Introduction to Data Science

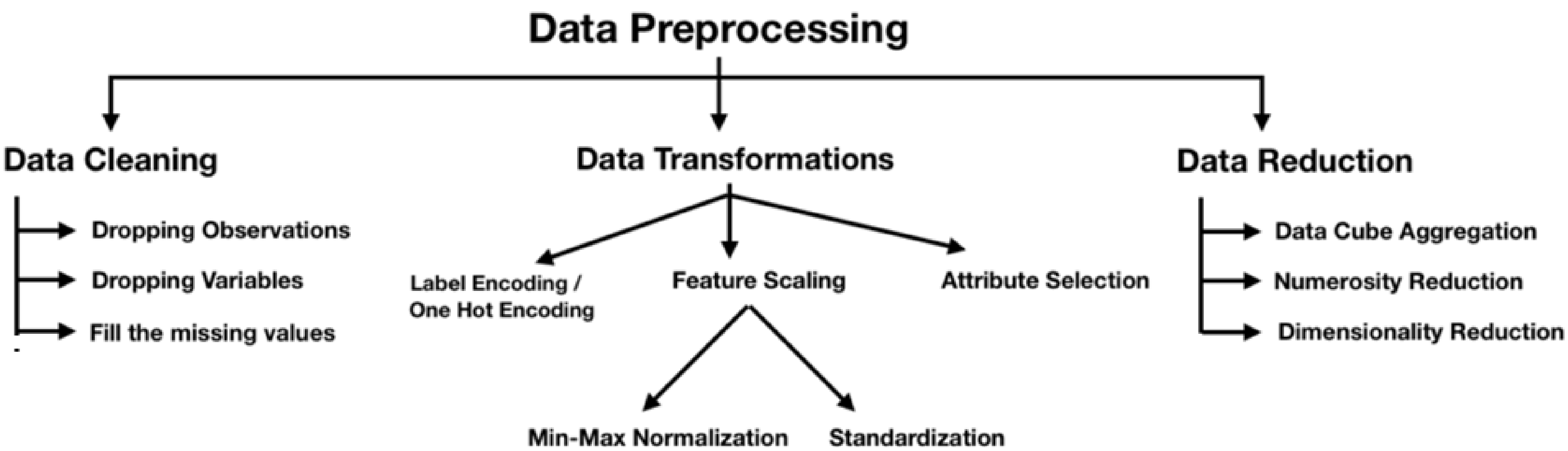
Dr. Irfan Yousuf
Department of Computer Science (New Campus)
UET, Lahore

(Week 14; April 14 - 18, 2025)

Outline

- Eigenvectors
- Dimensionality Reduction
    - Principle Component Analysis
    - Singular Value Decomposition

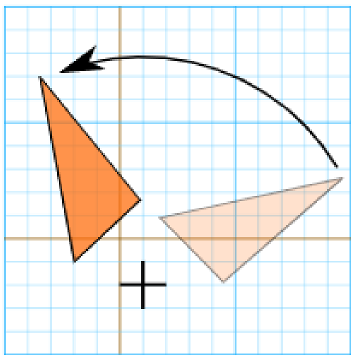
Data Preprocessing



Transformation

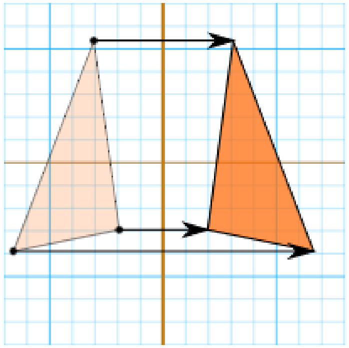
- In geometry, transformation refers to the **movement of objects in the coordinate plane**.
- In mathematics, a transformation is a function f (usually with some geometrical meanings) that maps a set X to itself.

Rotation



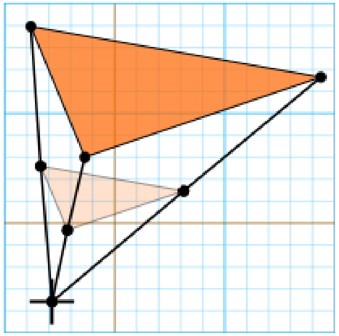
Turn!

Reflection



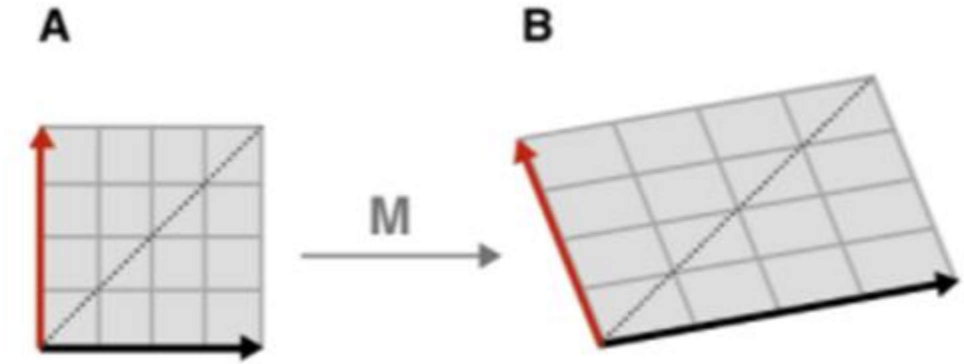
Flip!

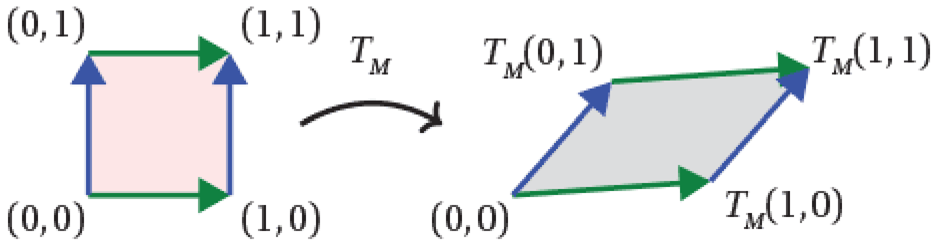
Resizing



Linear Transformation

- A **linear transformation** is one in which a straight line before the transformation results in a straight line after the transformation.



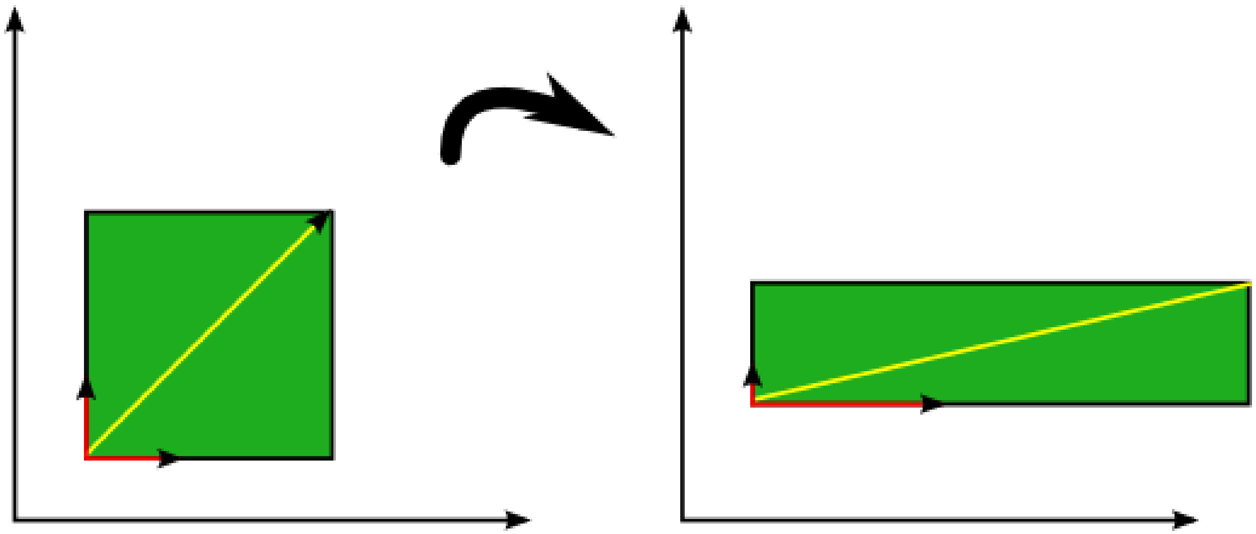


Eigenvectors and Eigenvalues

- In linear algebra, an **eigenvector or characteristic vector** of a linear transformation is a nonzero vector that **changes at most by a scalar factor** when that linear transformation is applied to it.

- The corresponding **eigenvalue**, often denoted by λ (Lambda) is the factor by which the eigenvector is scaled.

Eigenvectors and Eigenvalues



- An eigenvector is a vector whose direction remains unchanged when a linear transformation is applied to it. (Eigen means **Specific** in German)
- The transformation in this case is a simple scaling with factor 2 in the horizontal direction and factor 0.5 in the vertical direction, such that the transformation matrix A is defined as:

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix}$$

Eigenvectors and Eigenvalues

- **Matrix**: A rectangular array of numbers.
- **Vector**: A matrix with a single column.
- **Eigenvector**: Given a square matrix A, a vector v is an eigenvector of A if multiplying A by v results in a scaled version of v. In other words, the direction of v doesn't change. Mathematically:
  $$A \cdot v = \lambda \cdot v$$
  where λ is a scalar.
- **Eigenvalue**: The scalar λ is the eigenvalue corresponding to the eigenvector v.

- An **eigenvector** defines a *direction* in which a space is scaled by a transform.
- An **eigenvalue** defines a *length* of scaled change related to the eigenvector.

Eigenvectors and Eigenvalues

In general, the eigenvector $\vec{v}$ of a matrix A is the vector for which the following holds:

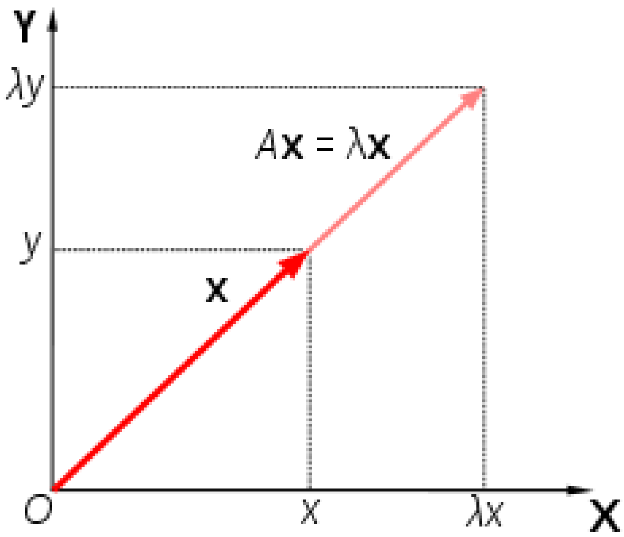
$$A\vec{v} = \lambda\vec{v}$$

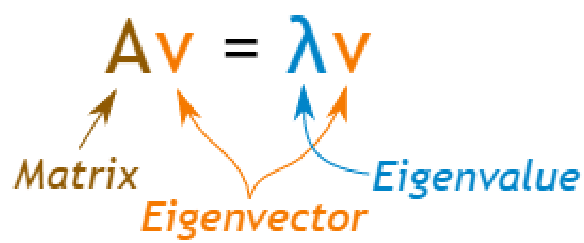
where λ is a scalar value called the 'eigenvalue'. This means that the linear transformation A on vector $\vec{v}$ is completely defined by λ.

$$\begin{aligned} A\vec{v} - \lambda\vec{v} &= 0 \\ \Rightarrow \vec{v}(A - \lambda I) &= 0, \end{aligned}$$

where I is the identity matrix of the same dimensions as A.

Eigenvectors and Eigenvalues





Example: For this matrix

$$\begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix}$$

an eigenvector is:

$$\begin{bmatrix} 1 \\ 4 \end{bmatrix}$$

with a matching eigenvalue of 6

Av gives us:

$$\begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix}\begin{bmatrix} 1 \\ 4 \end{bmatrix} = \begin{bmatrix} -6\times1+3\times4 \\ 4\times1+5\times4 \end{bmatrix} = \begin{bmatrix} 6 \\ 24 \end{bmatrix}$$

λv gives us :

$$6\begin{bmatrix} 1 \\ 4 \end{bmatrix} = \begin{bmatrix} 6 \\ 24 \end{bmatrix}$$

Eigenvectors and Eigenvalues

Start with $| A - \lambda I | = 0$

$$\left| \begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right| = 0$$

Which is:

$$\begin{vmatrix} -6-\lambda & 3 \\ 4 & 5-\lambda \end{vmatrix} = 0$$

Calculating that determinant gets:

$$(-6-\lambda)(5-\lambda) - 3\times 4 = 0$$

Which then gets us this Quadratic Equation:

$$\lambda^2 + \lambda - 42 = 0$$

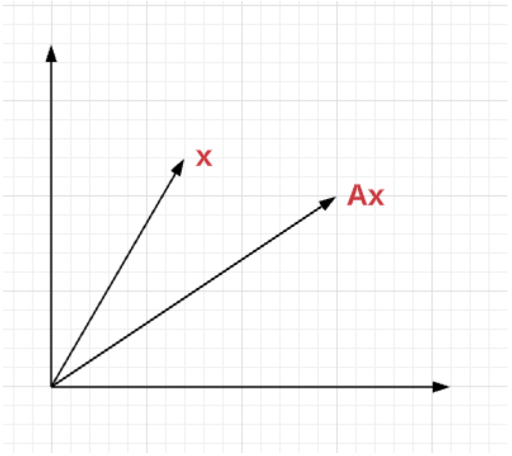
And solving it gets:

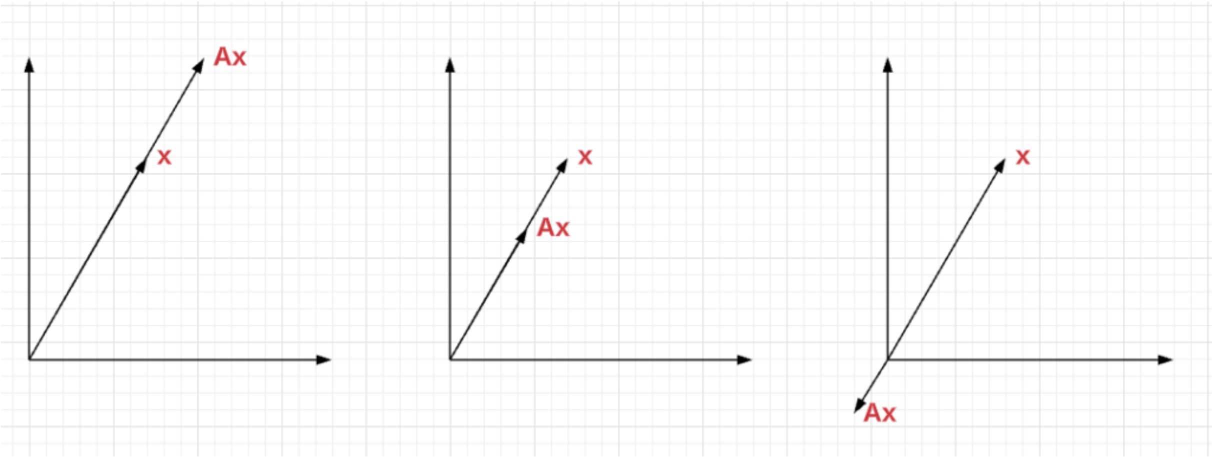
$$\lambda = -7 \text{ or } 6$$

And yes, there are **two** possible eigenvalues.

Eigenvectors and Eigenvalues

- When the matrix multiplication with vector results in another vector in the same/opposite direction but scaled in forward / reverse direction by a magnitude of scaler multiple (eigenvalue) then the vector is called the eigenvector of that matrix.
- The diagram representing the eigenvector x of matrix A because the vector Ax is in the same/opposite direction of x.





Eigenvectors in Data Science

- The concept of Eigenvectors and Eigenvalues is used to determine **a set of important variables** (in form of vector) along with **scale along different dimensions** (key dimensions based on variance) for analyzing the data in a better manner.

When you look at the picture (data) and identify it as tiger, what are some of the key information (dimensions / principal components) you use to call it out as tiger? Is it not body, face, legs etc. information?

Eigenvectors in Data Science

- Can we use the information stored in variables and **extract a smaller set of variables** (features) to train the models and do the prediction while ensuring that most of the information contained in the original variables is retained / maintained.

Dimensionality Reduction

- Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining **a set of principal variables**.
- The **number of input variables or features** for a dataset is referred to as its **dimensionality**.
- More input features often make a predictive modeling task more challenging to model.
- Sometimes, most of these features are correlated, and hence redundant

Dimensionality Reduction Methods

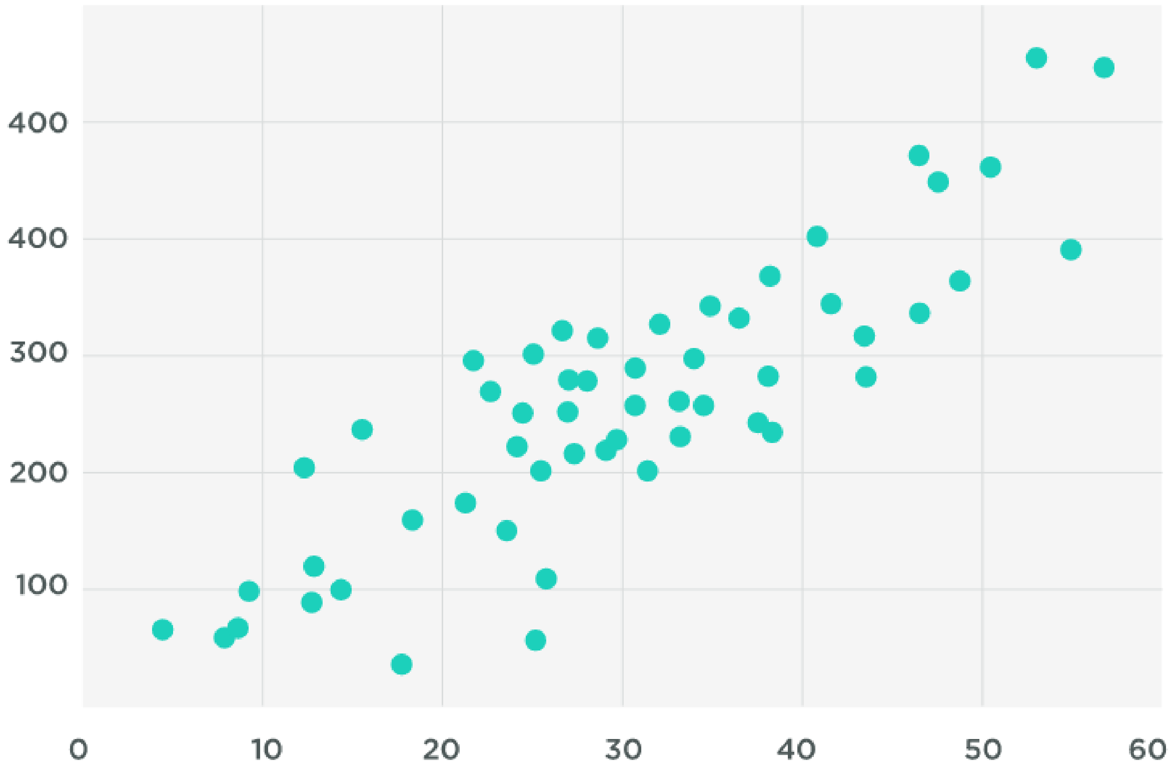
- The various methods used for dimensionality reduction include:
  - Principal Component Analysis
  - Singular Value Decomposition

Principal Component Analysis

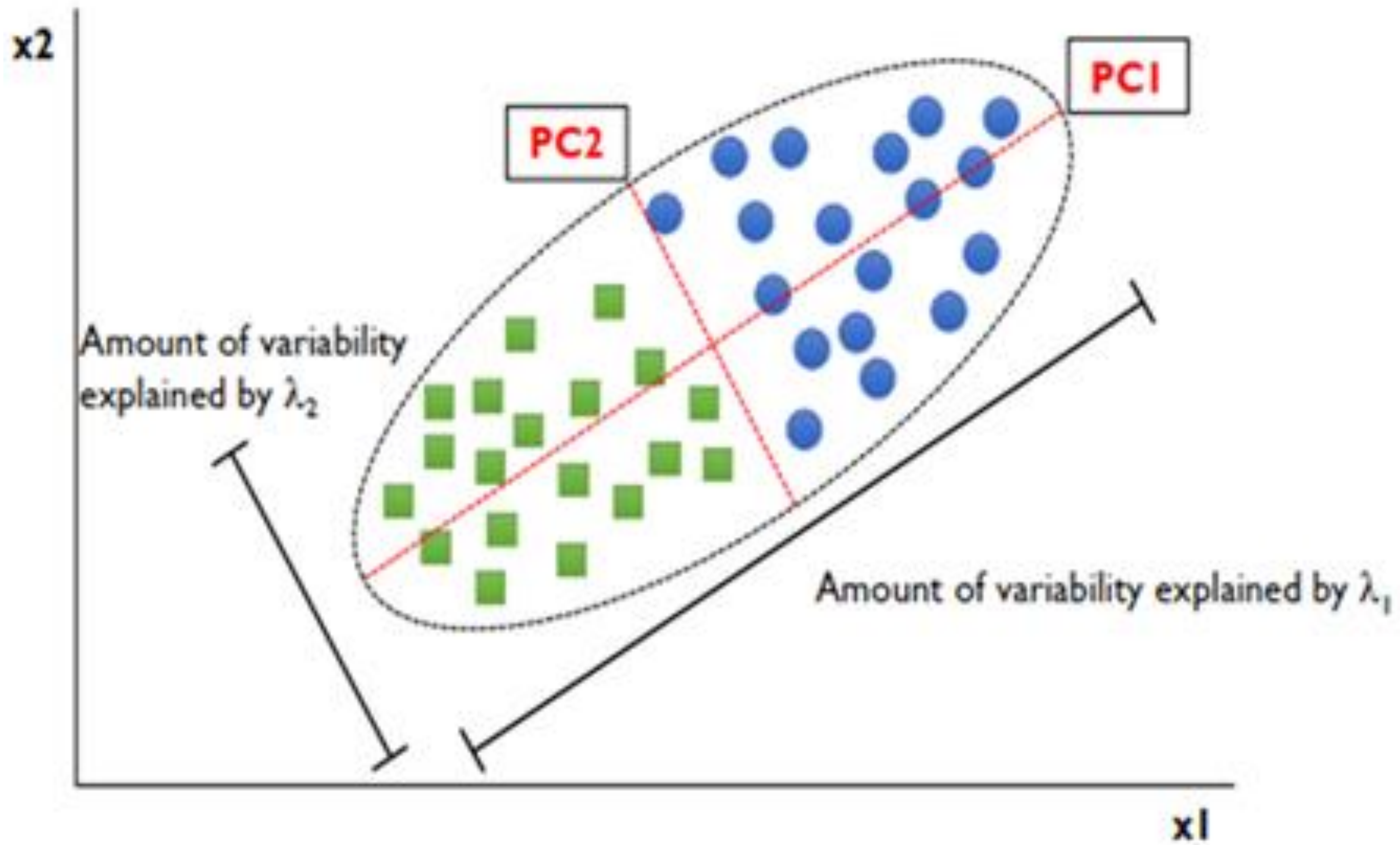
- Principal Component Analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by **transforming** a large set of variables into a smaller one that still contains most of the information of the large set.
- Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity.
- So, to sum up, the idea of PCA is simple — **reduce the number of variables of a data set**, while preserving as much information as possible.

Principal Component Analysis

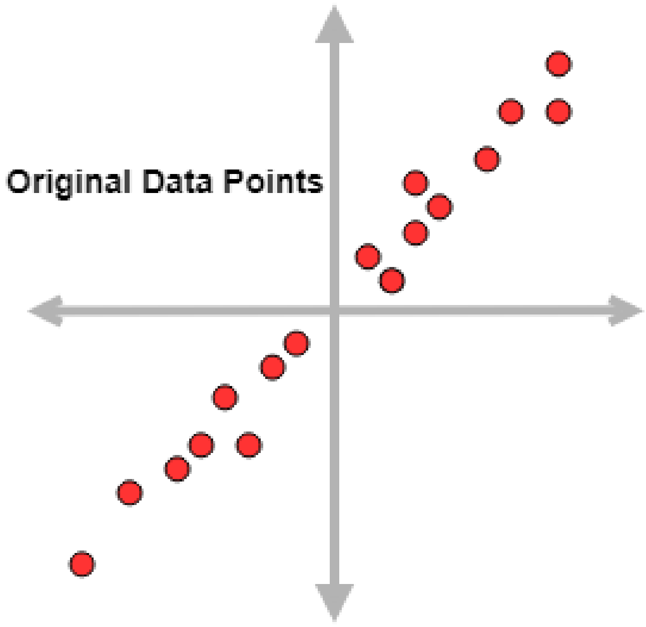
In what direction we can find more information?

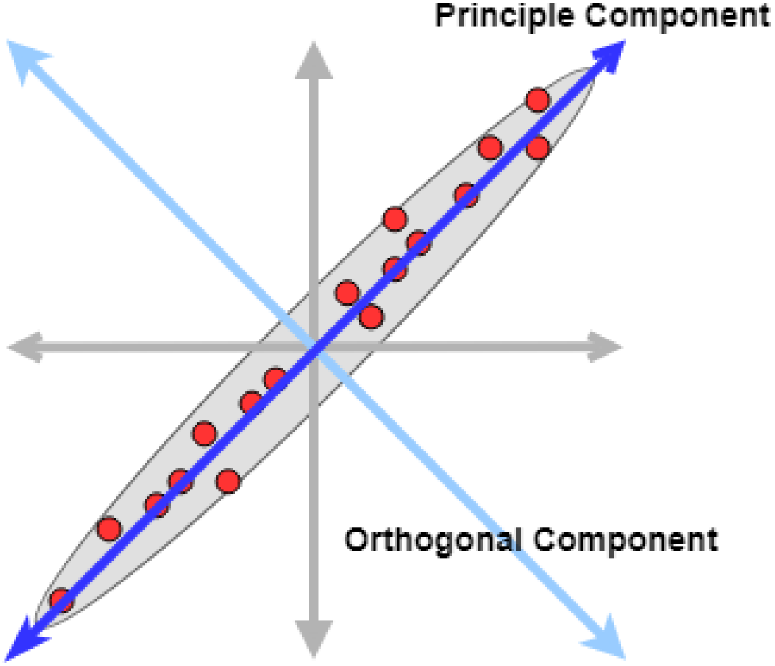


Principal Component Analysis



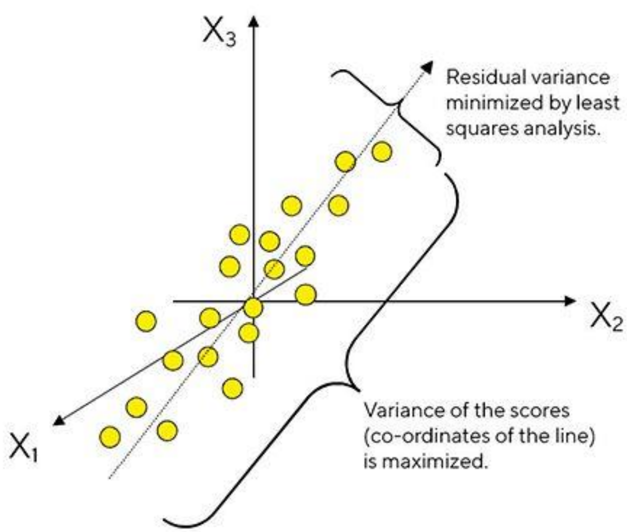
Principal Component Analysis





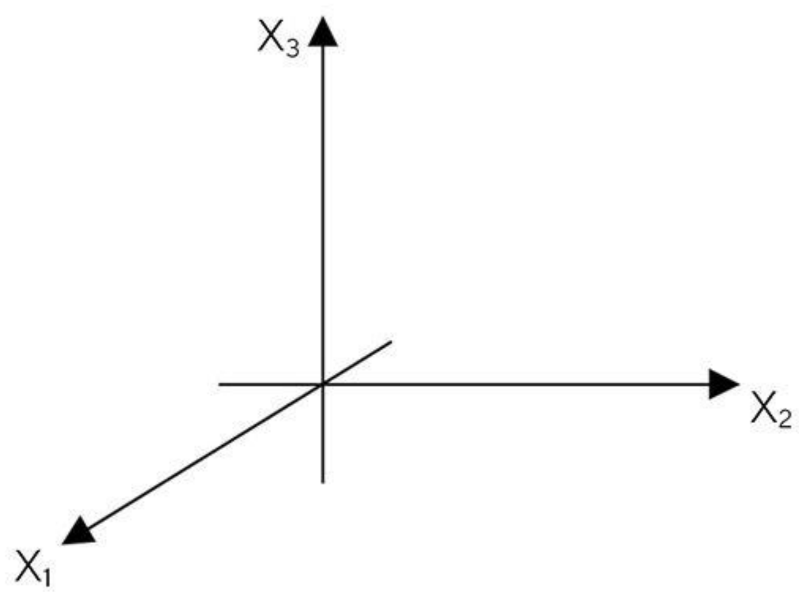
Principal Component Analysis

- Statistically, PCA finds lines, planes and hyper-planes in the K-dimensional space that approximate the data as well as possible in the least squares sense.
- A line or plane that is the least squares approximation of a set of data points makes the **variance of the coordinates** on the line or plane as large as possible.
- Eigenvectors of a matrix are **directions of maximum spread** or variance of data



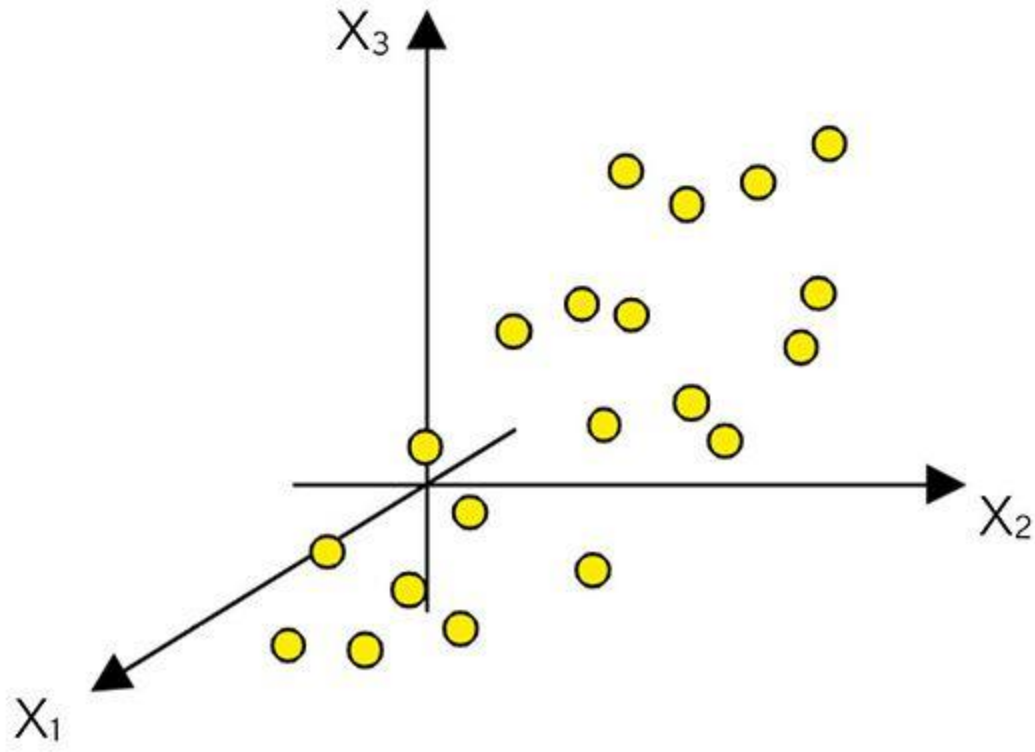
How PCA Works?

- Consider a matrix X with N rows (aka "observations") and K columns (aka "variables").
- For this matrix, we construct a variable space with as many dimensions as there are variables, i.e., K.
- Each variable represents one coordinate axis.



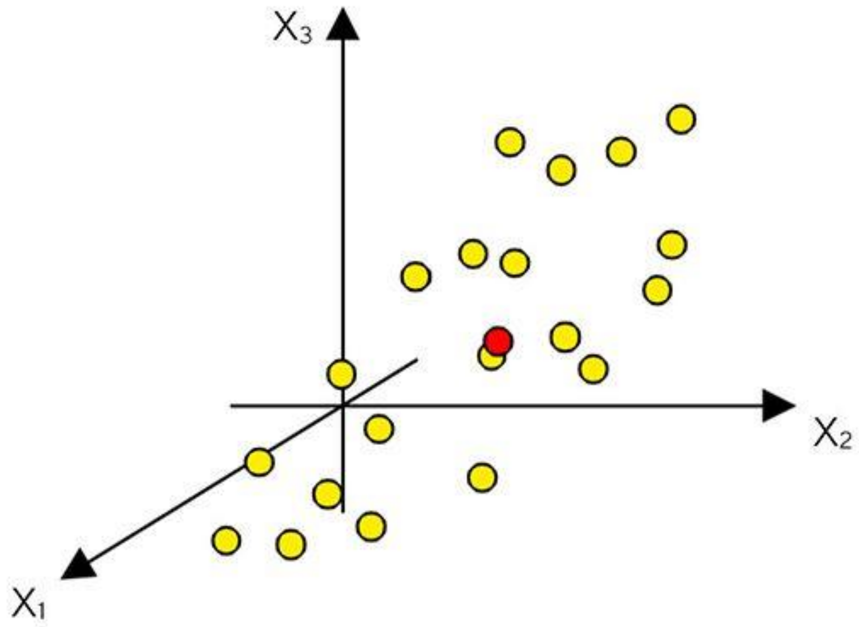
How PCA Works?

- In the next step, each observation (row) of the X-matrix is placed in the K-dimensional variable space.



How PCA Works?

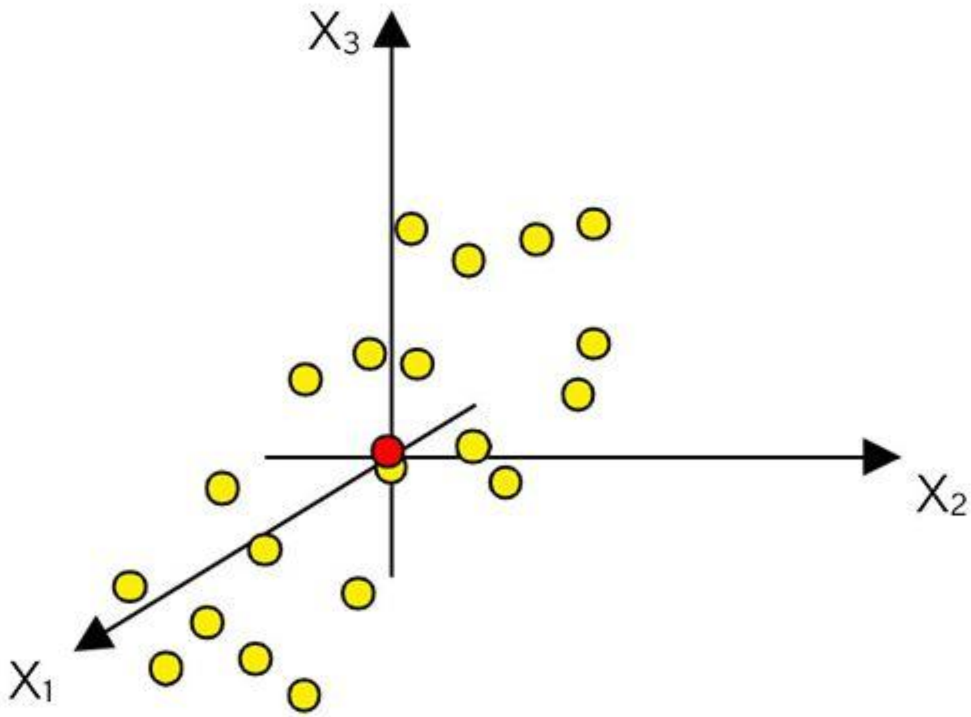
- Next, mean-centering involves the subtraction of the variable averages from the data. The vector of averages corresponds to a point in the K-space.



In the **mean-centering procedure**, we first compute the variable averages. This vector of averages is interpretable as a point (here in red) in space.

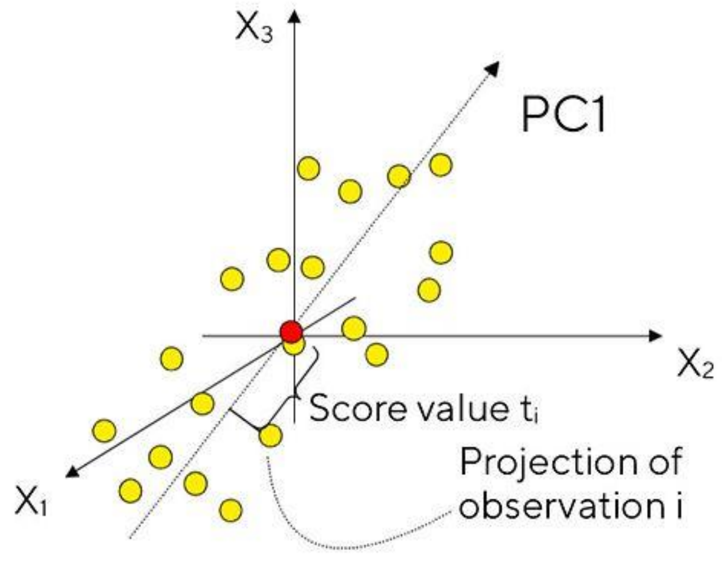
How PCA Works?

- The subtraction of the averages from the data corresponds to a re-positioning of the coordinate system, such that the average point now is the origin.



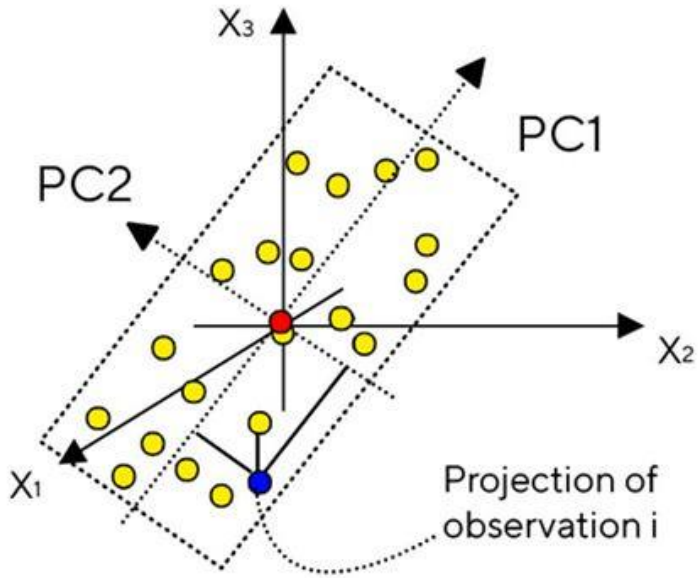
How PCA Works?

- The **first principal component (PC1)** is the line in the K-dimensional variable space that best approximates the data in the least squares sense.
- This **line goes through the average point**. Each observation (yellow dot) may now be projected onto this line in order to get a coordinate value along the PC-line. This new coordinate value is also known as the score.



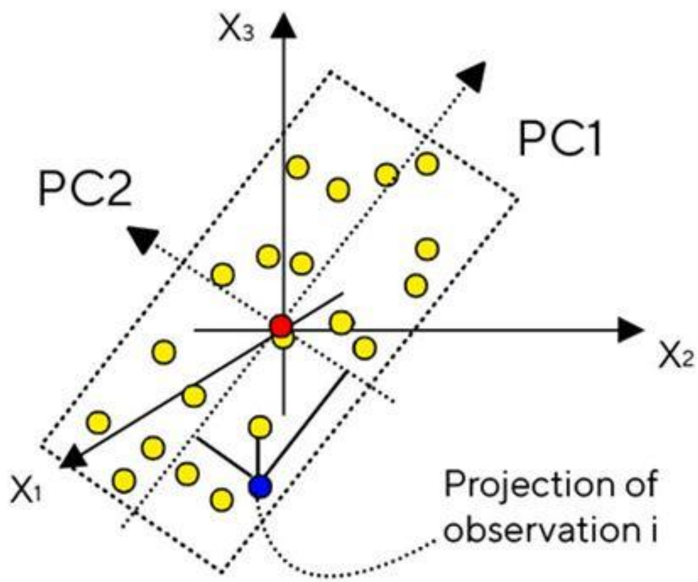
How PCA Works?

- One principal component is insufficient to model the systematic variation of a data set. Thus, a second **principal component (PC2)** is calculated.
- PC2 is also represented by a line in the K-dimensional variable space, which is **orthogonal to the first PC (i.e., PC1)**. This line also passes through the average point, and improves the approximation of the X-data as much as possible

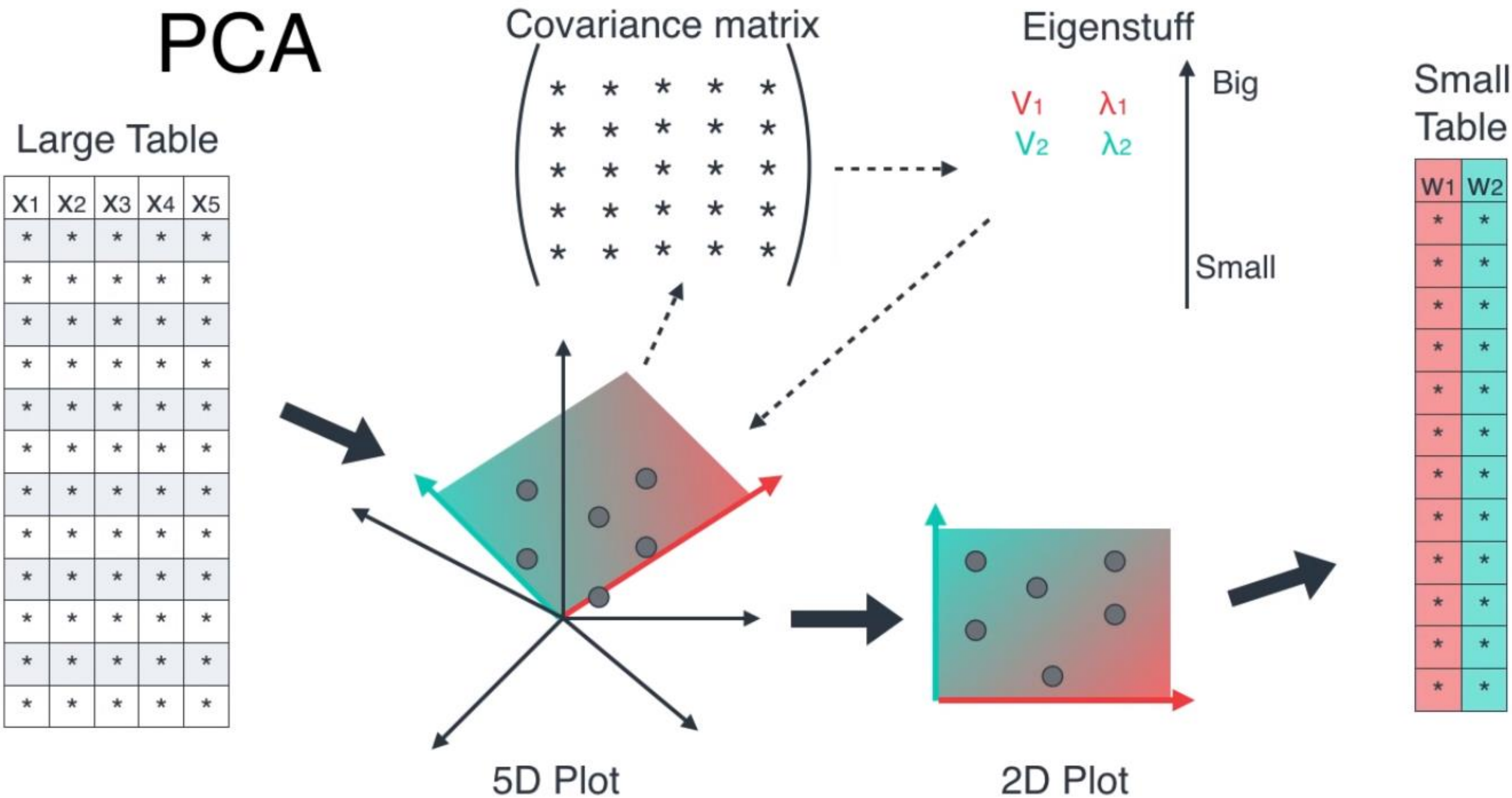


How PCA Works?

- Two PCs form a plane. This plane is a window into the multidimensional space, which can be visualized graphically. Each observation may be projected onto this plane, giving a score for each.



Principal Component Analysis



PCA Working Example

- Standardize the dataset.
- Calculate the **covariance matrix** for the features in the dataset.
- Calculate the **eigenvalues and eigenvectors** for the covariance matrix.
- Sort eigenvalues and their corresponding eigenvectors.
- Pick k eigenvalues and form a matrix of eigenvectors.
- Transform the original matrix.

PCA Working Example: Data

- Dataset with 4 features and 5 observations

f1	f2	f3	f4
1	2	3	4
5	5	6	7
1	4	2	3
5	3	2	1
8	1	2	2

https://medium.com/analytics-vidhya/understanding-principle-component-analysis-pca-step-by-step-e7a4bb4031d9

PCA Working Example: Step 1

- First, we need to standardize the dataset and for that, we need to calculate the mean and standard deviation for each feature.

$$x_{new} = \frac{x - \mu}{\sigma}$$

f1	f2	f3	f4
1	2	3	4
5	5	6	7
1	4	2	3
5	3	2	1
8	1	2	2

	f1	f2	f3	f4
μ =	4	3	3	3.4
σ =	3	1.58114	1.73205	2.30217

f1	f2	f3	f4
-1	-0.63246	0	0.26062
0.33333	1.26491	1.73205	1.56374
-1	0.63246	-0.57735	-0.17375
0.33333	0	-0.57735	-1.04249
1.33333	-1.26491	-0.57735	-0.60812

PCA Working Example: Step 2

- Calculate the covariance matrix for the whole dataset.

- **Variance** is a measure of the variability or spread in a set of data.

$$\sigma^2 = \frac{\Sigma (X - \mu)^2}{N}$$

- **Covariance** is a measure of the **joint variability** of two random variables. It is a measure of the relationship between two random variables. The metric evaluates how much – to what extent – the variables change together.

$$\boxed{COV(X,Y) = \frac{\sum_{i=1}^{n} \left(X_i - \bar{X}\right)\left(Y_i - \bar{Y}\right)}{n-1}}$$

PCA Working Example: Step 2

- Calculate the covariance matrix for the whole dataset.

- The diagonal entries of the covariance matrix are the variances and the other entries are the covariances.

$$\begin{bmatrix} V_a & C_{a,b} & C_{a,c} & C_{a,d} & C_{a,e} \\ C_{a,b} & V_b & C_{b,c} & C_{b,d} & C_{b,e} \\ C_{a,c} & C_{b,c} & V_c & C_{c,d} & C_{c,e} \\ C_{a,d} & C_{b,d} & C_{c,d} & V_d & C_{d,e} \\ C_{a,e} & C_{b,e} & C_{c,e} & C_{d,e} & V_e \end{bmatrix}$$

PCA Working Example: Step 2

- Calculate the covariance matrix for the whole dataset.

	f1	f2	f3	f4
f1	var(f1)	cov(f1,f2)	cov(f1,f3)	cov(f1,f4)
f2	cov(f2,f1)	var(f2)	cov(f2,f3)	cov(f2,f4)
f3	cov(f3,f1)	cov(f3,f2)	var(f3)	cov(f3,f4)
f4	cov(f4,f1)	cov(f4,f2)	cov(f4,f3)	var(f4)

	f1	f2	f3	f4
f1	0.8	-0.25298	0.03849	-0.14479
f2	-0.25298	0.8	0.51121	0.4945
f3	0.03849	0.51121	0.8	0.75236
f4	-0.14479	0.4945	0.75236	0.8

PCA Working Example: Step 3

- Calculate eigenvalues and eigen vectors

	f1	f2	f3	f4
f1	0.8 - λ	-0.25298	0.03849	-0.14479
f2	-0.25298	0.8- λ	0.51121	0.4945
f3	0.03849	0.51121	0.8 - λ	0.75236
f4	-0.14479	0.4945	0.75236	0.8 - λ

A-λI = 0

$$\lambda = 2.51579324\,,\ 1.0652885\,,\ 0.39388704\,,\ 0.02503121$$

PCA Working Example: Step 3

- Calculate eigenvalues and eigen vectors

Solving the (A-λI)v = 0 equation for v vector with different λ values:

$$\begin{pmatrix} 0.800000-\lambda & -(0.252982) & 0.038490 & -(0.144791) \\ -(0.252982) & 0.800000-\lambda & 0.511208 & 0.494498 \\ 0.038490 & 0.511208 & 0.800000-\lambda & 0.752355 \\ -(0.144791) & 0.494498 & 0.752355 & 0.800000-\lambda \end{pmatrix} \times \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = 0$$

For λ = *2.51579324, solving the above equation using Cramer's rule, the values for v vector are*

v1 = 0.16195986

v2 = -0.52404813

v3 = -0.58589647

v4 = -0.59654663

PCA Working Example: Step 3

- Calculate eigenvalues and eigen vectors

$$\lambda = 2.51579324\,,\ 1.0652885\,,\ 0.39388704\,,\ 0.02503121$$

e1	e2	e3	e4
0.161960	-0.917059	-0.307071	0.196162
-0.524048	0.206922	-0.817319	0.120610
-0.585896	-0.320539	0.188250	-0.720099
-0.596547	-0.115935	0.449733	0.654547

PCA Working Example: Step 4

- Sort eigenvalues and their corresponding eigenvectors.

Already Sorted

$$\lambda = 2.51579324\,,\ 1.0652885\,,\ 0.39388704\,,\ 0.02503121$$

e1	e2	e3	e4
0.161960	-0.917059	-0.307071	0.196162
-0.524048	0.206922	-0.817319	0.120610
-0.585896	-0.320539	0.188250	-0.720099
-0.596547	-0.115935	0.449733	0.654547

PCA Working Example: Step 5

- Pick k (here k=2) eigenvalues and form a matrix of eigenvectors

e1	e2
0.161960	-0.917059
-0.524048	0.206922
-0.585896	-0.320539
-0.596547	-0.115935

PCA Working Example: Step 6

- Transform the original matrix.

Feature matrix * top k eigenvectors = Transformed Data

f1	f2	f3	f4
-1.000000	-0.632456	0.000000	0.260623
0.333333	1.264911	1.732051	1.563740
-1.000000	0.632456	-0.577350	-0.173749
0.333333	0.000000	-0.577350	-1.042493
1.333333	-1.264911	-0.577350	-0.608121

(5,4)

*

e1	e2
0.161960	-0.917059
-0.524048	0.206922
-0.585896	-0.320539
-0.596547	-0.115935

(4,2)

=

nf1	nf2
0.014003	0.755975
-2.556534	-0.780432
-0.051480	1.253135
1.014150	0.000239
1.579861	-1.228917

(5,2)

PCA: Important Notes

- PCA tries to summarize as much information as possible in the first PC, the rest in the second, and so on…
- PC's **do not have an interpretable meaning**, being a linear combination of features.
- Eigenvectors of the covariance matrix are actually directions of the axes where there is most variance.

Advantages of PCA

- Eradication of correlated features.
- Speeds-up algorithm
- Reduces overfitting
- Improves visualization

Disadvantages of PCA

- Less interpretable
- Data standardization is necessary
- Loss of Information

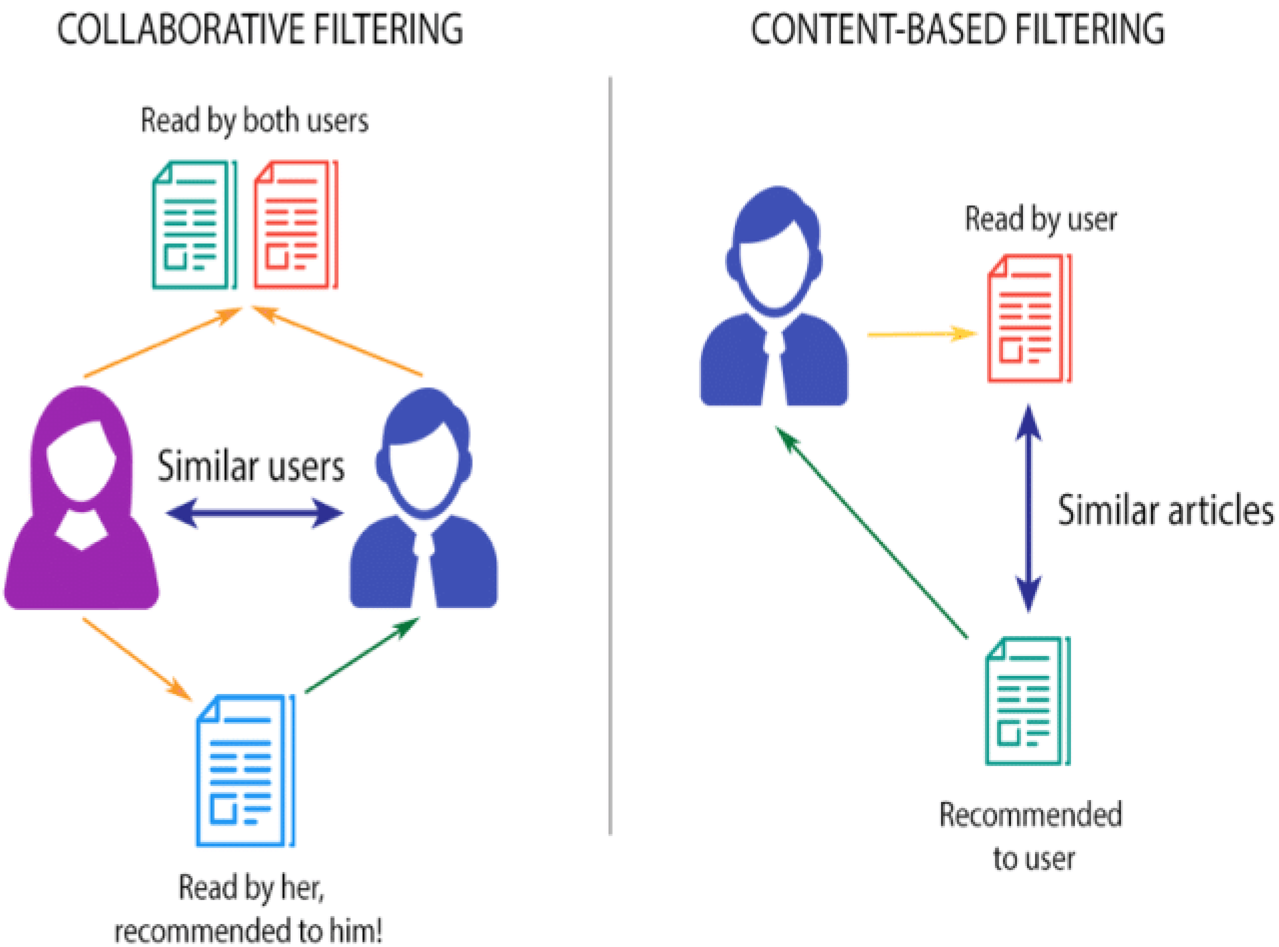
Principal Component Analysis

Implementation

Recommendation Systems

- Recommender systems are the systems that are designed to recommend things to the user based on different factors.
- These systems predict the most likely product that are of interest to the user.
- The recommender system deals with a large volume of information present by filtering the most important information based on the data provided by a user and other factors that take care of the user's preference and interest.

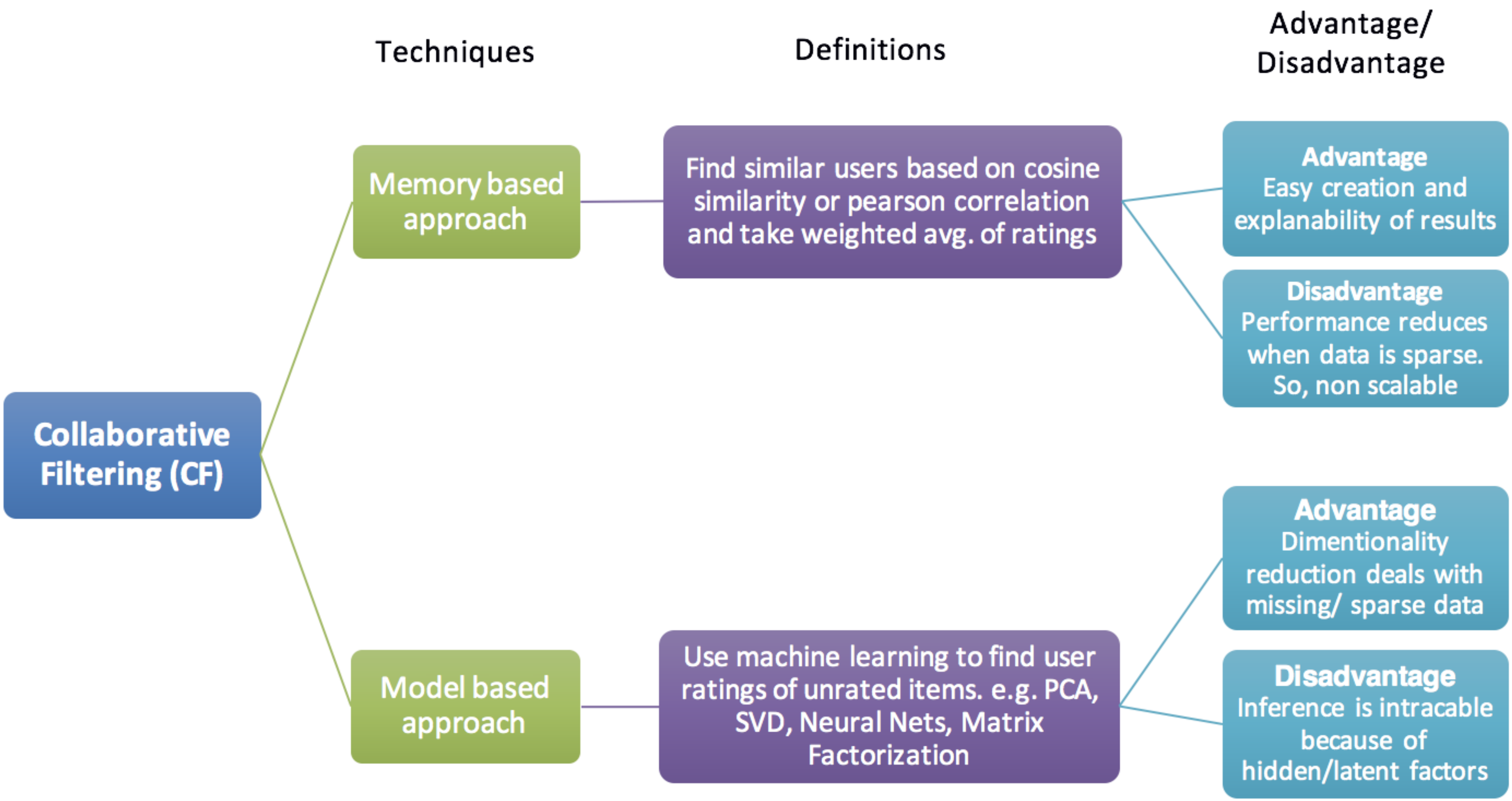
Types of recommender Systems



Collaborative Filtering

- Collaborative filtering technique works by building a database (**user-item matrix**) of preferences for items by users.
- It then matches users with relevant interest and preferences by **calculating similarities** between their profiles to make recommendations. Such users build a group called neighborhood.
- A user gets recommendations to those items that he has not rated before but that were already positively rated by users in his neighborhood.

Types of Collaborative Filtering



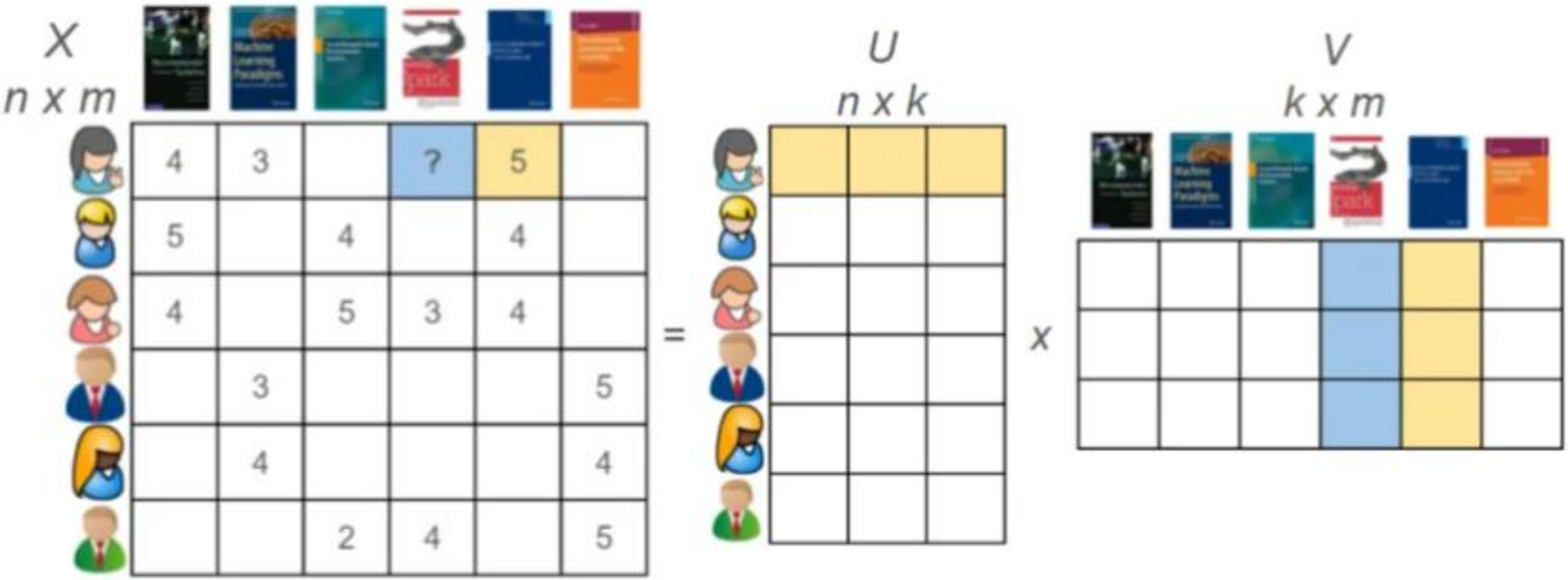
Matrix Factorization

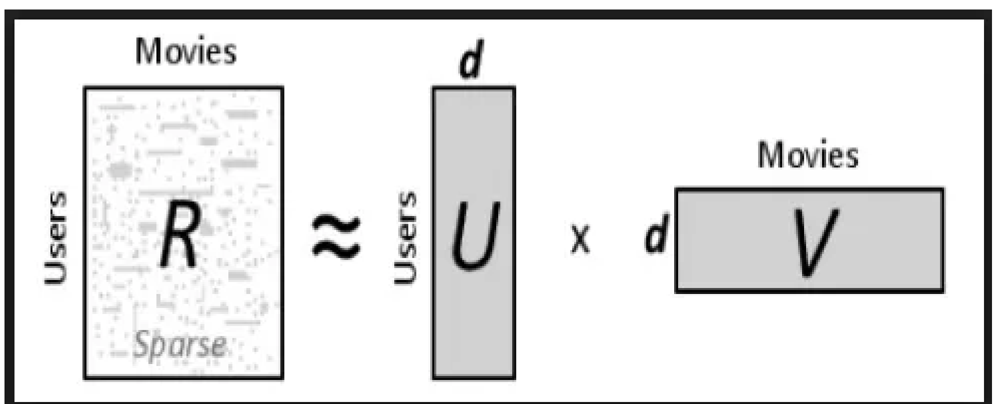
- Matrix factorization is used to factorize a matrix, i.e., to find out two (or more) matrices such that when you multiply them, you'll get back the original matrix.

- Matrix factorization can be used to discover features underlying the interactions between two different kinds of entities.

- One obvious application is to predict ratings in collaborative filtering—in other words, to recommend items to users.

Matrix Factorization





Singular Value Decomposition

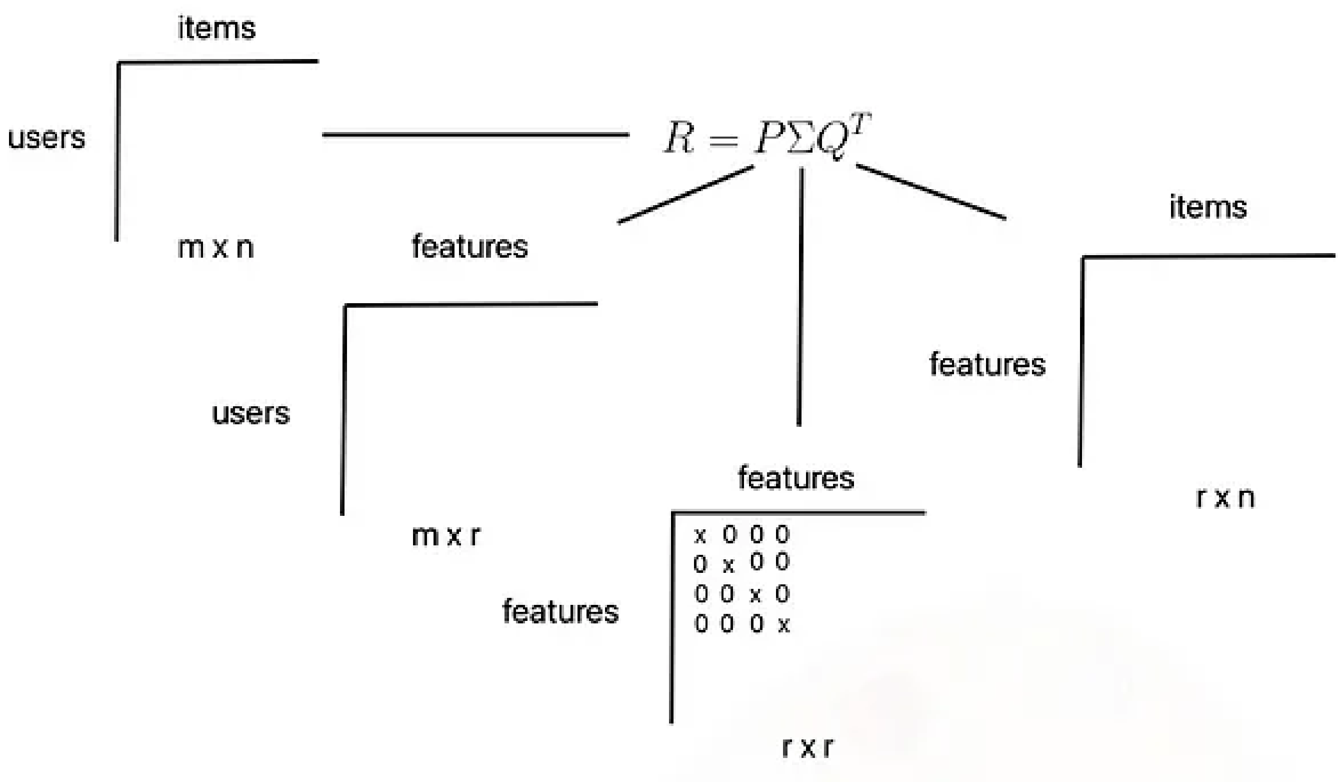
- When we have millions of users and/or items, computing pairwise correlations is expensive and slow.
- Is it possible to reduce the size of the ratings matrix?
- Is it really critical that we store every single item?
- For example, if a user liked a movie and its sequels (e.g. The Terminator, The Matrix). Do we really need to have a column in our ratings matrix for each movie?
- What if the set of movies could be represented using only k latent features where each feature corresponds to a category (e.g. genre) with common characteristics?

Singular Value Decomposition

- Similarly, the users could be represented using k dimensions.
- We can think of each feature as a demographic group (e.g. age, occupation) with similar preferences.

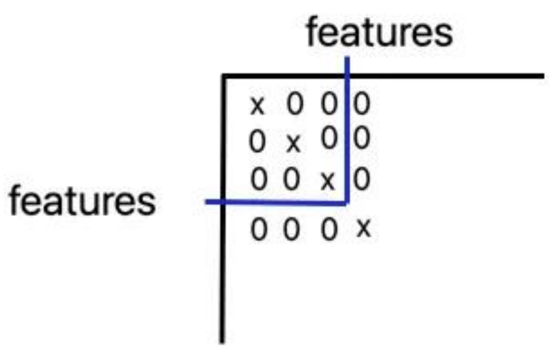
Singular Value Decomposition

- Suppose we had a ratings matrix with **m users** and **n items**. We can factor the matrix into two other matrices P and Q, and a diagonal matrix Sigma.



Singular Value Decomposition

- The key piece of information here is that each of the absolute values in the **diagonal matrix Sigma** represent how important the associated dimension is in terms of expressing the original ratings matrix.
- If we sort those values in descending order and pick top k (user defined) features, we can obtain the best approximation of the ratings matrix by multiplying the truncated matrices back together.

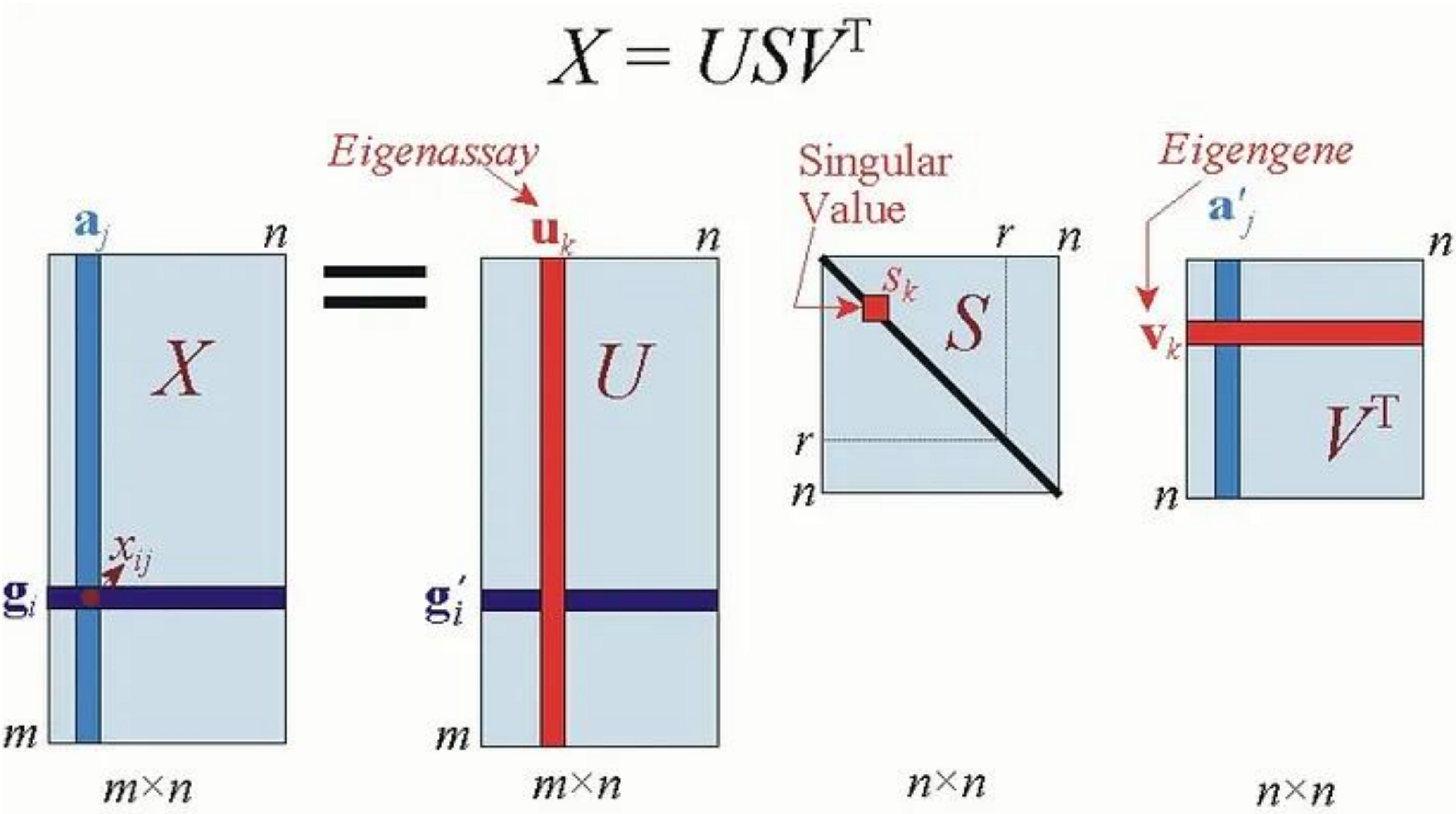


Singular Value Decomposition

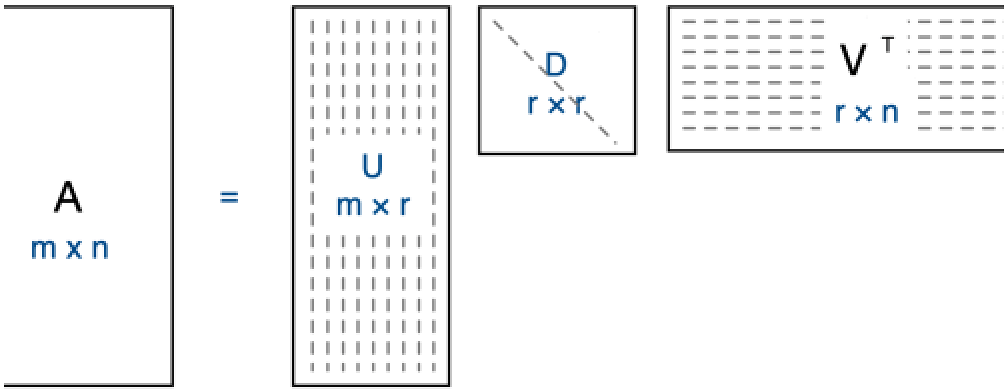
- The Singular Value Decomposition (SVD) of a matrix is a factorization of that matrix into **three** matrices.

- It has some interesting algebraic properties and conveys important geometrical and theoretical insights about linear transformations.

Singular Value Decomposition (SVD)



Singular Value Decomposition (SVD)



Rank: maximal number of linearly independent columns of A

$$A_{[m\times n]} = U_{[m\times r]}\Sigma_{[r\times r]}\left(V_{[n\times r]}\right)^T$$

- **A: Input (data) matrix**
  - m x n matrix (e.g., m users, n products)
- **U: Left singular vectors**
  - m x r matrix (e.g., m users, n 'concepts')
- **Σ: Singular values**
  - r x r diagonal matrix (strength of each 'concept')
    - ✓(r: rank of matrix A)
- **V: Right singular vectors**
  - n x r matrix (n products, r 'concepts')

SVD: Example

	Data	Information	Retrieval	Brain	Lungs
CS1	1	1	1	0	0
CS2	2	2	2	0	0
CS3	1	1	1	0	0
CS4	5	5	5	0	0
MD1	0	0	0	2	2
MD2	0	0	0	3	3
MD3	0	0	0	1	1

SVD: Example

$$
\begin{array}{cc}
 & \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \end{array} \\
\begin{array}{c} \\ \text{CS} \\ \\ \\ \\ \text{MD} \\ \\ \end{array} &
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{bmatrix}
\text{x}
\begin{bmatrix}
9.64 & 0 \\
0 & 5.29
\end{bmatrix}
\text{x}
\begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
$$

SVD: Example

$$\boxed{\mathbf{A}_{[\mathbf{n} \times \mathbf{m}]} = \mathbf{U}_{[\mathbf{n} \times \mathbf{r}]} \, \mathbf{\Lambda}_{[\mathbf{r} \times \mathbf{r}]} \, (\mathbf{V}_{[\mathbf{m} \times \mathbf{r}]})^T}$$

- **A**: n x m matrix (eg., n documents, m terms)
- **U**: n x r matrix (n documents, r concepts)
- **Λ**: r x r diagonal matrix (strength of each 'concept') (r : rank of the matrix)
- **V**: m x r matrix (m terms, r concepts)

SVD: Example

- $\mathbf{A} = \mathbf{U}\, \mathbf{\Lambda}\, \mathbf{V}^T$ - example:

doc-to-concept similarity matrix

$$
\begin{array}{c|ccccc}
 & \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \\
\hline
\text{CS} & 1 & 1 & 1 & 0 & 0 \\
\text{CS} & 2 & 2 & 2 & 0 & 0 \\
\text{CS} & 1 & 1 & 1 & 0 & 0 \\
\text{CS} & 5 & 5 & 5 & 0 & 0 \\
\text{MD} & 0 & 0 & 0 & 2 & 2 \\
\text{MD} & 0 & 0 & 0 & 3 & 3 \\
\text{MD} & 0 & 0 & 0 & 1 & 1
\end{array}
=
\begin{array}{cc}
\text{CS-concept} & \text{MD-concept} \\
\hline
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{array}
\times
\begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix}
\times
\begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}
$$

SVD: Example

- $\mathbf{A} = \mathbf{U}\, \mathbf{\Lambda}\, \mathbf{V}^T$ - example:

‘strength’ of CS-concept

$$
\begin{array}{cc}
 & \begin{array}{ccccc}\text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung}\end{array} \\
\begin{array}{c} \\ \text{CS} \\ \\ \\ \\ \text{MD} \\ \\ \end{array} &
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{bmatrix}
\mathrm{x}
\begin{bmatrix}
9.64 & 0 \\
0 & 5.29
\end{bmatrix}
\mathrm{x}
\begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
$$

SVD: Example

- $\mathbf{A} = \mathbf{U}\,\mathbf{\Lambda}\,\mathbf{V}^T$ - example:

term-to-concept
similarity matrix

CS-concept

$$
\begin{array}{c|c}
 & \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \end{array} \\
\begin{array}{c} \text{CS} \\ \\ \\ \\ \text{MD} \\ \\ \\ \end{array} &
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{bmatrix}
\times
\begin{bmatrix}
9.64 & 0 \\
0 & 5.29
\end{bmatrix}
\times
\begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
$$

Summary

- Eigenvectors
- Dimensionality Reduction
  - Principle Component Analysis
  - Singular Value Decomposition