Матричные разложения

SVD и его применения

Иван Сафонов
БПМИ 182

29.09.2020

План (SVD)

- Введение
- Наилучшее малоранговое приближение и применения
- PCA (метод главных компонент) и применения
- Рекомендательные системы
- LSA (латентно-семантический анализ)

Определение SVD

- Квадратная матрица U размера nxn называется **унитарной**, если $U^TU=I_n$
- Пусть A - матрица размера mxn. Разложение $A=U\Sigma V^T$ называется **сингулярным разложением** матрицы A. При этом здесь:
  - U унитарная матрица размера mxm
  - Σ диагональная матрица, на диагонали стоят $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_k \geq 0$ (k = min(m, n))
  - V унитарная матрица размера nxn
- можно оставить только r столбцов в U, Σ, V, где r = rank(A) (compact SVD)

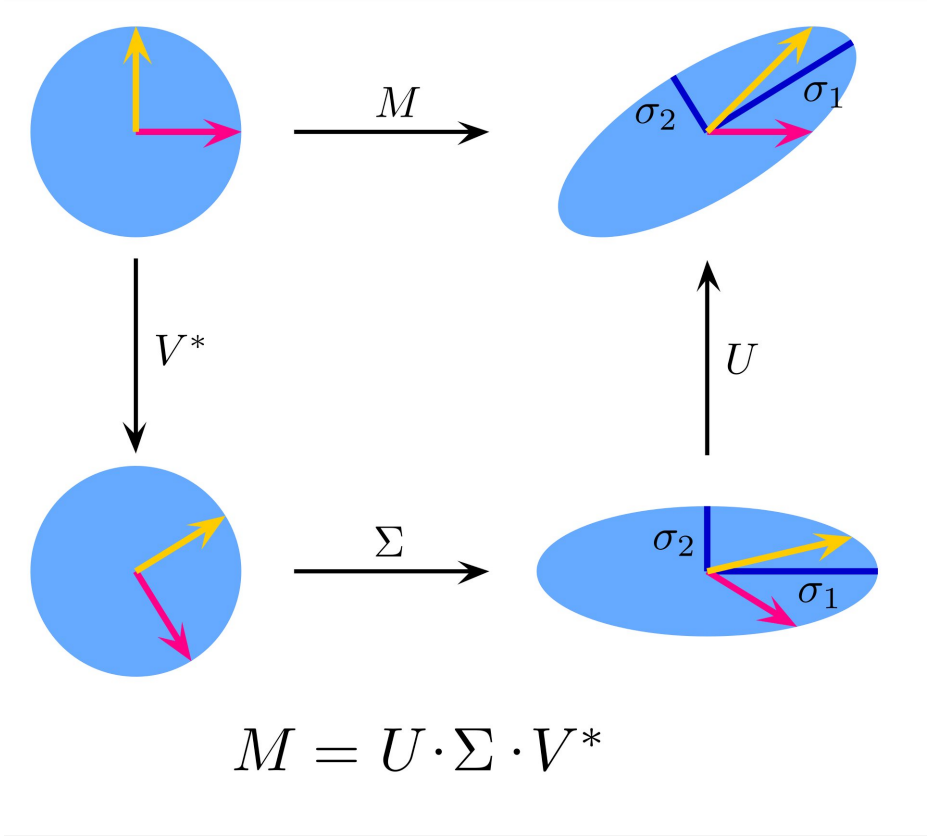
Некоторые свойства

- разложение существует для любой матрицы
- сингулярные числа определены однозначно
- количество ненулевых сингулярных чисел равно рангу матрицы

Физический смысл SVD

Если посмотреть на матрицу как на линейное отображение из R^n в R^m, то это отображение можно в терминах SVD разложить в композицию трех линейных отображений:

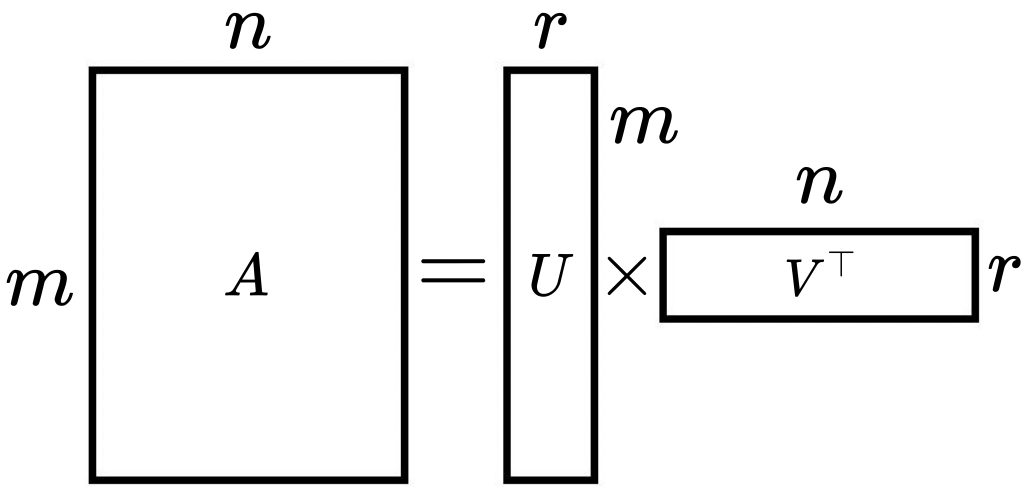
1. Некоторое движение пространства. Оно не меняет расстояния
2. расширение новой i-ой координаты в σ_i раз
3. Снова некоторое движение пространства



$$M = U \cdot \Sigma \cdot V^*$$

Скелетное разложение

- Пусть A - матрица размера mxn, r = rank(A). Разложение $A=UV^T$ называется **скелетным разложением**. При этом тут:
  - U матрица размера mxr
  - V матрица размера nxr



Плюсы

- ❖ прямая связь с SVD
- ❖ удобный взгляд на малоранговые матрицы

Удобный взгляд

- Векторы строки матрицы U это u_1, u_2, …, u_m
- Векторы строки матрицы V это v_1, v_2, …, v_n
- Вышеперечисленные векторы имеют r координат
- Тогда $A_{ij}=\langle u_i,v_j\rangle$(скалярное произведение)
- Мы можем вычислять элемент матрицы такого вида за O(r)
- Память, которую мы тратим, для того, чтобы сохранить матрицу в таком виде r(n+m) чисел

Не все матрицы имеют маленький ранг, чтобы использовать описанные преимущества.

Вывод: можно пробовать приближать данную матрицу матрицей маленького ранга.

Теорема о наилучшем ранговом приближении

- $A = U\Sigma V^T$ - сингулярное разложение
- Задан некоторый ранг $r \le m$ и $r \le n$
- Векторы строки матрицы U это $u_1, u_2, \ldots, u_m$ (имеют m координат)
- Векторы строки матрицы V это $v_1, v_2, \ldots, v_n$ (имеют n координат)

Тогда матрица $B = \sigma_1 u_1^T v_1 + \sigma_2 u_2^T v_2 + \ldots + \sigma_r u_r^T v_r$ будет матрицей ранга $\le r$, лучше всего приближающей A.

Формально, $||A-B||_F$ будет минимально (Фробениусова норма, квадратный корень из суммы квадратов коэффициентов матрицы).

Заметим, что описанная формула сразу дает нам удобное скелетное разложение матрицы B.

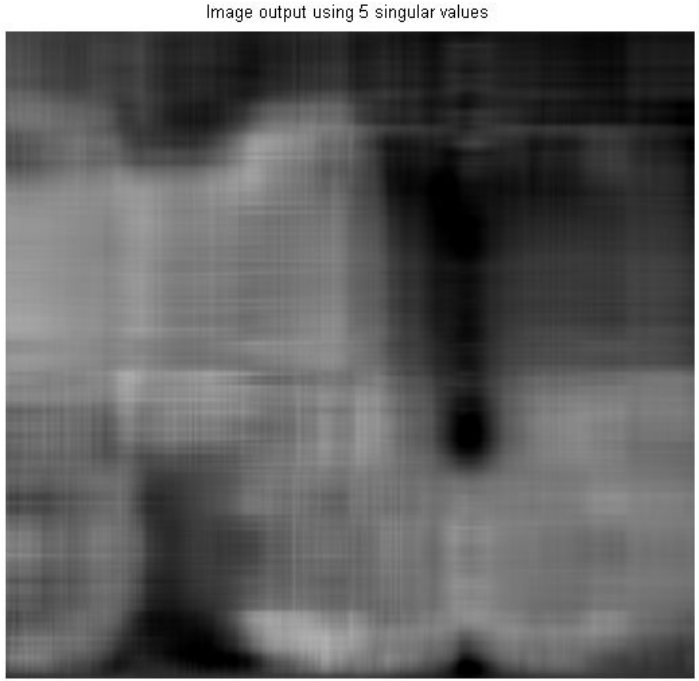
Сжатие изображений

Можно сжать изображение по следующему алгоритму:

1. Взять матрицу цветов её пикселей
2. Выбрать небольшое значение r
3. Найти наилучшее малоранговое приближение этой матрицы ранга ≤ r
4. Теперь можно использовать эту матрицу в качестве изображения, но памяти мы будем тратить меньше и визуальные изменения картинки будут незаметны (или несильно заметны) человеческому глазу

Мы будем использовать долю $\frac{r(n+m)}{nm}$ от изначальной памяти.

Сжатие изображений (пример)



Сжатие изображений (пример)

Image output using 30 singular values

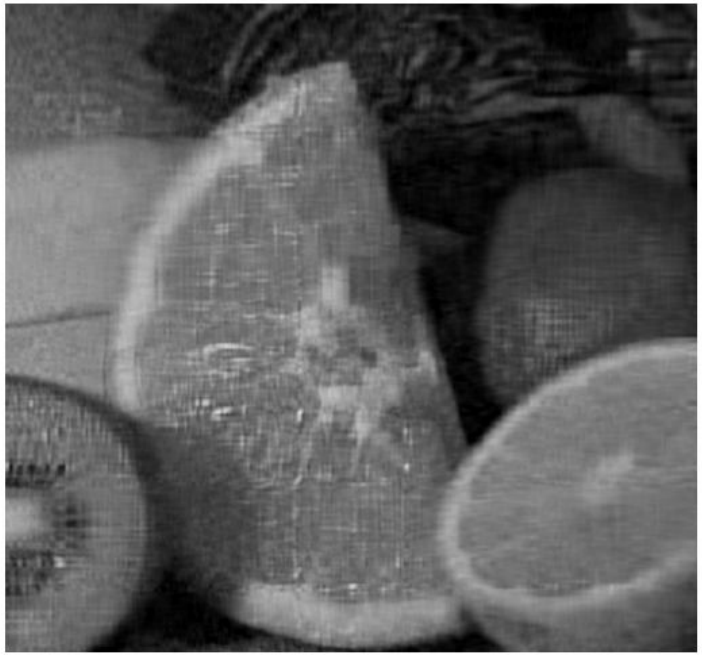
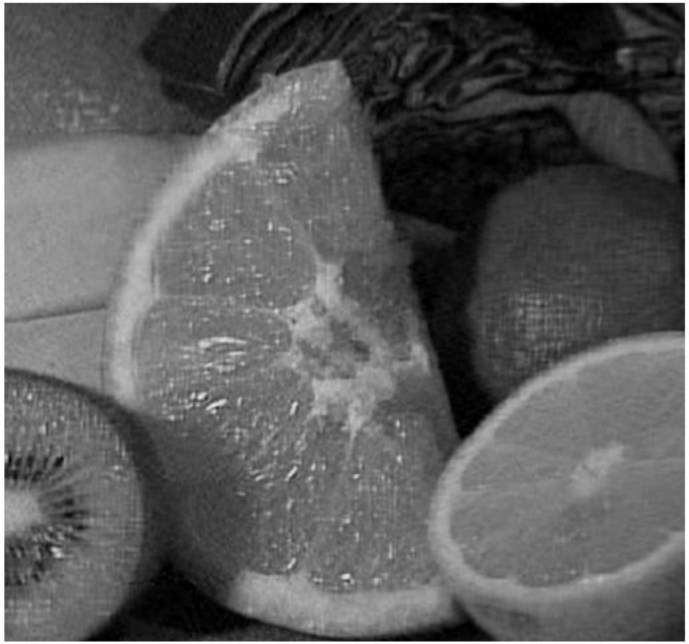
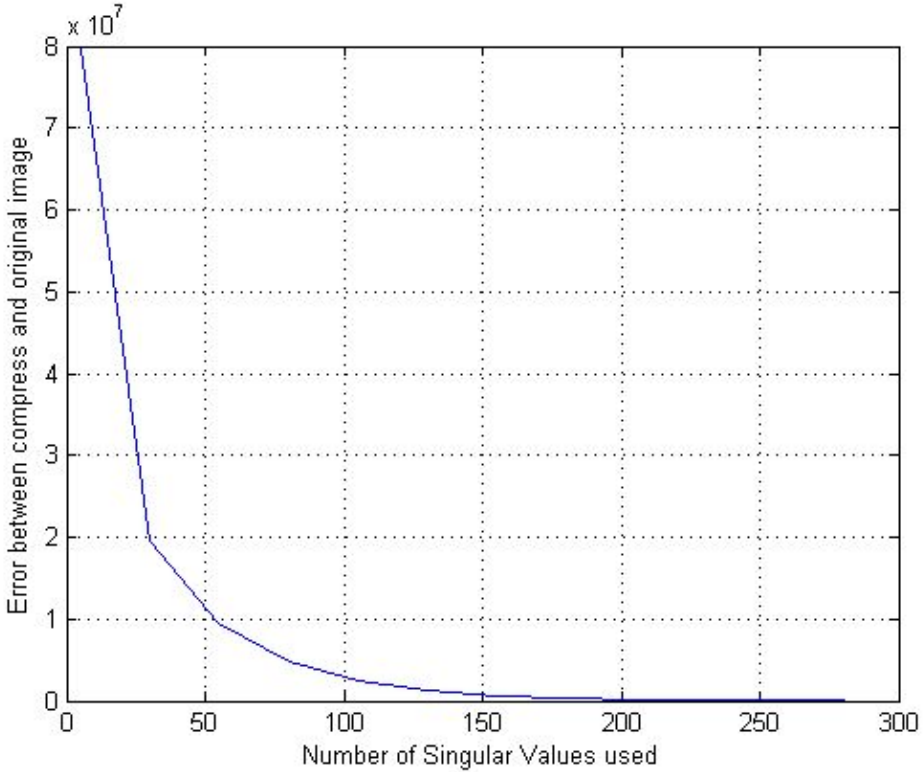


Image output using 55 singular values



Сжатие изображений (пример)



- Матрица изображения имеет размер 300x300
- Потратим примерно в 3 раза меньше памяти (используя 55 сингулярных чисел)
- По такому алгоритму можно сжимать не только фотографии, но другие матрицы

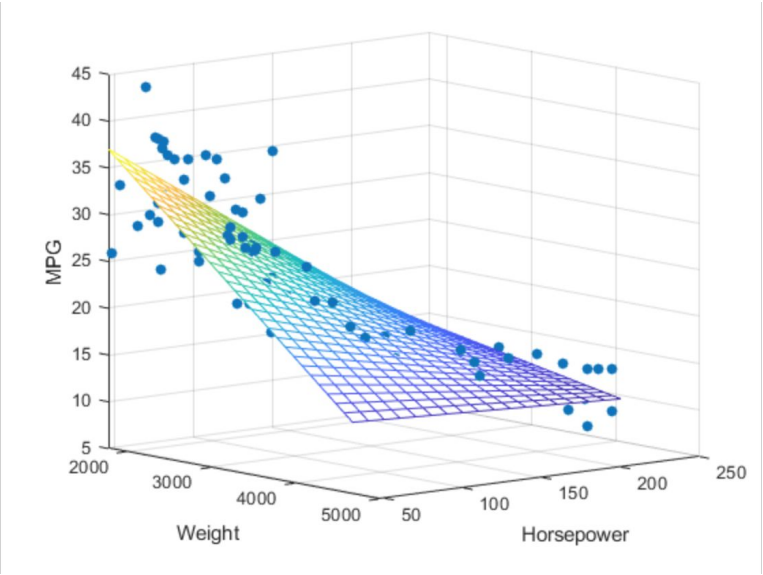
Аппроксимация линейным многообразием

- Пусть у нас есть n точек $x_1, x_2, \ldots, x_n$ в m-мерном пространстве
- Мы хотим найти некоторое линейное многообразие размерности ≤ k (где k ≤ m), такое что сумма квадратов расстояний от наших точек до него минимально
- Линейное многообразие: $L_k = \{a_0 + \alpha_1 a_1 + \ldots + \alpha_k a_k \mid \alpha_1, \alpha_2, \ldots, \alpha_k$-вещественные$\}$ Здесь $a_0, a_1, \ldots, a_k$ какие-то m-мерные вектора
- Ищем такое L_k, что сумма $\text{dist}(x_i, L_k)^2$ будет минимальна

Пример применения

- Можем увидеть какую-то линейную зависимость в данных, понять вокруг чего лучше всего концентрируются точки

Примеры



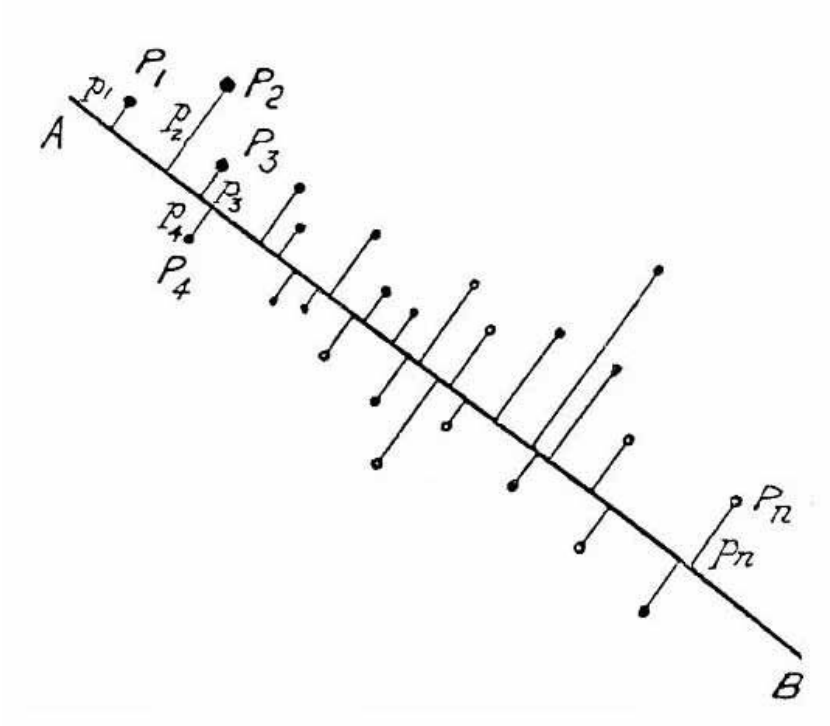


Рисунок к знаменитой работе Пирсона (1901)

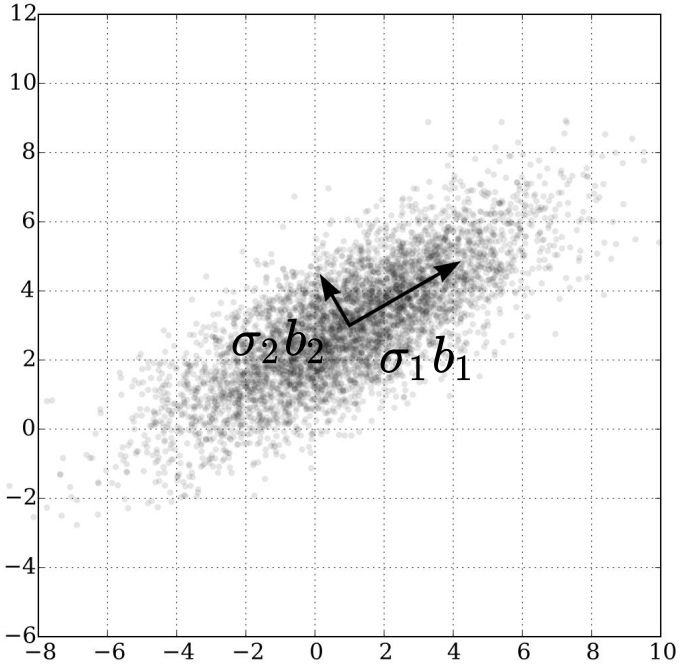
Решение с помощью SVD

- Нетрудно понять, что оптимальное a_0= (среднее x). Можем вычесть его
- Рассмотрим матрицу A=[x_1 | x_2 | … | x_n] (размер mxn)
- Рассмотрим её сингулярное разложение $A = U\Sigma V^T$
- Рассмотрим матрицы:
  1. B=[первые k столбцов U] (размер mxk). Пусть B = [b_1|b_2|…|b_k]
  2. C=[первые k строк V^T] (размер kxn). Пусть C = [c_1 | c_2 | … | c_n]
- Как мы уже знаем A≈BΣC (наилучшее приближение A матрицей ранга ≤ k)
- Тогда мы можем рассмотреть пространство L_k=span(b_1, b_2, …, b_k)
- Можно понять, что a_0+L_k будет нужным линейным многообразием
- (Σc_i) будет вектором координат проекции x_i на L_k (в базисе b_1, b_2, …, b_k)

Почему вообще брали SVD?

- Заметим, что сумма $dist(x_i, L_k)^2 = ||A-B\Sigma C||_F^2$. A SVD как раз будет давать наименьшее значение правой части.

Примеры



Физический смысл:
Когда мы делаем SVD, мы выделяем ортогональный базис, лучше всего описывающий наше множество точек. Поэтому вектора b_1, b_2, …, $b_{min(n,m)}$ называются главными компонентами.

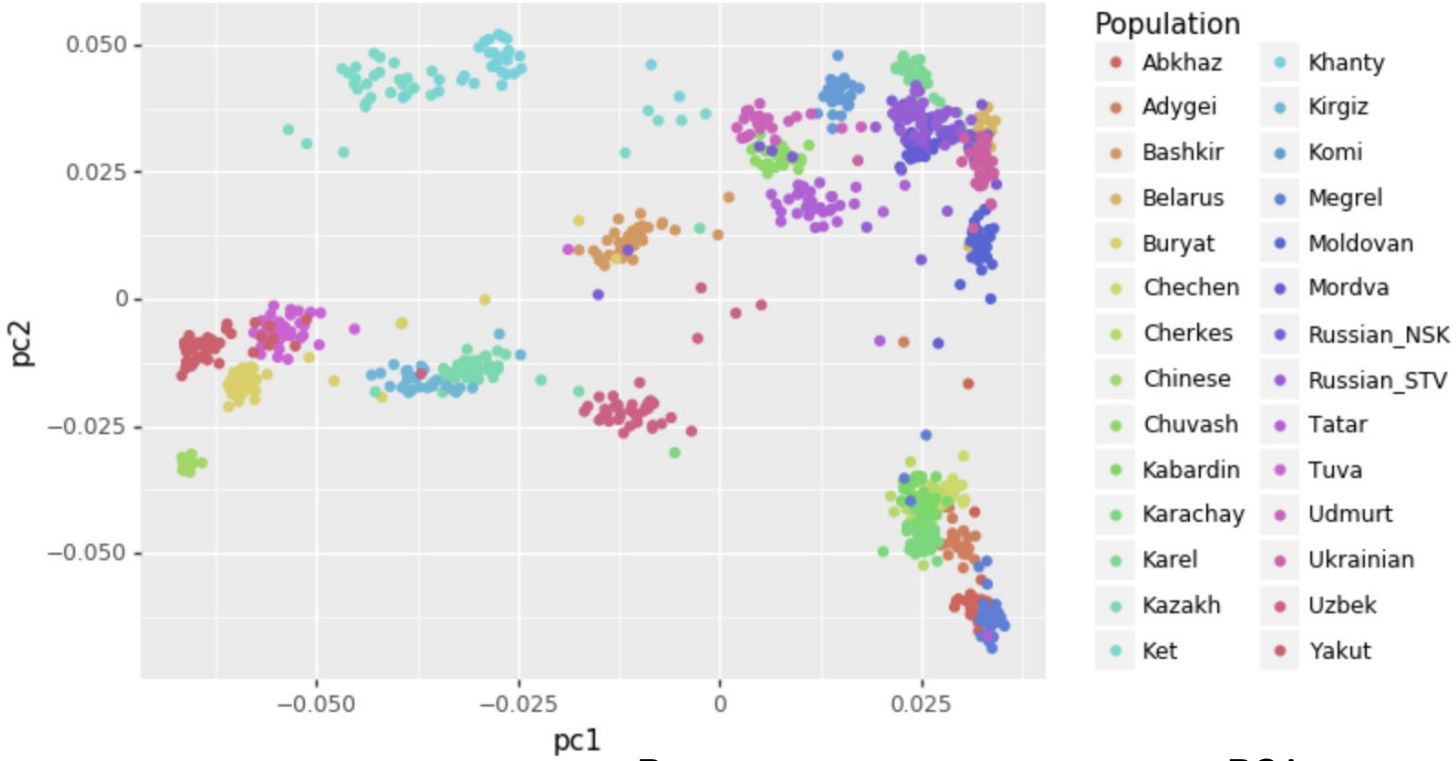
Если нас просят оставить размерность k, то мы должны оставить самые первые k главных компонент.

Физический смысл 2:
Если мы предположим, что x_1, x_2, …, x_n - выборка из m-мерного нормального распределения, то мы находим оценку максимального правдоподобия, выделяя математическое ожидание и базис из m векторов, координаты при которых будут независимыми нормальными распределениями $\mathcal{N}(0, \sigma_i^2)$

PCA (principal component analysis)

- Пусть у нас есть n точек $x_1, x_2, \ldots, x_n$ в m-мерном пространстве
- Вместо них мы хотим получить новые точки $p_1, p_2, \ldots, p_n$ в k-мерном пространстве, где k сильно меньше m. Неформально мы хотим, чтобы потерялось как можно меньше информации об изначальных точках
- Из сказанного выше понятно, какие точки надо выбрать, чтобы сохранилось как можно больше информации: пусть p_i будет координатами x_i в базисе главных k компонент
- Мы уже получали, что $p_i = \Sigma c_i$
- Поскольку $b_1, b_2, \ldots, b_k$ - ортонормированный базис L_k, расстояния между p_i и p_j будет равно расстоянию между проекциями x_i и x_j на наше линейное многообразие
- Также поскольку базис был ортонормированный, произвольный вектор a будет отображаться в вектор $(\langle a, b_1 \rangle, \langle a, b_2 \rangle, \ldots, \langle a, b_k \rangle)$

PCA (примеры)



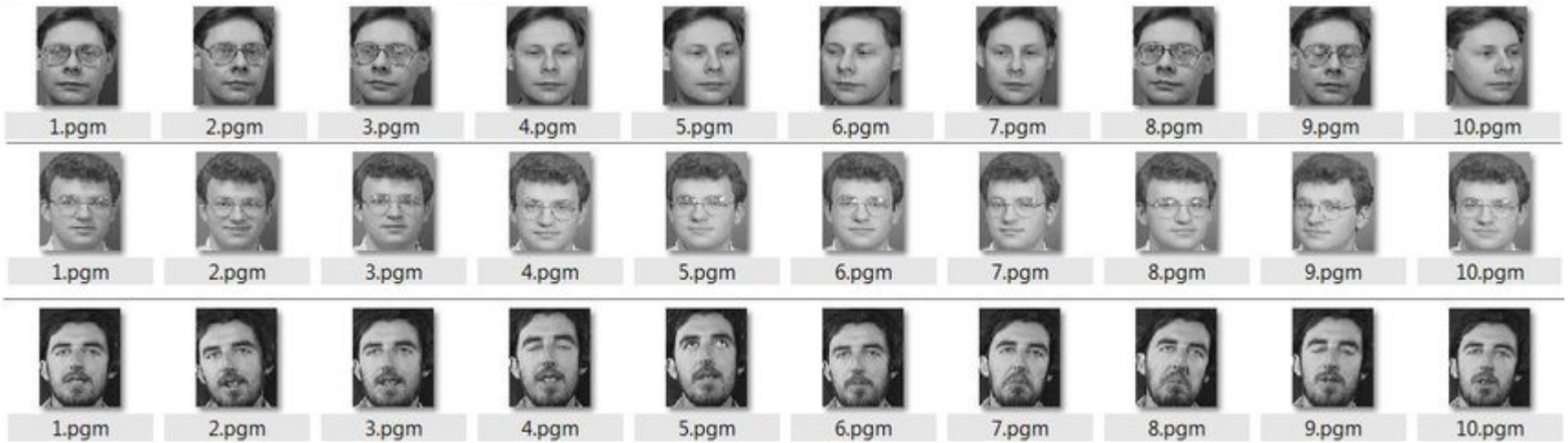
Народы проживающие на территории России, две главные компоненты PCA векторов, построенных на основе ДНК представителей этих народов.

Применения PCA

- Если у вас много числовых признаков - их количество можно сильно уменьшить с помощью PCA и при этом признаки могут не сильно ухудшиться от этого
- Если мы решаем задачу кластеризации/поиска ближайших соседей, но все вектора имеют слишком большую размерность (что может быть плохо для метода ближайших соседей), мы можем уменьшить ее с помощью PCA
- Можем классифицировать новые объекты, находя ближайших соседей к векторам, к которым применили PCA

Eigenfaces

- Научимся выделять из фотографий признаки с помощью PCA и распознавать лица
- У нас будет обучающая выборка, состоящая из фотографий лиц, которые мы хотим распознавать. Для каждого лица лучше выделить несколько ракурсов

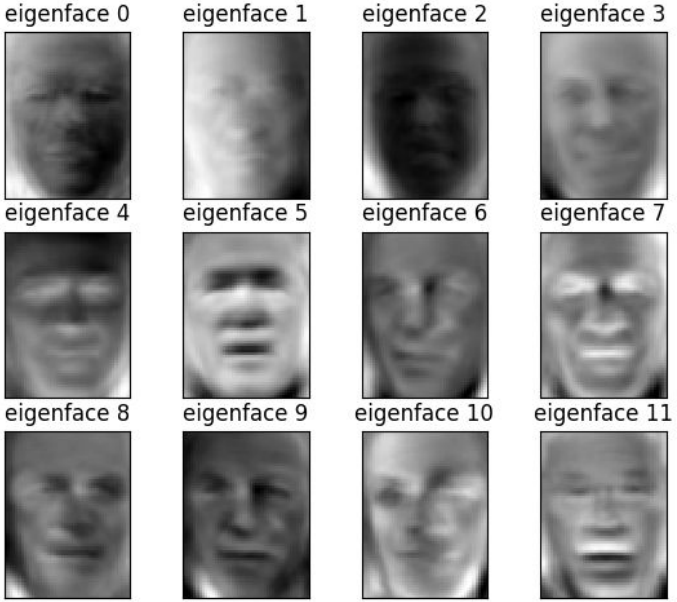


Eigenfaces

- Каждую фотографию переведем в матрицу цветов и выпишем в длинный вектор
- Мы получим l векторов в (mn)-мерном пространстве, где
  - ➔ l количество фотографий в выборке
  - ➔ mxn размер каждой фотографии
- Вычтем из всех векторов выборки среднее значение векторов
- Сделаем PCA на небольшую размерность $k << mn$
- Каждая фотография теперь соответствует вектору небольшого размера

Eigenfaces

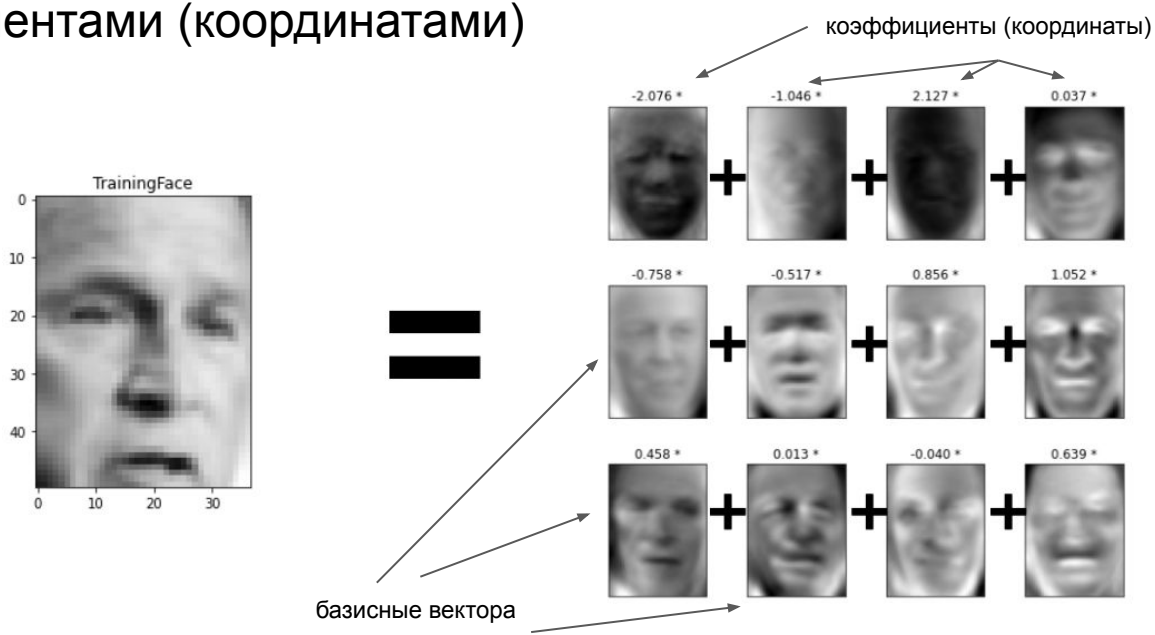
Так будут выглядеть базисные вектора (мы снова придаем им форму mxn). То есть мы выделяем какие-то основные признаки лиц.



Eigenfaces

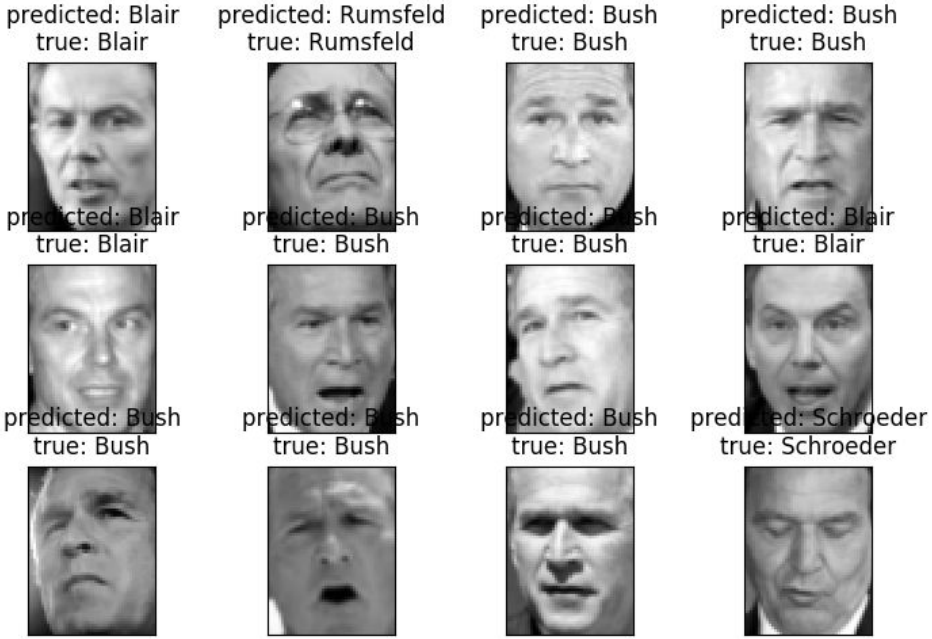
- Каждой компоненте PCA будет соответствовать некоторый вектор из базиса, который мы можем перевести в фотографию
- Любую фотографию мы можем разложить как сумму базисных векторов с некоторыми коэффициентами (координатами)

Получаем такую картинку:



Eigenfaces

- Можем распознавать лица
- Применяем PCA к вектору фотографии
- Используя “расстояния” (евклидово или расстояние Махалонобиса) можем посчитать в какой степени фотография принадлежит к каждому из классов
- Взяв самый близкий класс получаем предсказание
- Метод работает быстро, потому что после PCA вектора будут иметь маленькую размерность

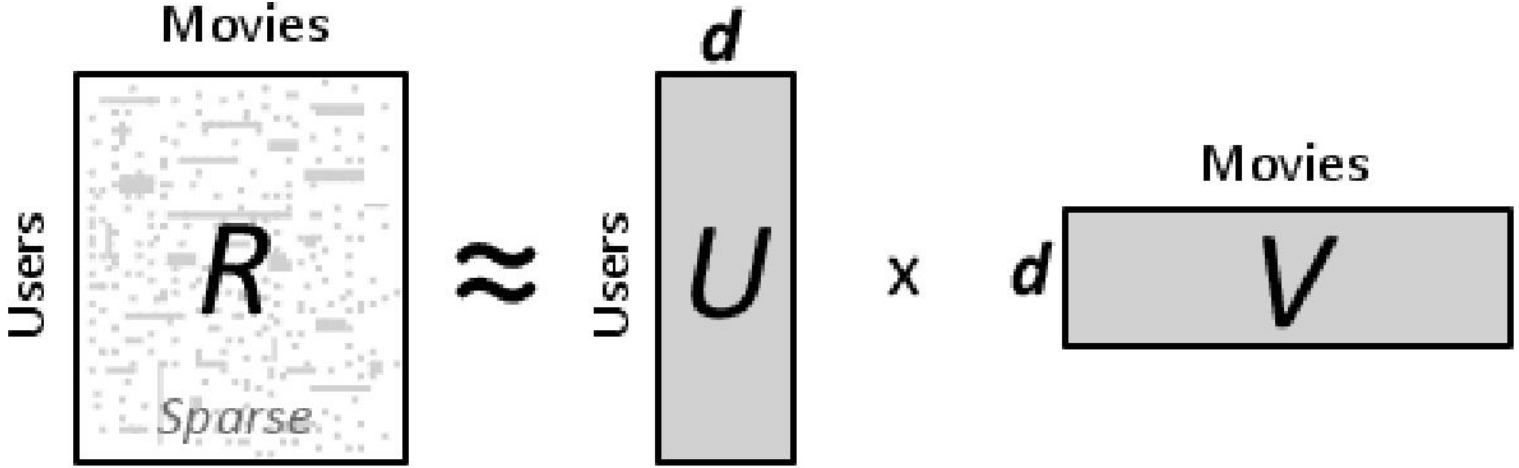


Рекомендательная система (постановка задачи)

- Пусть у нас есть множество пользователей и множество фильмов
- Каждый пользователь поставил оценки некоторым фильмам
- Хотим научиться предсказывать оценки, которые будут ставить пользователи другим фильмам
- Составим матрицу R, строки которых будут соответствовать пользователям, а столбцы - фильмам. Запишем в соответствующие клетки оценки, которые пользователи дали фильмам.
- Эта матрица будет иметь большой размер, но будет **разреженной**, что является очень полезным свойством для алгоритмов.

Рекомендательная система

- Найдем наилучшее скелетное разложение матрицы R для какого-нибудь небольшого ранга d (с помощью SVD)



Рекомендательная система

- Строки матрицы U это u_1, u_2, …, u_m
- Столбцы матрицы V это v_1, v_2, …, v_n
- Эти вектора имеют размерность d
- Тогда мы получаем, что $R_{ij} \approx \langle u_i, v_j \rangle$
- То есть предсказание оценки i-го пользователя j-му фильму мы можем вычислить за O(d)
- Такой метод уже дает неплохие результаты и может быть использован для выделения “скрытых признаков”. Например первая координата вектора будет описывать пол пользователя, вторая его возраст и т.п.

Рекомендательная система (практика)

- Мы знаем только некоторые клетки матрицы R (остальные не определены). Это задача **matrix completion** (восстановление матрицы)
- Введем функцию $f(U, V) = ||R-UV^T||_F^2+\lambda(||U||_F^2+||V||_F^2)$, при этом будем брать в первую норму только те коэффициенты матрицы, которые в R определены. Мы добавляем l2-регуляризацию по стандартным причинам
- Теперь можем сделать градиентный спуск, находя минимум функции

ALS-алгоритм (Alternating Least Squares)

- Выберем изначально какие-то U_0, V_0.
- На i-м шаге:
  - $U_{i+1}=\text{argmin}_U f(U, V_i)$
  - $V_{i+1}=\text{argmin}_v f(U_{i+1}, V)$
- Найти формулу для argmin можно выписав производную для f по U (при фиксированном V) и для f по V (при фиксированном U)

ALS для наилучшего малорангового разложения

- Разберем, как поиск SVD (будем искать в виде скелетного разложения) решается с помощью ALS алгоритма
- У нас есть матрица A размера mxn, мы хотим найти такие U (размера mxk) и V (размера nxk), что $||A-UV^T||_2^2$ минимально
- То есть мы берем функцию $f(U, V) = ||A-UV^T||_2^2$

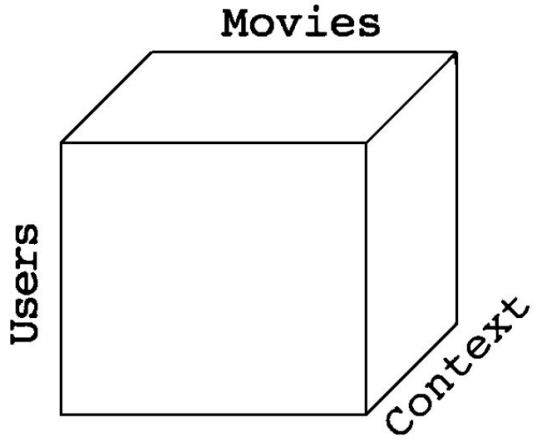
Можно вывести такие формулы пересчета:

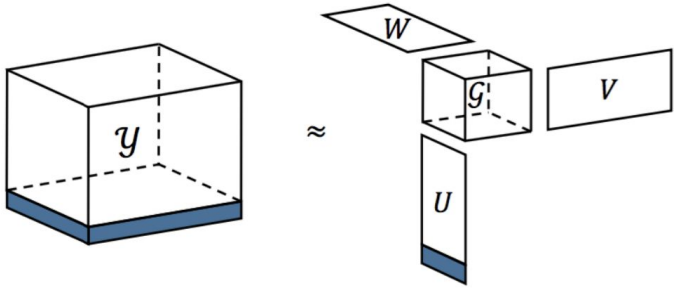
- $AV_i = Q_1R_1$ - QR разложение матрицы AV_i (ортогонализация)
- $A^TQ_1 = Q_2R_2$ - QR разложение матрицы A^TQ_1
- Пересчитываем так:
  - $U_{i+1} = Q_1R_2^T$
  - $V_{i+1} = Q_2$
- Работает быстро, применимо к разреженным матрицам

Рекомендательная система (улучшения)

- Формулу $R_{ij} \approx \langle u_i, v_j \rangle$ можно усложнить, добавив параметры:
  - глобальную константу μ
  - константу для каждого пользователя p_{user}
  - константу для каждого фильма q_{movie}
- Тогда формула предсказания будет $R_{ij} \approx \mu + p_i + q_j + \langle u_i, v_j \rangle$
- Тоже можно оптимизировать ошибку для таких формул

Рекомендательная система (контекст)

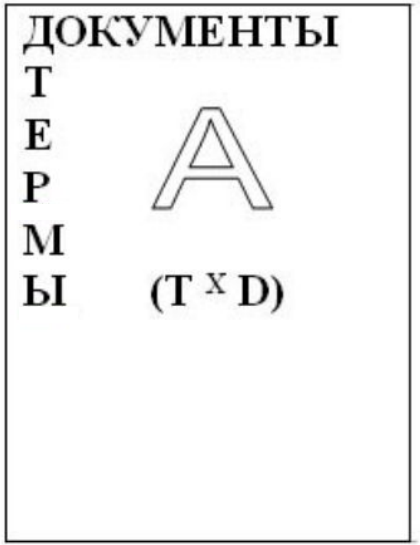




- Кроме пары (пользователь, фильм) на оценку может влиять некоторый контекст. Например, время суток, в которое ставится оценка.
- В этом случае нам надо будет иметь дело с многомерными массивами - **тензорами**
- Тоже можно оптимизировать ошибку, аналогами скелетного разложения в многомерном случае будут разложения Таккера и tensor-train разложение

LSA (Латентно-семантический анализ)

- Пусть у нас есть множество текстов (документов)
- Мы хотим разбить их на темы, а также научиться определять темы новых документов
- Рассмотрим матрицу, строки которой будут соответствовать некоторым термам (частые слова, N-граммы и т.п.), а столбцы будут соответствовать документам
- В ячейки этой матрицы запишем некоторую характеристику, например количество вхождений терма в документ (могут быть более сложные вероятностные показатели)
- Матрица получится разреженной
- Каждый документ теперь это вектор - столбец в этой матрице



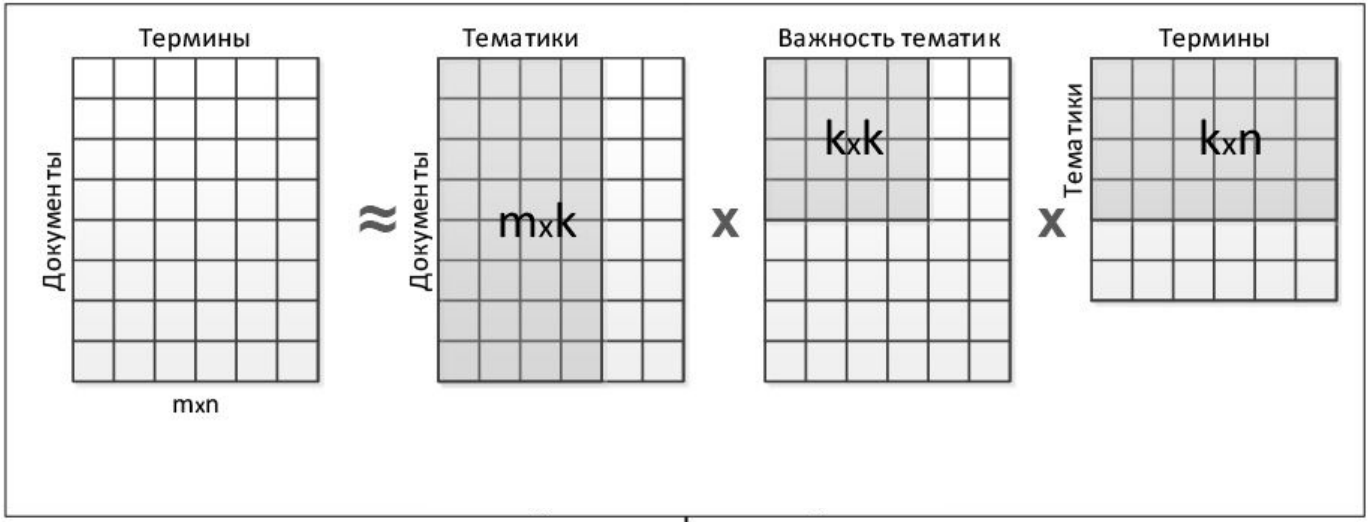
LSA + PCA

- Можно сделать PCA векторов документов (столбцов матрицы) на сильно меньшую размерность
- Близкие точки будут соответствовать близким по теме документам
- С помощью кластеризации/метода ближайших соседей можем разбить документы на темы и находить похожие по темам документы

- Аналогично можно сделать PCA векторов термов (строк матрицы) на сильно меньшую размерность
- Близкие точки будут соответствовать термам, которые встречаются в одном контексте
- С помощью кластеризации можем разбить термы на контексты

LSA + SVD

- Сделаем SVD разложение построенной матрицы, при этом выберем небольшой ранг k
- Тогда всем коэффициентам разложения можно придать смысл
- Можно использовать эту информацию



На самом деле из понимания того, как работает PCA мы получаем тоже самое, что на предыдущем слайде, но такой взгляд иногда может быть удобнее.

Заключение

- SVD дает наиболее точное малоранговое приближение матриц
- Матрица маленького ранга требует меньше памяти, её удобно представлять в виде скелетного разложения
- С помощью PCA можно описывать структуру множества точек и уменьшать их размерность, оставляя проекции на нужное число главных компонент
- Используя идею PCA можно построить метод распознавания лиц Eigenfaces
- Можно построить рекомендательную систему на основе малорангового приближения матрицы, состоящей из уже известных оценок
- Можно проводить анализ документов и текстов, выделять тематики и делать разбиение по темам на основе PCA и малоранговых разложений матрицы (Термы x Документы)

Главный вывод: с помощью SVD можно выделять скрытые главные признаки из векторных данных большого размера. Это можно применять практически в любой области

Источники

- Пример со сжатием изображения: https://askdev.ru/q/ispolzovanie-svd-dlya-szhatiya-izobrazheniya-v-matlab-219826/
- PCA, wikipedia: https://ru.wikipedia.org/wiki/Метод_главных_компонент
- PCA:https://medium.com/@jonathan_hui/machine-learning-singular-value-decomposition-svd-principal-component-analysis-pca-1d45e885e491
- Eigenfaces, wikipedia: https://en.wikipedia.org/wiki/Eigenface
- Eigenfaces, статья на habr: https://habr.com/ru/post/68870/
- Расстояния в eigenfaces: https://www.cs.colostate.edu/evalfacerec/papers/eemcvcsu.pdf
- Рекомендательные системы, Михаил Ройзнер (Яндекс): https://habr.com/ru/company/yandex/blog/241455/
- Рекомендательные системы, другая статья на habr: https://habr.com/ru/company/surfingbird/blog/140555/
- Рекомендательные системы: https://aspirantura.hse.ru/data/2016/06/11/1117726495/2016-06-09-khalkechev.pdf
- LCA, wikipedia: https://ru.wikipedia.org/wiki/Латентно-семантический_анализ
- LCA, статья на habr: https://habr.com/ru/post/110078/
- Topic Modeling: https://linis.hse.ru/data/2018/10/11/1155920858/TopicModeling.pdf