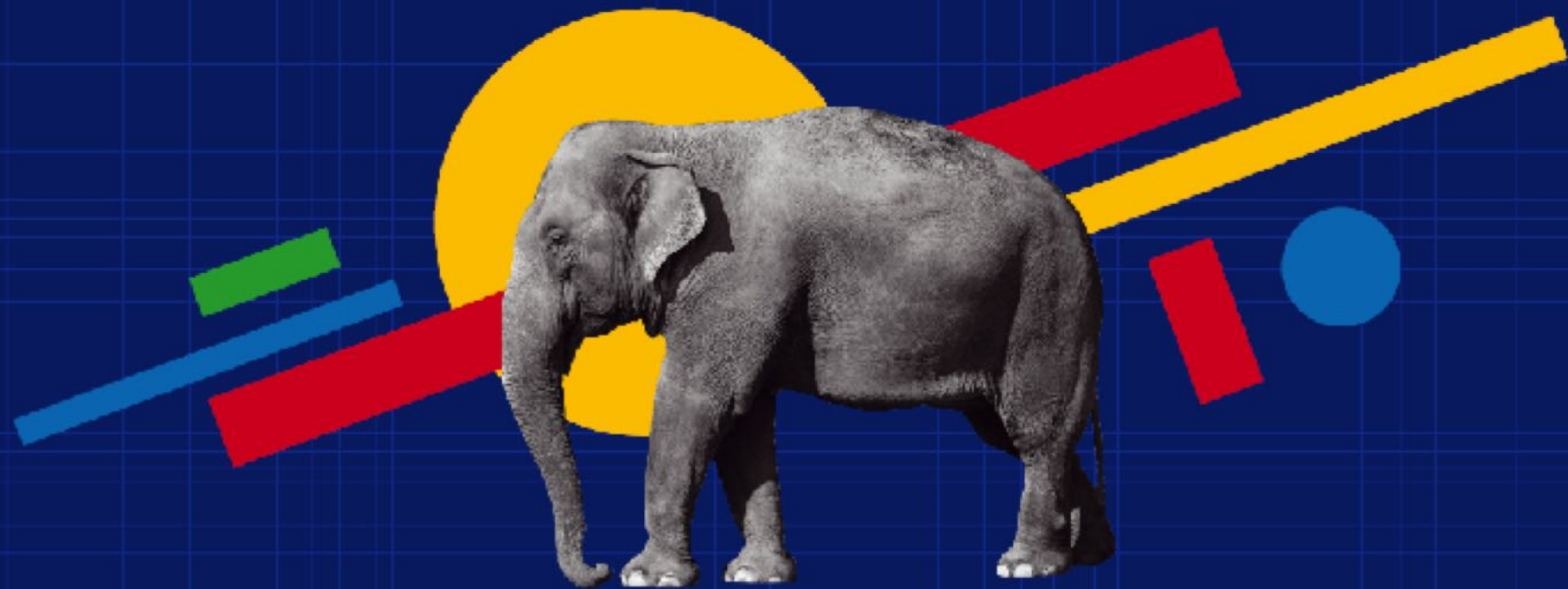
Пример: Surfingbird



Выбери интересы, чтобы мы могли понять, что тебе рекомендовать

Далее





BIG DATA IS LOVE

NEWPROLAB.COM