Лекция 3

1. Лекция 3: SVD-разложение и его применение

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

Ранее мы с вами рассмотрели QR-разложение и очень кратко пробежались по двум простейшим методам его получения, а также по областям применения самого разложения.

Для дальнейшего разговора предположим, что мы имеем матрицу $X_{m\times n}$ ранга p. Очевидно, что $p \leqslant \min(m, n)$.

Будем рассматривать эту матрицу как **матрицу данных**, в которой

- каждый столбец — то индивид или момент времени, зависит от задачи;
- каждая строка — атрибут или признак индивида или момента времени.

Мы сконцентрируемся на двух ситуациях:

1. **Низкая и толстая** матрица X, то есть $n \ll m$;
2. **Высокая и стройная** матрица X, то есть $n \gg m$.

В первом случае SVD-разложение приводит к методу **главных компонент**. Во втором — к своеобразному виду МНК.

1.1. SVD-разложение

SVD-разложением матрицы $X_{n\times m}$ ранга $p \leqslant \min(n, m)$ называется разложение вида

$$X = U\Sigma V^*$$

где

$$\begin{aligned} UU^* = I \quad VV^* = I \\ U^*U = I \quad V^*V = I \end{aligned}$$

- $U_{n\times n}$ — ортогональная матрица **левых сингулярных векторов**. Столбцы U — собственные векторы XX^*.
- $V_{m\times m}$ — ортогональная матрица **правых сингулярных векторов**. Столбцы V — собственные векторы X^*X.
- $\Sigma_{n\times m}$ — матрица, на главной диагонали которой первые p значений являются положительными числами $\sigma_1, \dots, \sigma_p$, а остальные равны нулю. Эти **сингулярные значения** равны квадратным корням собственных значений XX^* и X^*X.

- $\square^*$ — эрмитово сопряжение, представляющее собой транспонирование и замену чисел на сопряженные. Очевидно, что для действительной матрицы $X^* \equiv X'$.

U и V — матрицы собственных векторов, так как они удовлетворяют **разложению по собственным векторам**

$$X = Q\Lambda Q^{-1}$$

где Q — матрица собственных векторов, Λ — матрица, на главной диагонали которой находятся собственные значения. Если X является положительно определенной, то $Q^{-1} = Q'$.

Имеем

$$XX^* = U\Sigma V^* V\Sigma^* U^* = U\Sigma^2 U^*$$

$$X^*X = V\Sigma^* U^* U\Sigma V^* = V\Sigma^2 V^*$$

где $\Lambda = \Sigma^2$.

Это разложение имеет следующую интерпретацию:

1. Умножение вектора на U и V приводит к его повороту, сохраняющему длины и углы.
2. Умножение вектора на Σ приводит к сохранению углов между векторами, но меняет масштаб.

Описанная выше процедура — это **полное** SVD-разложение.

1.2. Теорема Эккарта-Янга

Предположим, что мы хотим найти матрицу X_r ранга $r < p$, наилучшим образом «приближающую» нашу матрицу X.

Наилучшей считается матрица, которая является решением

$$\min_{X_r} \|X - X_r\|$$

где $\|\cdot\|$ — какая-либо норма матрицы, например

- **спектральная** или l^2: $\|X\|_2 = \max_{\|y\|\neq 0} \dfrac{\|Xy\|}{\|y\|} = \sigma_1$,
- **Фробениуса**: $\|X\|_F = \sqrt{\sigma_1^2 + \cdots + \sigma_p^2}$,
- **ядерная**: $\|X\|_N = \sigma_1 + \cdots + \sigma_p$.

Теорема Эккарта-Янга говорит, что для **любой** из этих трех норм наилучшей матрицей ранга r будет матрица

$$X_r = \sigma_1 u_1 v_1^* + \sigma_2 u_2 v_2^* + \cdots + \sigma_r u_r v_r^*$$

где u_i, v_i — i-й столбцы матриц U и V, соответственно.

Собственно, метод главных компонент очень тесно связан с этим выражением: если какая-то часть сингулярных значений несет основную часть информации, то мы можем оставить только соотвествующие им слагаемые и получить наилучшее приближение, несущую нужную нам информацию.

1.3. Сокращенное SVD-разложение

Сокращенное SVD-разложение имеет тот же вид, что и обычное, но итоговые матрицы имеют другие размеры:

$$U_{n\times p}, \quad \Sigma_{p\times p}, \quad V_{m\times p}$$

Также имеем следующее:

1. Для **низкого и толстого** варианта $V^*V \neq I$,
2. Для **высокого и стройного** варианта $UU^* \neq I$.

Такой вид связан с тем, что матрица Σ имеет только p ненулевых сингулярных значений.

1.3.1. Пришло время посмотреть на код

Сгенерируем случайную матрицу $X_{5\times 2}$. Каждое наблюдение выбирается из равномерного распределения.

[2]:
```
X = np.random.rand(5, 2)
X
```

[2]:
```
array([[0.0763592 , 0.52882701],
       [0.40634323, 0.20404434],
       [0.61626948, 0.40681486],
       [0.56614903, 0.61600772],
       [0.49424945, 0.10199734]])
```

Посмотрим на полное SVD-разложение.

[3]:
```
U, S, V = np.linalg.svd(X, full_matrices=True)
U, S, V
```

[3]:
```
(array([[-0.30157932,  0.76082963, -0.18075628, -0.5154355 ,
         0.17844215],
        [-0.33090934, -0.23701457, -0.5432443 , -0.20939324,
        -0.70382044],
        [-0.54938255, -0.19960325,  0.74108479, -0.28868059,
        -0.16060578],
        [-0.62295346,  0.21444933, -0.098568  ,  0.74191996,
         0.07602363],
        [-0.33114458, -0.52832982, -0.33658652, -0.23811552,
         0.66424497]]),
 array([1.33358024, 0.46234622]),
 array([[-0.75916793, -0.65089481],
        [-0.65089481,  0.75916793]]))
```

Обратите внимание, что матрица Σ представлена в виде массива, содержащего сингулярные значения. Все равно остальные элементы матрицы $\Sigma_{5\times 2}$ равны нулю.

[4]:
```
Us, Ss, Vs = np.linalg.svd(X, full_matrices=False)
Us, Ss, Vs
```

```
[4]: (array([[-0.30157932,  0.76082963],
             [-0.33090934, -0.23701457],
             [-0.54938255, -0.19960325],
             [-0.62295346,  0.21444933],
             [-0.33114458, -0.52832982]]),
      array([1.33358024, 0.46234622]),
      array([[-0.75916793, -0.65089481],
             [-0.65089481,  0.75916793]]))
```

Обратите внимание, как «обрезана» матрица U. Также $UU^* \neq I$.

```
[5]: np.isclose(U @ U.T, np.eye(5))
```

```
[5]: array([[ True,  True,  True,  True,  True],
            [ True,  True,  True,  True,  True],
            [ True,  True,  True,  True,  True],
            [ True,  True,  True,  True,  True],
            [ True,  True,  True,  True,  True]])
```

```
[6]: np.isclose(U.T @ U, np.eye(5))
```

```
[6]: array([[ True,  True,  True,  True,  True],
            [ True,  True,  True,  True,  True],
            [ True,  True,  True,  True,  True],
            [ True,  True,  True,  True,  True],
            [ True,  True,  True,  True,  True]])
```

Но...

```
[7]: np.isclose(Us @ Us.T, np.eye(5))
```

```
[7]: array([[False, False, False, False, False],
            [False, False, False, False, False],
            [False, False, False, False, False],
            [False, False, False, False, False],
            [False, False, False, False, False]])
```

```
[8]: np.isclose(Us.T @ Us, np.eye(2))
```

```
[8]: array([[ True,  True],
            [ True,  True]])
```

1.4. Метод главных компонент

1.4.1. QR

Сначала рассмотрим, как QR-разложение может быть применено в контексте метода главных компонент (PCA).

PCA-разложение можно описать в матричном виде как $T = XW$, где W — матрица собственных векторов матрицы $X'X$, расположенных в порядке убывания соб-

ственных значений. То есть имеем следующее

$$X'X = W\hat{\Lambda}W'$$

где $\hat{\Lambda}$ — матрица собственных значений.

Рассмотрим первую главную компоненту. По определению, первая главная компонента отлавливает направление с максимальным разбросом данных вдоль себя

$$w_1 = \arg\max_{\|w\|=1}(\|Xw\|^2) = \arg\max_{w}\frac{w'X'Xw}{w'w}$$

Последнее выражение является отношением Рэлея и принимает максимальное значение, равное максимальному собственному значению $X'X$. Из этого следует, что w_1 — собственный вектор.

Последующие значения w_k находятся вычитанием из X первых $k-1$ главных компонент

$$\hat{X}_k = X - \sum_{i=1}^{k-1} Xw_i w_i'$$

и решением описанной выше задачи. Оказывается, что решением будут соответствующие собственные значения $X'X$.

Рассмотрим QR-разложение матрицы X': $X' = QR$. Найдем собственные значения RR': $RR' = \tilde{P}\tilde{\Lambda}\tilde{P}'$.

Очевидно, что $X'X = QRR'Q' = Q\tilde{P}\tilde{\Lambda}\tilde{P}'Q'$.

Оказывается, что $\hat{\Lambda} = \tilde{\Lambda}$, и, соответственно, $W = Q\tilde{P}$.

[9]:
```
k = 5
n = 1000

mu = np.random.random(size=k)
C = np.random.random((k, k))
Sigma = C.T @ C

X = np.random.multivariate_normal(mu, Sigma, size=n)
X.shape
```

[9]:
```
(1000, 5)
```

[10]:
```
Q, R = np.linalg.qr(X.T)
Q.shape, R.shape
```

[10]:
```
((5, 5), (5, 1000))
```

[11]:
```
L, P_tilde = np.linalg.eig(R @ R.T)
```

[12]:
```
L2, P = np.linalg.eig(X.T @ X)
```

[13]:
```
L
```

```
[13]: array([6452.61206105,  103.9831228 ,  430.98543786,  231.
        ↪8905333 ,
              332.25406838])
```

```
[14]: np.isclose(L, L2).all()
```

```
[14]: np.True_
```

```
[15]: np.isclose(np.abs(P), np.abs(Q @ P_tilde)).all()
```

```
[15]: np.True_
```

```
[16]: np.isclose(Q @ P_tilde @ np.diag(L) @ P_tilde.T @ Q.T, X.T␣
        ↪@ X).all()
```

```
[16]: np.True_
```

1.4.2. SVD

Рассмотрим случай $n \gg m$ (высокий и стройный).

Обозначим матрицу X как

$$X = [X_1 \mid X_2 \mid \cdots \mid X_m]$$

$$X_j = \begin{bmatrix} x_{1j} \\ x_{2j} \\ \vdots \\ x_{nj} \end{bmatrix}$$

1. Стандартизуем данные. Найдем средние по столбцам

$$\bar{X}_j = \frac{1}{n} \sum_{i=1}^{n} x_{ij}$$

Построим матрицу средних

$$\bar{X} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} \bar{X}_1 & \bar{X}_2 & \dots & \bar{X}_n \end{bmatrix}$$

Отцентрируем X

$$B = X - \bar{X}$$

2. Найдем ковариационную матрицу

$$C = \frac{1}{n-1} B^* B$$

3. Найдем SVD матрицы B и отсортируем сингулярные значения по убыванию. Имеем

$$B^*B = V\Sigma^* U^* U \Sigma V^* = V\Sigma^* \Sigma V^* = V\Sigma^2 V^*$$

Опционально можем отбросить значения, дающие слишком мало информации, ориентируясь на отношение

$$\frac{\sum_{i=1}^{r} \sigma_i^2}{\sum_{i=1}^{p} \sigma_i^2}$$

4. Строим итоговую матрицу

$$T = BV = U\Sigma V^* V = U\Sigma$$

Если вспомнить теорему Эккарта-Янга, то понятно, что

- $v_k,\ \forall k = \overline{1, m}$ — k-й столбец V — это веса признаков в главной компоненте.
- $u_k = \begin{bmatrix} u_{1k} \\ \vdots \\ u_{nk} \end{bmatrix},\ \forall k = \overline{1, n}$ — временной ряд k-й главной компоненты.
- $\sigma_k,\ \forall k = \overline{1, p}$ — мощность k-й главной компоненты.

1.4.3. Связь собственных значений и SVD

Имеем разложение ковариационной матрицы на собственные векторы и матрицу собственных значений

$$X^* X = P\Lambda P^*$$

Пусть

$$\epsilon^* \epsilon = \Lambda$$

Тогда

$$X = \epsilon P^*$$

и

$$\epsilon = XP$$

Имеем SVD-разложение

$$X = U\Sigma V^*$$

Рассчитаем

$$\begin{aligned} X^* X &= V\Sigma^* U^* U\Sigma V^* = \\ &= V\Sigma\Sigma^* V^* = \\ &= V\Lambda V^* \end{aligned}$$

Очевидно, что $P = V$. Тогда по определению PCA-разложения

$$T = XV = U\Sigma = \epsilon$$

$$\epsilon^* \epsilon = \Sigma U^* U\Sigma = \Sigma^2 = \Lambda$$