Кластеризация. Topic Modelling.

Маша Шеянова, masha.shejanova@gmail.com

Обучение без учителя

Для чего?

Итак, у нас есть данные, но нет про них “правильных ответов”. Можем ли мы всё ещё сделать с ними что-то толковое?

- научиться по контексту слова предсказывать само слово (ага, Word2Vec)
- рекламное агентство может раскидать пользователей по “кучкам” по интересам и таргетить каждую отдельно
- “человеко-читаемо” нарисовать на плоскости сложное явление
- снизить требуемый объём памяти и время вычислений

Обучение без учителей — часто инструмент для **анализа**, а не продукта. Ещё чаще — **промежуточный шаг**, чтобы потом лучше решить задачу с учителем.

Виды

- кластеризация

“Раскидай мои фотографии по папочкам. Сам реши, по каким”.

У нас есть выборка объектов, но нет заданных классов. Мы хотим разбить их на группы так, чтобы объекты в разных группах сильно отличались.

- снижение размерности

Часто приходится иметь дело с данными больших размерностей. Их сложно хранить и обрабатывать. Мы хотим снизить размерность, оставив наиболее значимые компоненты. Пример: визуализация в 2D.

Supervised vs. Unsupervised ML

С учителем:

- нужны размеченные данные (дорого, не всегда есть, зависим от качества)
- много хороших алгоритмов, метрик, понятно, чего мы хотим достичь
- если данные хорошие и из них можно извлечь говорящие признаки, хороших результатов добиться легко

Без учителя:

- разметка не нужна, ура! данные валяются на каждом шагу.
- непонятно, как измерять качество (или сложнее понять)
- хороших результатов добиться сложно (сначала понять бы, что нужно)

Кластеризация

Что и зачем

Разбивает объекты на кучки по неизвестному признаку. Сделать так, чтобы похожее было с похожим.

Например:

- разбить покупателей на кучки, а потом понять, что кому нужно
- управлять новостными потоками, понимая что о чём
- разложить фотографии по папочкам
- активное обучение (какие данные надо разметить для supervised ml)
- найти необычное поведение (чего угодно где угодно, например тех же покупателей)

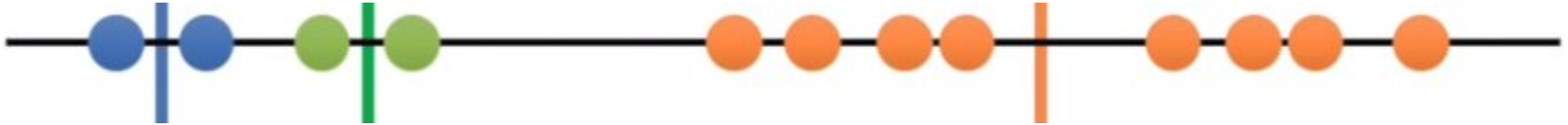
k-means: идея (картинки — из видео)

Допустим, у нас есть точки на прямой, и мы хотим разбить их на 3 кластера.

Давайте закинем в эти данные (куда придётся) 3 кружка — центроида — и для каждой точки найдём, к какому кружку она ближе.

k-means: идея

Когда все точки покрашены в цвета самых близких центроидов, мы двигаем центроиды в среднее значение подключённых к ним точек...



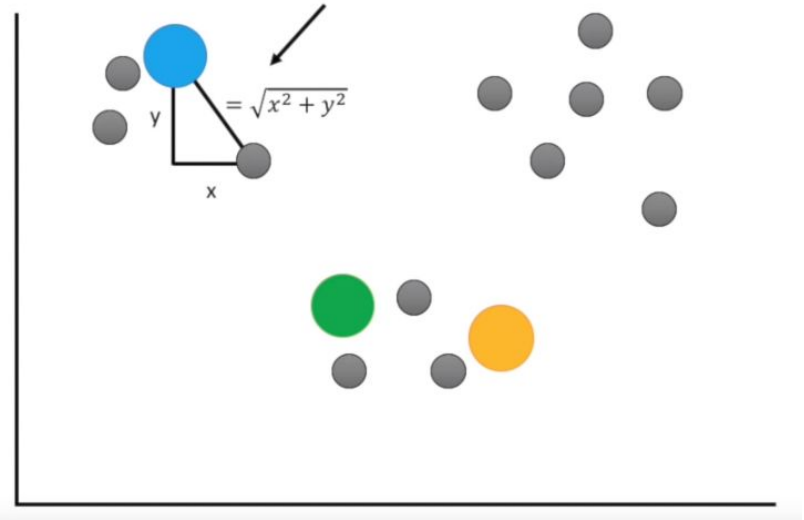
... и повторяем предыдущий шаг. Пока не сойдётся.

На плоскости

... то же самое, считаем расстояние от центоридов до точек.

Правда, вместо простого xc - xn теперь более хитрый подсчёт расстояния.

Евклидово расстояние:

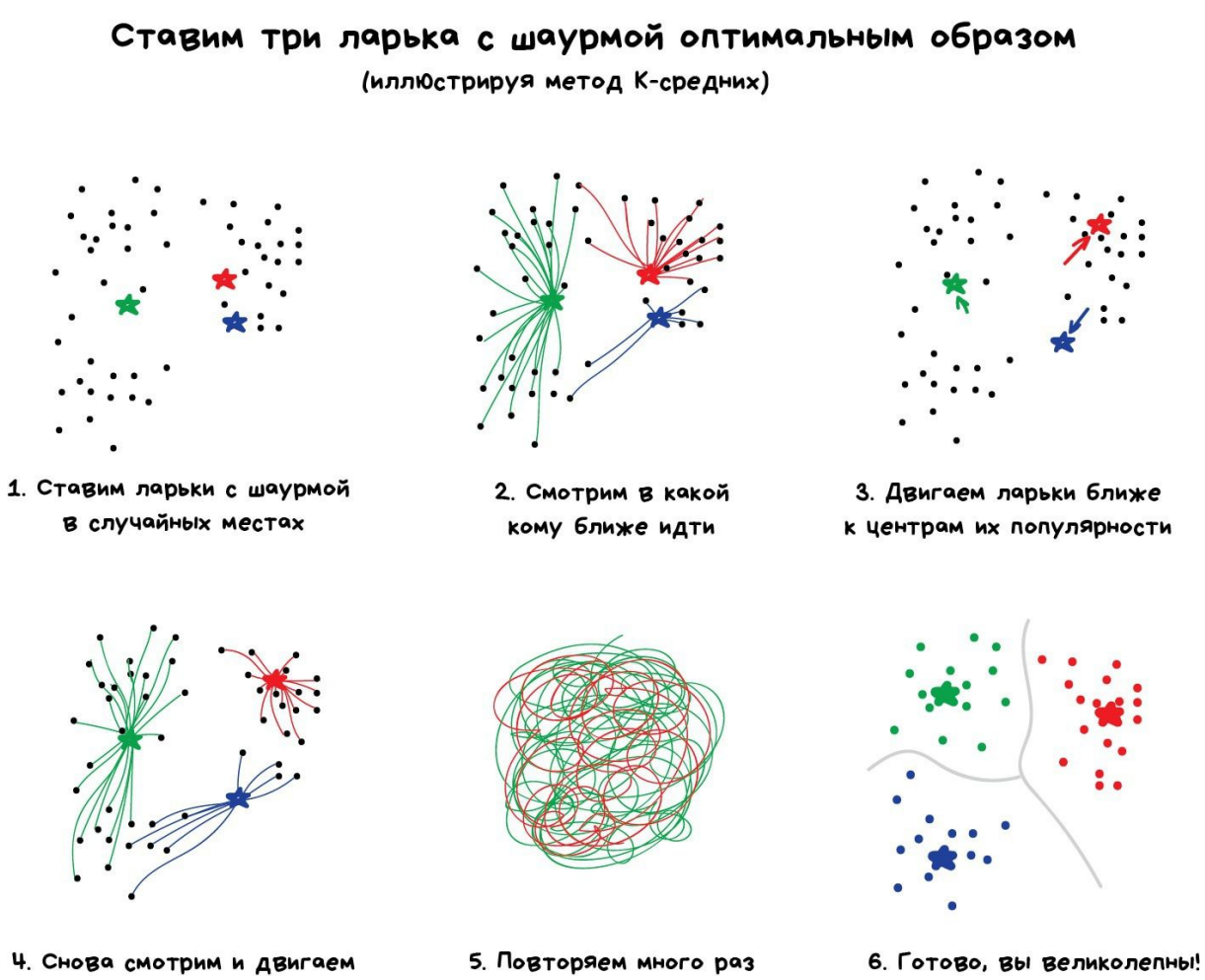


$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2}$$

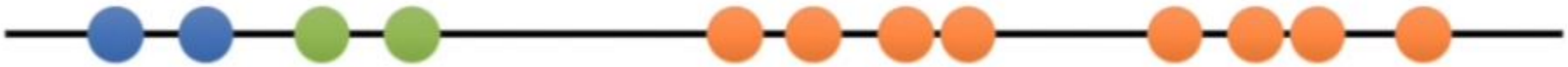
K-means...

... на примере ларьков с шаурмой (отсюда, конечно).

А вот здесь есть ещё хорошая визуализация в движении.



Но что делать, если кластеры построились плохо?



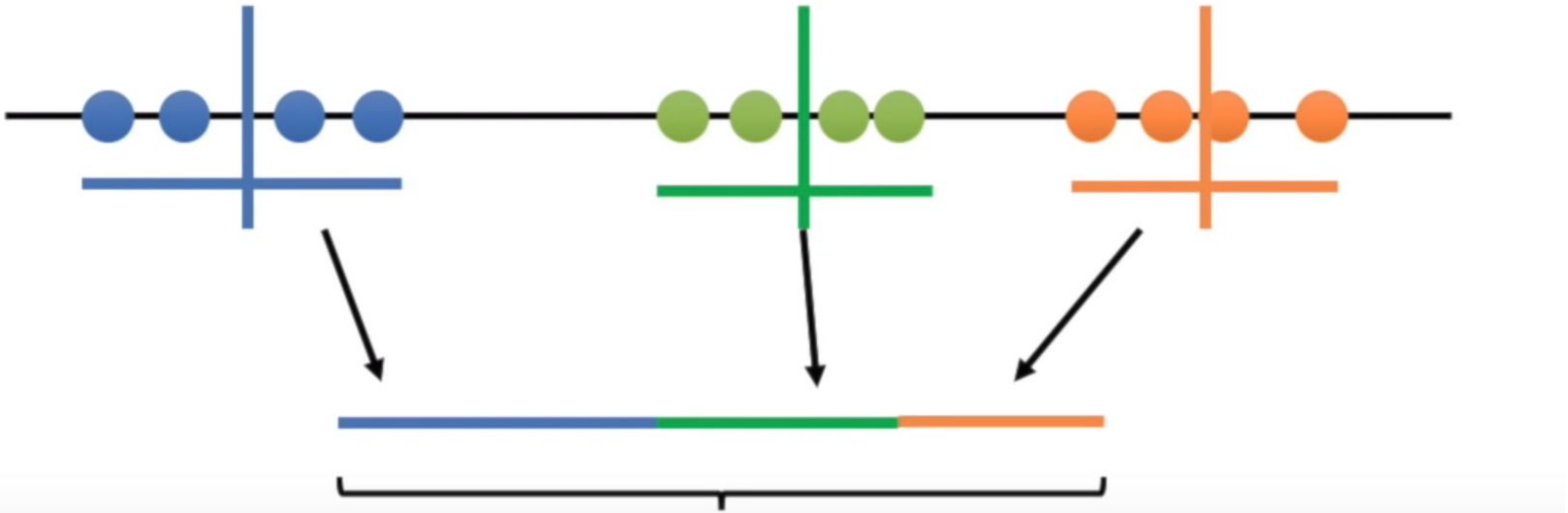
И вообще, как понять, что они построены плохо, ведь ответов у нас нет! Возьмём каждый кластер и посчитаем **дисперсию**, или среднее отклонение от центров кластеров.

Чем больше отклонение от центра, тем хуже.

А теперь будем случайным образом кидать центроиды несколько раз и использовать дисперсию как критерий, насколько хорошо получилось.

Хорошие кластеры

Вот так гораздо лучше — и среднее отклонение от центров меньше.



K-means: алгоритм

- случайно определить k центроидов
- для каждого объекта найти ближайший центроид и приписать его к соответствующему кластеру
- передвинуть центроиды к центрам своих кластеров
- посчитать дисперсию
- повторять предыдущие шаги, пока не сойдётся
- когда сошлось, посчитать дисперсию
- повторять все предыдущие шаги сколько хотим, а потом выбрать лучший результат

k-means в sklearn

from sklearn.cluster import KMeans

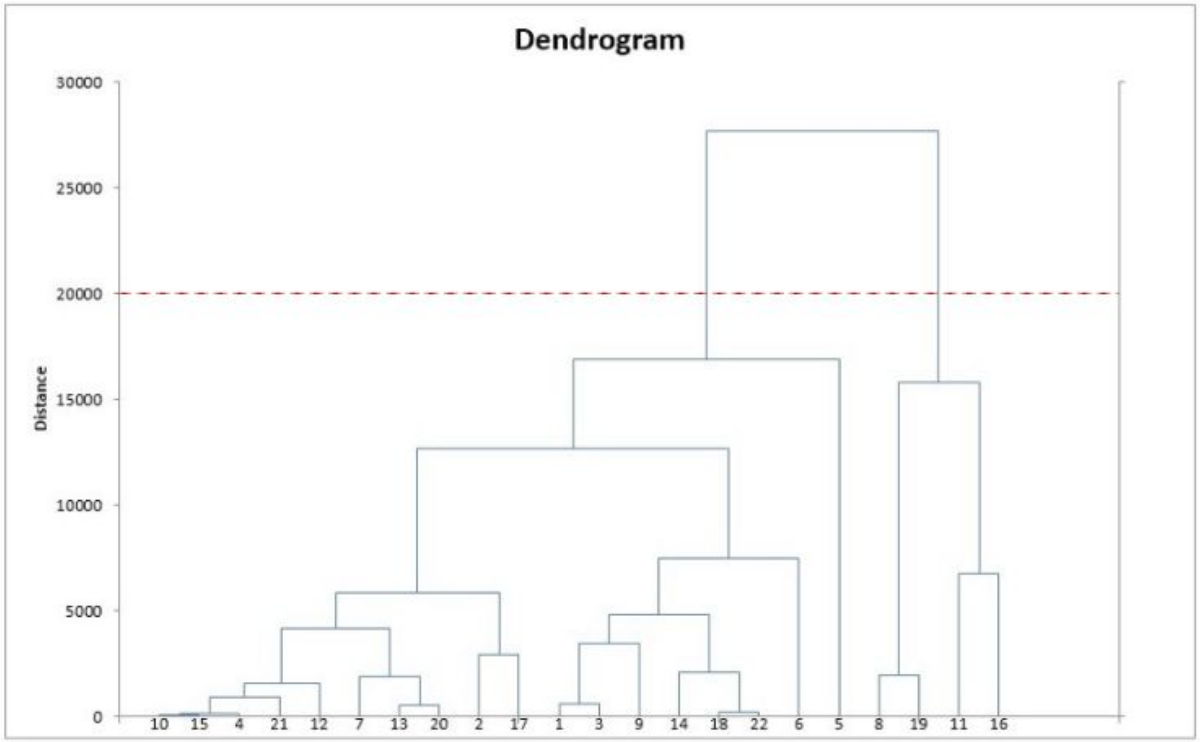
- n_clusters — количество центроидов (дефолт 8)
- init — как инициализировать центроиды
  - k-means++ — умный способ кинуть центроиды хорошо
  - random — случайным образом
  - ndarray — задать самим
- n_init — сколько раз кидать разные центроиды
- max_iter — сколько раз двигать центроиды

Hierarchical clustering

Почти то же самое , но позволяет решить, сколько кластеров надо, позже.

1) выбрать N кластеров (на первом шаге кластеров столько же, сколько объектов было изначально)
2) слить два кластера, которые ближе всего (for some definition of “ближе”)
3) пересчитать расстояния между кластерами (“адрес” нового кластера — среднее между его точками)
4) продолжать 1-3, пока не останется один кластер
5) выбрать, какое расстояние будем считать достаточным, чтобы разбивать на кластеры (или сколько кластеров мы хотим в итоге)

Hierarchical clustering

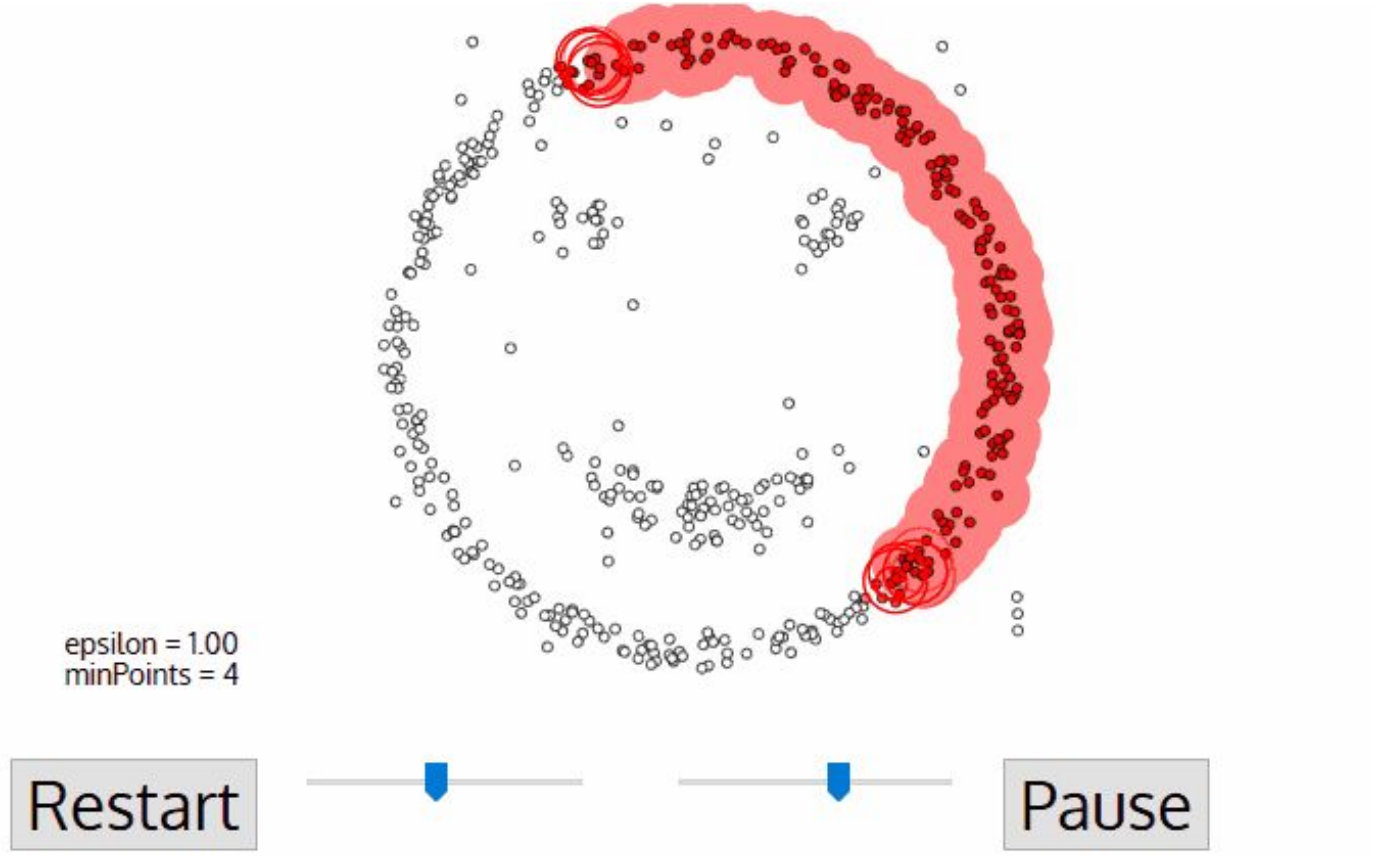


Дендрограмма кластеров.

Выбираем, сколько кластеров нам надо и проводим прямую, где хотим.

DBSCAN

Density-based spatial clustering.



Метрики оценки кластеризации

Как я уже говорила, измерить это гораздо сложнее. Но метрики всё же есть.

- Rand Index (Adjusted Rand Index)
- Гомогенность
- Полнота
- V-мера

Метрики оценки кластеризации

Rand Index (RI)

Rand Index (RI) выражает схожесть двух разных кластеризаций одной и той же выборки.

N — число объектов, a — число пар объектов, имеющих **одинаковые метки** и находящихся в **одном кластере**, b — число пар объектов, имеющих **различные метки** и находящихся в **разных кластерах**.

$$RI = \frac{2(a+b)}{n(n-1)}$$

Это доля объектов, для которых разные разбиения "согласованы".

Adjusted Rand Index

Берём Rand Index и делаем с ним так, чтобы он давал значения близкие к нулю для случайных кластеризаций при любом n и числе кластеров:

$$ARI = \frac{RI - E[RI]}{max(RI) - E[RI]}$$

Его значения, близкие к нулю, соответствуют случайным разбиениям, а положительные значения говорят о том, что два разбиения схожи.

Тематическое моделирование

Что и зачем

Тема — “о чём документ” $\backsimeq$ набор часто совместно встречающихся слов

Мы считаем, что употребление того или иного слова зависит от темы (например, слово *днк* вероятнее всего встретится в тексте про генетику, а *кварк* — про физику частиц). А тема зависит от документа. Мы хотим найти способ **разложить документы по темам**.

Зачем это нужно:

- поиск в специализированных областях
- мягкая кластеризация текстов для эксплоративного анализа
- “продвинутый” эмбеддинг документа

topic modeling vs. clustering

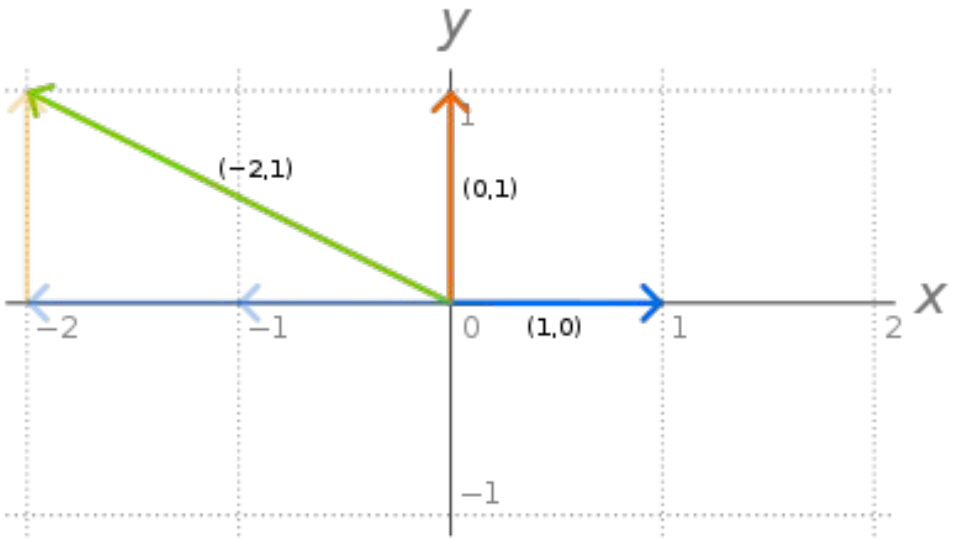
Что похожего: есть документы, раскидываем их по кучкам, заранее не знаем по каким.

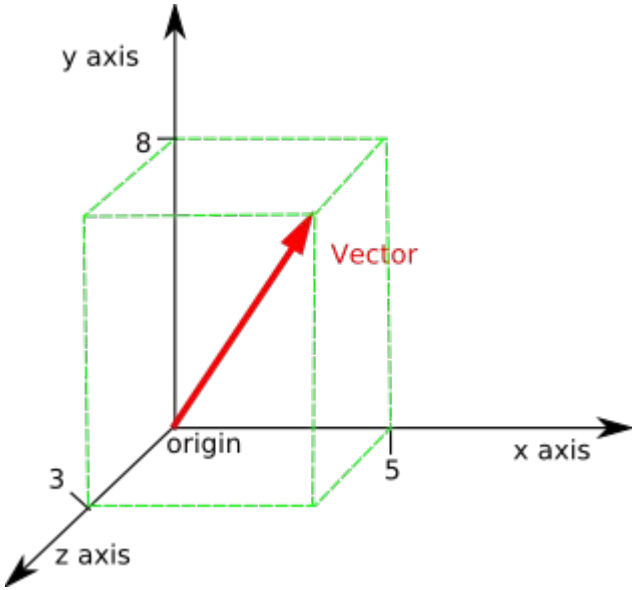
Что разного: у одного документа может быть высокая степень принадлежности больше, чем к одной теме.

PCA (метод главных компонент): идея

Базис линейного пространства

Стандартный базис:





Замена базиса

На самом деле, базисные вектора можно выбирать как угодно — главное чтобы можно было выразить через них все вектора пространства.

(И чтобы сами базисные вектора нельзя было выразить друг через друга).

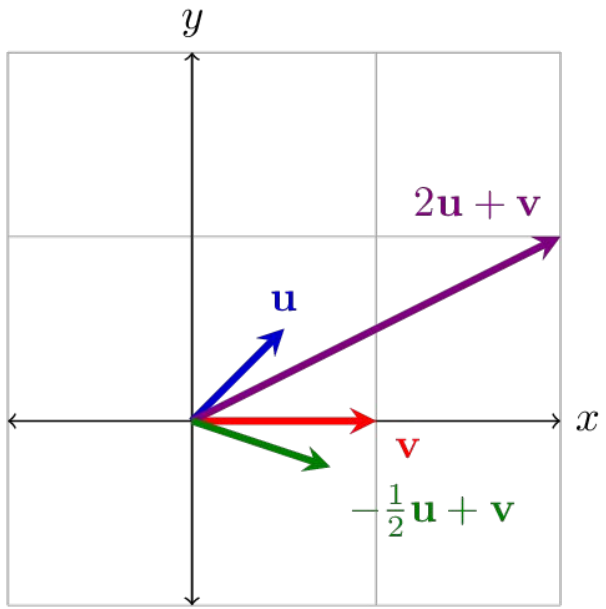
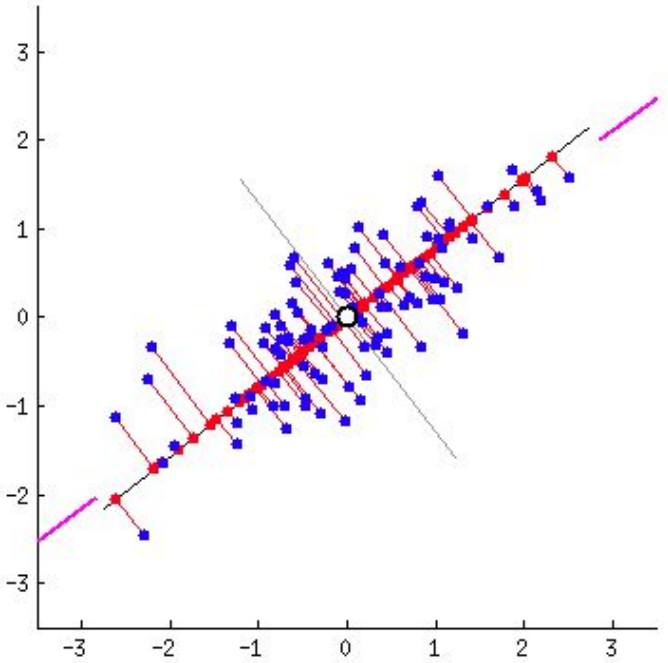


Figure 1: Vector combinations.

PCA

Найдём такой базис, чтобы как можно лучше выразить как можно больше значений за счёт фиксированного количества базисных векторов.

Сделаем проекцию всех данных на эти вектора.



SVD (сингулярное разложение): реализация

Для текстов: матрица слово-документ

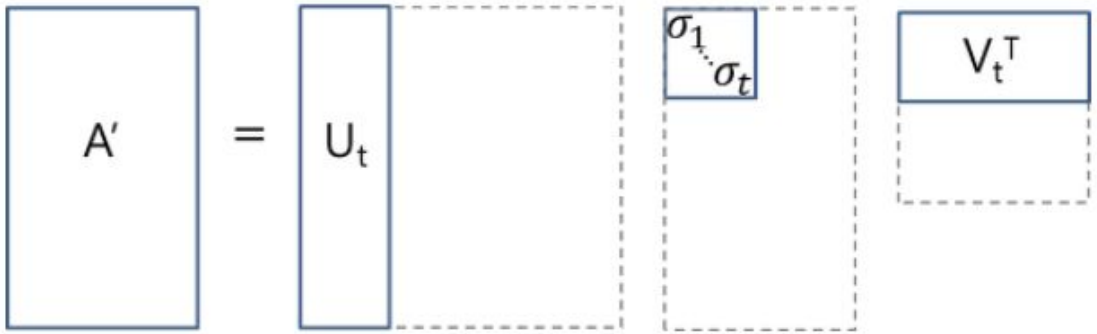
Для начала, считаем табличку о том, сколько раз какое слово вошло в какой документ, например, с помощью CountVectorizer.

	котик	играть	авокадо	манго
документ 1	2	2	0	0
документ 2	1	3	0	0
документ 3	1	2	0	1
документ 4	0	1	1	2
документ 5	0	0	3	2

Truncated SVD

$$A \approx U_t S_t V_t^T$$

Intuitively, think of this as only keeping the t most significant dimensions in our transformed space.



(скрин из вот этой статьи)

Truncated SVD = LSA (latent semantic analysis) в тематическом моделировании

Что в итоге

- в средней матрице диагонали по убыванию выстроены компоненты — измерения “хорошего” базиса; чем выше, тем значимей компонента
- выбираем первые n (сколько хотим) компонент; их мы будем сохранять, а остальные выкинем
- в итоге значительно сократим объём используемой памяти
- и дополнительно получим разложение документов по этим компонентам, или, как говорят в тематическом моделировании, темам
- (NB: компоненты выстроены по убыванию для всего датасета, но каждый документ имеет свои пиковые компоненты)

Truncated SVD в sklearn

from sklearn.decomposition import TruncatedSVD

гиперпараметры:

- n_components — какого размера должны быть конечные векторы
- algorithm — randomized, arpack
- n_iter

Может применяться в связке с классификацией.

SVD на собачках

SVD можно применять и для других матриц — например, для картинок, ведь картинки — это просто матрицы из пикселей.

При большом количестве компонент разница незаметна.

(источник)

Ресурсы

О том, что было

Почитать:

- Machine Learning for Humans: Unsupervised Learning
- The 5 Clustering Algorithms Data Scientists Need to Know
- про кластеризацию на Хабрахабр (рус)

Посмотреть:

- StatQuest: k-means
- StatQuest: hierarchical clustering
- StatQuest: PCA