Lecture Notes

Principal Component Analysis and Dimensionality Reduction

Detailed agenda

- Motivation - 2 Situations
- What is principal component analysis?
    - Going back to the regression example
    - A very high level summary
- Some building blocks
    1. Columns as dimensions, basis (Note: this is needed if SVM isn't done already)
    2. Basis transformation idea
    3. Variance as information
- Putting it all together
    - Rephrase what Principal Component Analysis using our new vocabulary
- Illustration – how Principal components are derived
    - 2D Example – pick principal components to represent data
- SVD to find principal components
- Choosing number of principal components
- (attempting) interpreting the Principal Components using loadings/plots
- Demo with real data
- Endnotes/ practical considerations

Motivation

“Surely, there must be a better way!”

Situation 1: A logistic regression setting where you have a lot of correlated variables (high multicollinearity). How do you handle this?

- One way would be doing a variable selection (stepwise/forward/backward).
- But each time you drop a variable, aren’t you losing some information?

There must be a better way of doing this!

Situation 2: You’re doing EDA on a dataset with n records and p variables. You want to visualize this dataset.

- You could look at pairwise scatter plots
- You’ll need to look at (p*(p-1)/2) plots
- But even if p = 20, this would mean 190 plots!

Again, there must be a better way of doing this!

What is Principal Component Analysis?

Going back to our logistic regression example

- Recall that multicollinearity, a situation where predictor variables are correlated with each other, is not good in a regression setup. Indeed, one of the assumptions of linear/logistic regression is that the predictors/independent variables are independent of each other.

- The approach we learnt so far: drop variables that are not very helpful in prediction
  - drop variables that have zero variance
  - leave further variable selection to the forward/ backward/stepwise variable selection methods

At the end of the process, you may still have a fairly large number of variables, and may have somewhat related variables – leading to unstable models. The result could still be far from ideal.

What we have: a large number of potentially correlated variables

What we want: a lower number of variables, that are *not correlated*, *without losing information* contained in the dataset

Principal Component Analysis let's us do just that!

What is Principal Component Analysis or PCA?

Statistical procedure to convert observations of possibly correlated variables into ‘Principal Components’ that are –

- Uncorrelated with/independent of each other
- Principal components are constructed to capture **maximum information** (?) in the data
- The principal components are **linear combinations** of the original variables

PCA is an **unsupervised** technique: there is no ‘Y’ or dependent/response variable

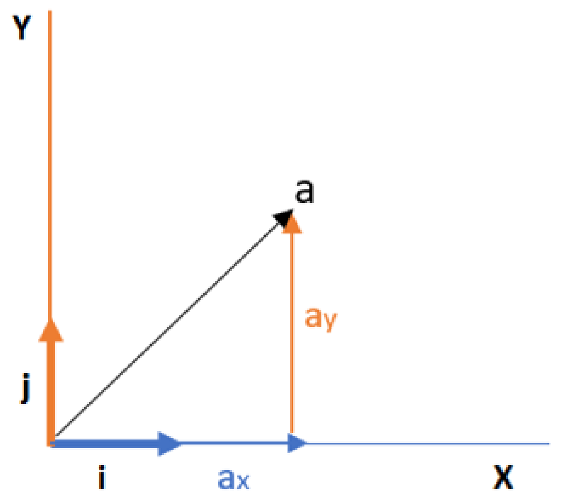
A very powerful technique, Principal Component Analysis, has several uses –

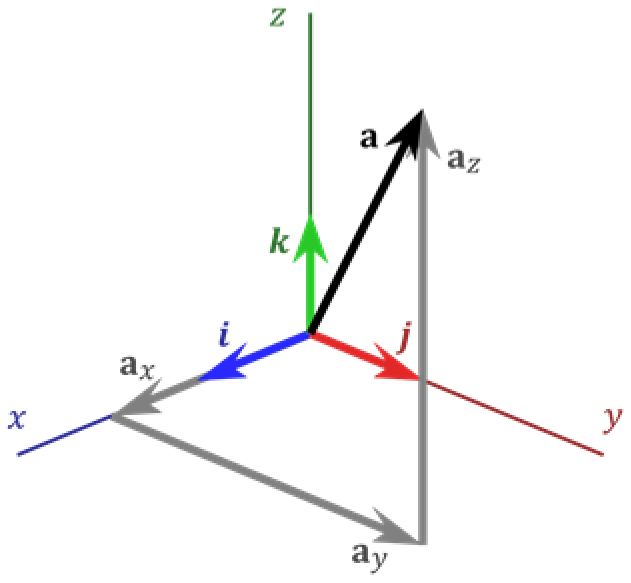
1. Dimensionality reduction
2. Data visualization and Exploratory Data Analysis
3. Create uncorrelated features/variables that can be an input to a prediction model
4. Uncovering latent variables/themes/concepts
5. Noise reduction in the dataset

Before we get into the specifics of the workings of PCA, we’ll need to get familiar with some building blocks.

Building blocks

Building Block 1: The ‘basis’ of a ‘space’





Two new terms, but the ideas are very familiar, and we understand them fairly well.

We’re all (probably) familiar with co-ordinate geometry, so we’ll do a very quick refresher

- Also called 2D geometry, X and Y axes are dimensions
- ***i*** (1,0) is a unit vector in X direction, ***j*** (0,1) is a unit vector in the Y direction.
- For point a: a_x, a_y are the units to move in i and j directions respectively to reach a
    - Also denoted as: a_x ***i*** + a_y ***j***
- Any point is in this 2D ‘space’ can be expressed in terms of ***i*** and ***j***
- The ***i*** and ***j*** vectors are the ‘basis’ of this ‘space’
- ***i*** and ***j*** are independent i.e. ***i*** can’t be expressed in terms of ***j*** and vice versa

The idea can be extended to any number of dimensions

Defining the basis of a space:

- Set of linearly independent vectors/directions that span the entire space i.e.
- Any point in the space can be represented as a combination of these vectors

For our dataset/dataframe/table/matrix :

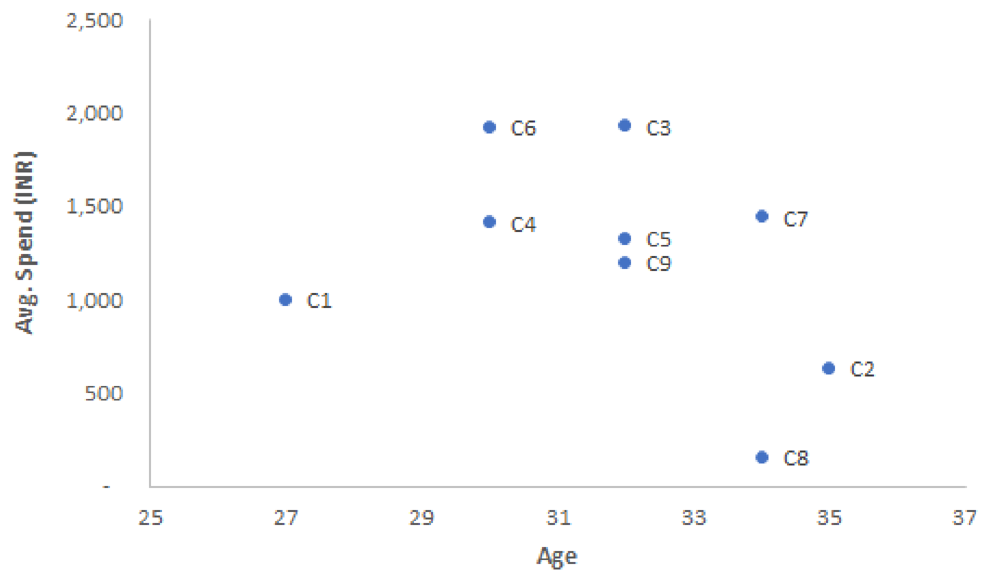
- Each row is a ‘point’ in the space
- Each column is a basis vector – any point is represented in terms of columns

Building Block 1: The ‘basis’ of a ‘space’ …. contd.

For our dataset/dataframe/table/matrix :

- Each column is a basis vector – any point is represented in terms of columns
- Each row is a ‘point’ in the space
- Let’s look at a simple 2D case
    - We visualize the data using the columns as dimensions covering the ‘space’
    - ‘Age’ and ‘Avg Spend’ are the basis for this space
- With 50 variables, we will have a 50D space at hand

	Customer Data	
Customer Identifier	Age	Avg Spend (INR)
C1	27	1,003
C2	35	636
C3	32	1,933
C4	30	1,416
C5	32	1,332
C6	30	1,930
C7	34	1,446
C8	34	155
C9	32	1,200



Building Block 2: Basis transformation

i, j aren't always the best basis:

It's World War 3, you're the captain of a ship who has noticed a threat on the Radar, and wants to aler your crew.

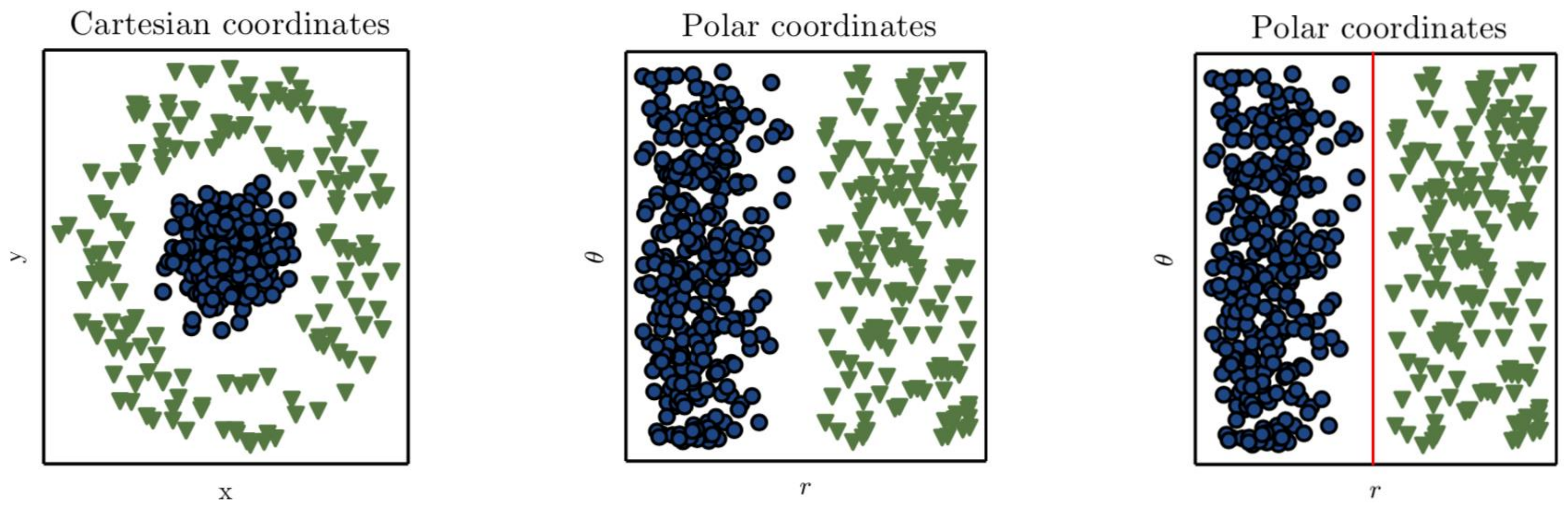
How do you communicate the position of the threat?

- Latitude and longitude? X, Y coordinate system, where your ship is the origin (0,0)?
- You'll probably use a polar coordinate system and report the angle and distance of the threat

This is most informative and the **best basis for the situation**.

Basis transformation is the process of converting your information from one set of basis to another. Or, representing your data in new columns different from the original. Often for convenience, efficiency, or just from common sense

Building Block 2: Basis transformation ... contd.



More examples where basis transformation helps –

Mechanical motion equations are significantly simplified and neat in polar coordinates

Electric field calculations are much cleaner in spherical coordinates

Dimensionality reduction: Watching a 2D movie! Yes, a 3D world is captured and represented on a 2D screen!

Dropping or adding a column to your dataset! Any modification on your columns, for that matter. Dropping column(s) is basis reduction.

You see where we're going with this?

Building Block 3: Variance as information

Situation: In a logistic regression setup, we're predicting if the customer will return the product. The table here has 4 variables.

1) Date: One single value in all records
2) Gender: all M, except one record
3) City Tier: fair amount of variations in values
4) Shoe Size (cm): fair amount of variation in values

For our prediction purpose, we would drop 'Date', and maybe also 'Gender'. Why?

Because these variables had no information to add. We acknowledge that -

Variance = Information!

Date	Gender	CityTier	Shoe Size (cm)
25-Jan	M	Metro	35
25-Jan	M	Non-Metro	35
25-Jan	M	Metro	32
25-Jan	M	Metro	30
25-Jan	M	Non-Metro	32
25-Jan	M	Metro	30
25-Jan	M	Metro	34
25-Jan	M	Metro	34
25-Jan	F	Metro	32
25-Jan	M	Non-Metro	33
25-Jan	M	Non-Metro	27

Therefore, variables which capture variance in the data, are the variables that capture the information in the data!

Taking the idea further, if two variables are very highly correlated, they together don't add a lot information than they do individually. So you can drop one of them. Do you now see why?

With this, we have covered the 3 building blocks needed to understand PCA.

Let's get to it!

Putting it all together

What does PCA do?

Principal component analysis (PCA) is one of a family of techniques for taking high-dimensional data, and using the dependencies between the variables to represent it in a more tractable, lower-dimensional basis, without losing too much information.

Given p features/variables in a dataset, PCA finds the principal components as

- a linear combination of the original features
- the principal components capture maximum variance in the dataset

1. The first principal component, capturing most variance, is calculated as:

$$\boldsymbol{Z_1 = \varphi_{11}X1 + \varphi_{21}X2 + \ldots + \varphi_{p1}Xp}$$

PCA finds the $\boldsymbol{\varphi}$ values such that the variance on Z_1 is maximum

2. The 2nd is found as one that has maximal variance of all linear combinations that are uncorrelated with Z_1

3. And like this, each additional component is capturing incremental variance

4. The algorithm calculates p Principal components (equal to number of variables in dataset)

- If we choose to represent the data with k components, and if $k < p$, then we are reducing dimensionality

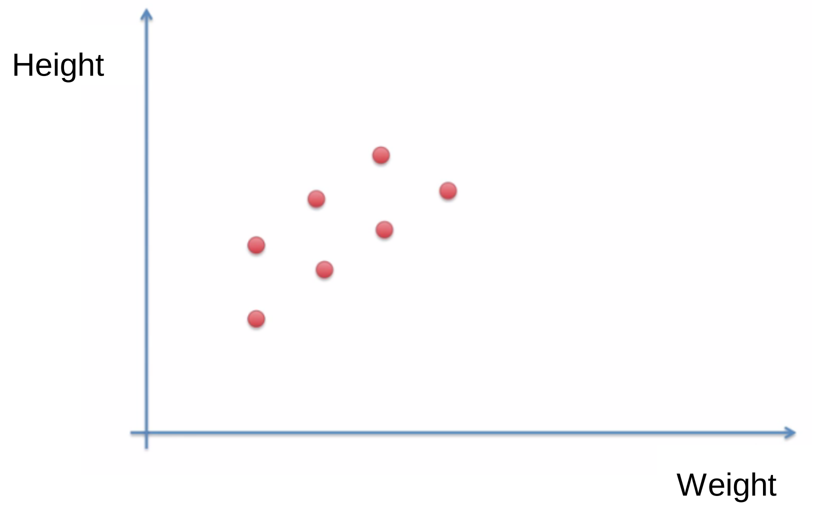
Illustration – workings of a PCA

Illustration – finding the principal components

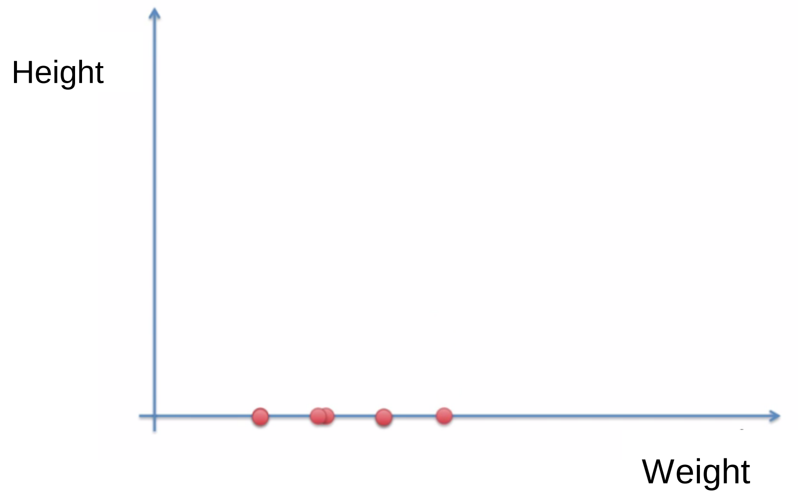
X1, X2 have correlation, but aren’t perfectly correlated.

Objective: to find directions/lines on which the projected data has maximum variance. Or, variance in data points, should be seen in the projections too.

We have several (infinite, actually) options here.



Let’s first try projecting on Weight axis/dimension
- We see that points vertically aligned now overlap
- Vertical variation not captured



Projecting on Height axis/dimension
- Points horizontally aligned now overlap
- Horizontal variation not captured

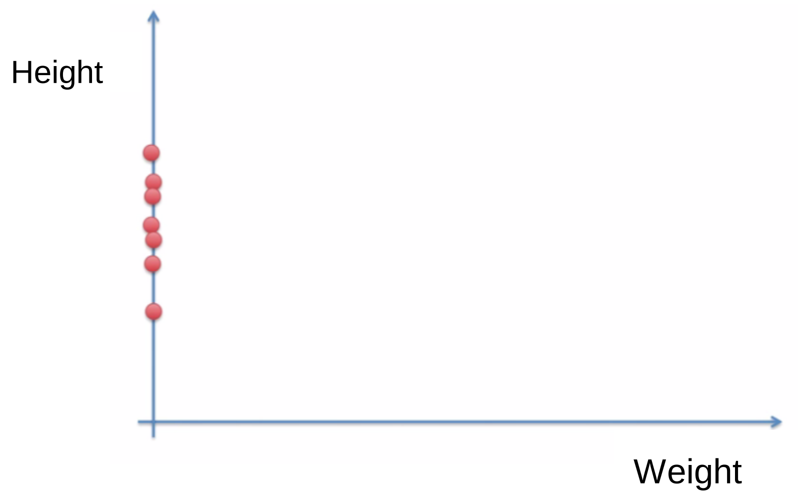


Illustration – finding the principal components … contd.

Objective: to find direction/line on which the projected data has maximum variance

We saw that a purely horizontal or vertical axis will not suffice as neither captures variation in both directions

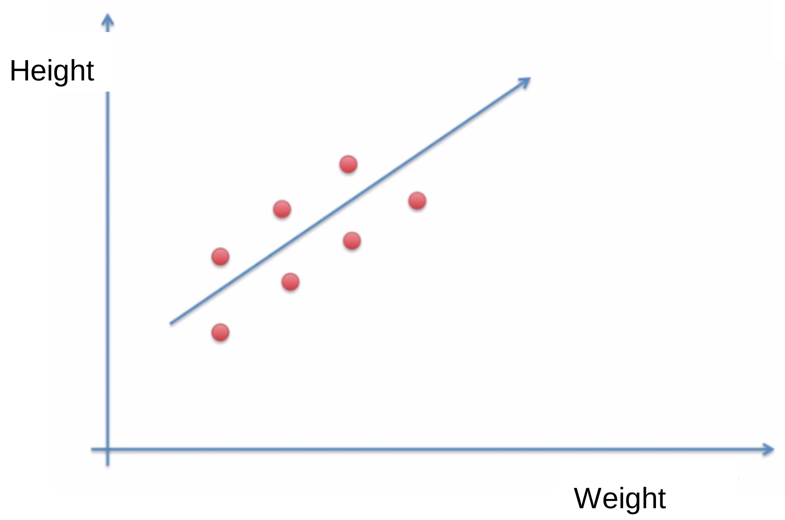
We therefore need a line that is angled

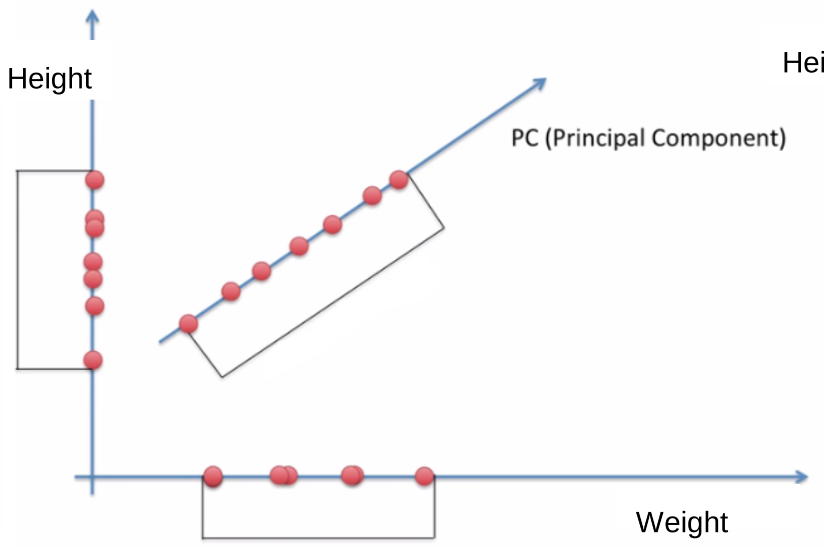
One such line is a line in the diagram. This is the line is closest to the data.

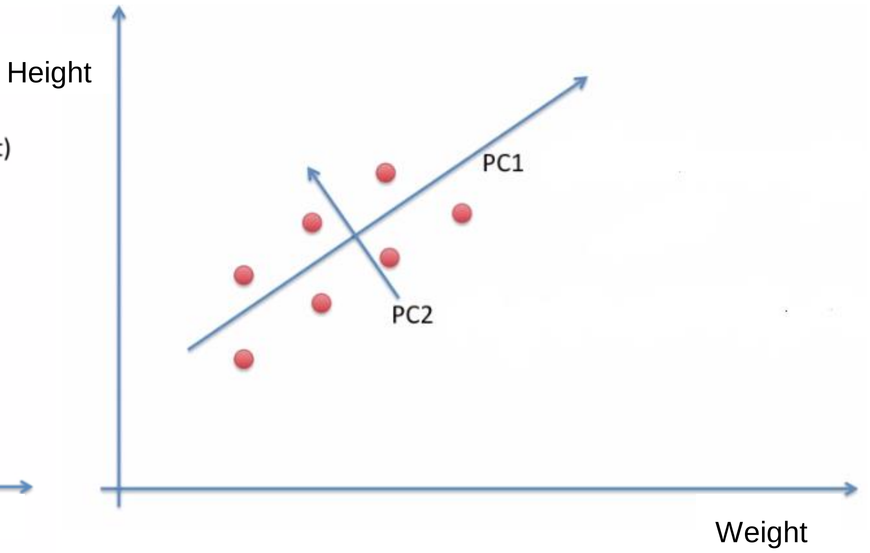
- Projections on this line will retain maximum variation in the original data points
- Note: in fact, PCA can also be considered as finding lines/planes/surfaces closest to data points

This line is our **first Principal Component!**

We still have some variance that is left – in the direction perpendicular to our first PC. This is our **2nd principal component**!

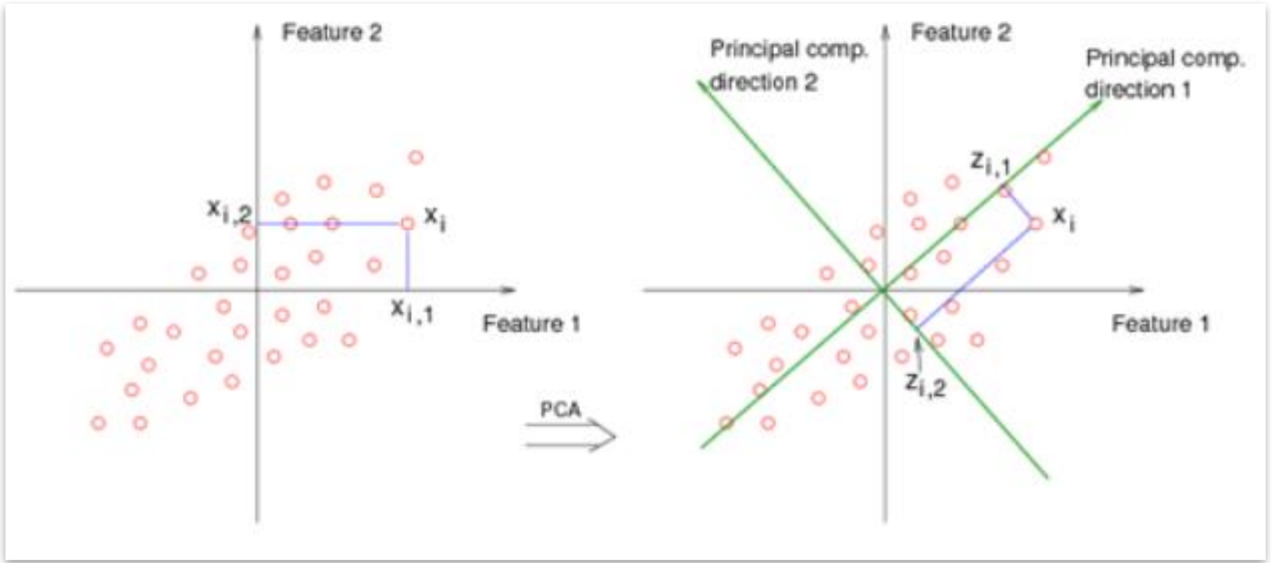


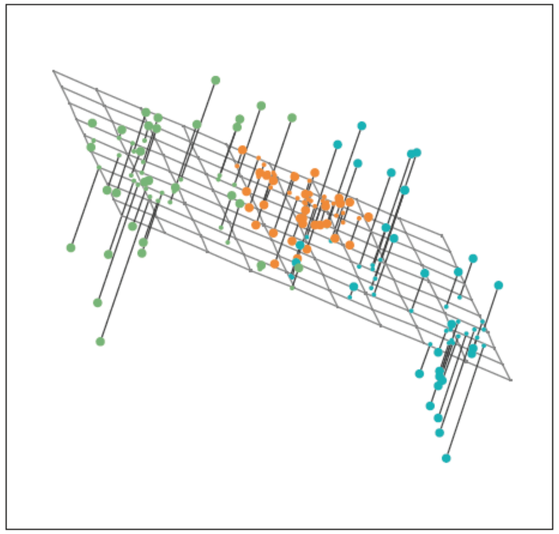


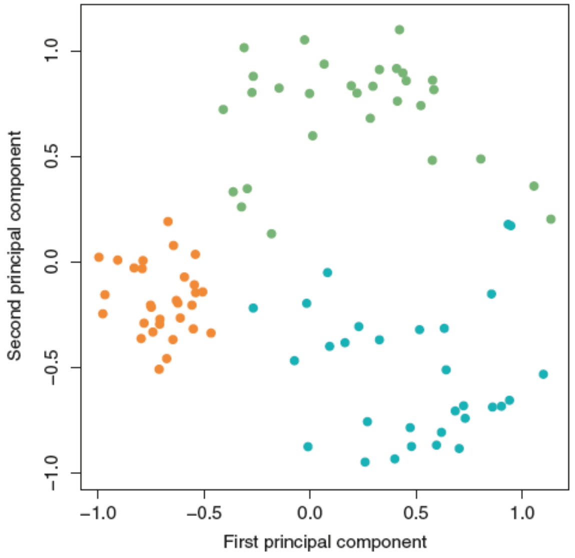


Summarizing the workings of a PCA

- PCA finds new basis/dimensions (Principal Components) which are uncorrelated
- The components are chosen to capture maximum variance in the data i.e. projections of data on the line should have maximum variance.
- Components are found sequentially, each time capturing incremental variance
- The process can be interpreted as finding lower dimensional lines/planes/surfaces that are closest to the data
    - together the first M principal components, along with their projections, provide the score vectors and the best M-dimensional approximation of the data
    - i.e. if M < p, then we have found a lower dimensional approximation of the data





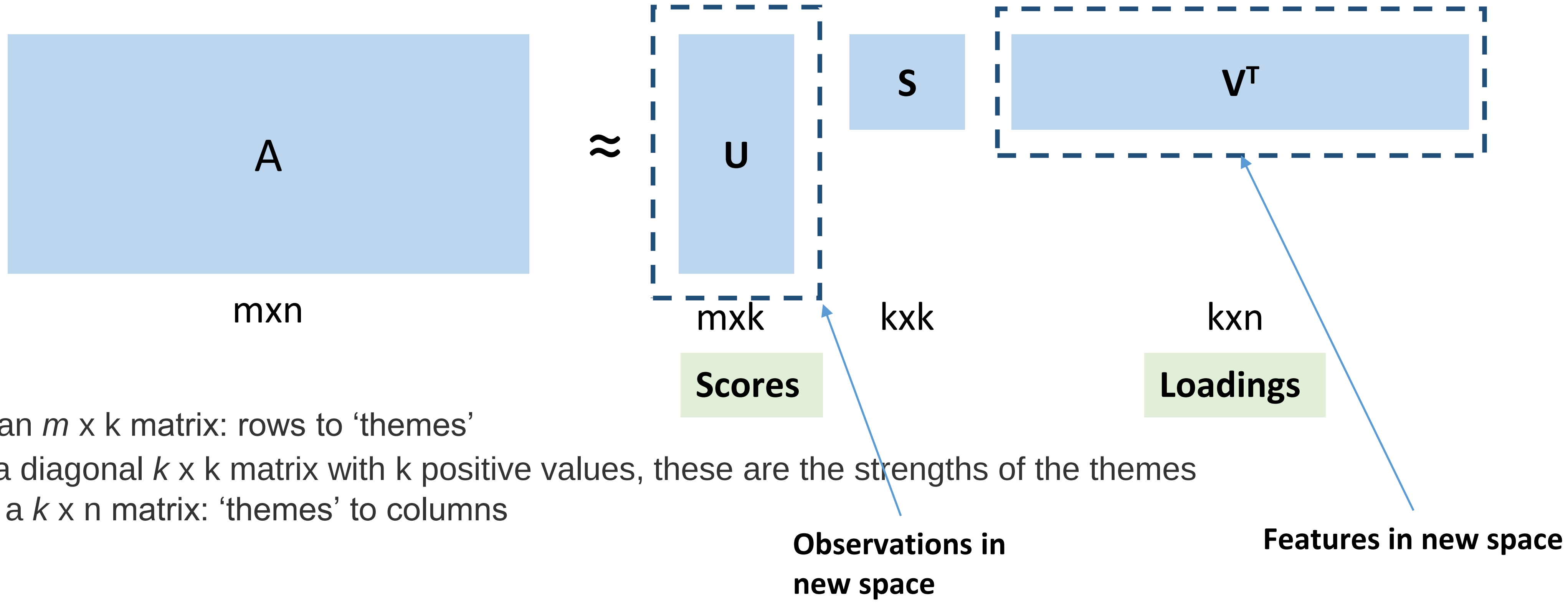


SVD to find principal components

Singular Value Decomposition

One of the techniques to calculate principal components
- ‘Decomposition’ because it breaks the original data matrix into 3 new matrices
- A robust method that is predominantly used in statistical packages today

$$A=U\Sigma V^T$$



U is an *m* x k matrix: rows to ‘themes’

Σ is a diagonal *k* x k matrix with k positive values, these are the strengths of the themes

V^T is a *k* x n matrix: ‘themes’ to columns

‘Hidden factors’ for recommender systems

- Find features that capture some characteristics of the rated items

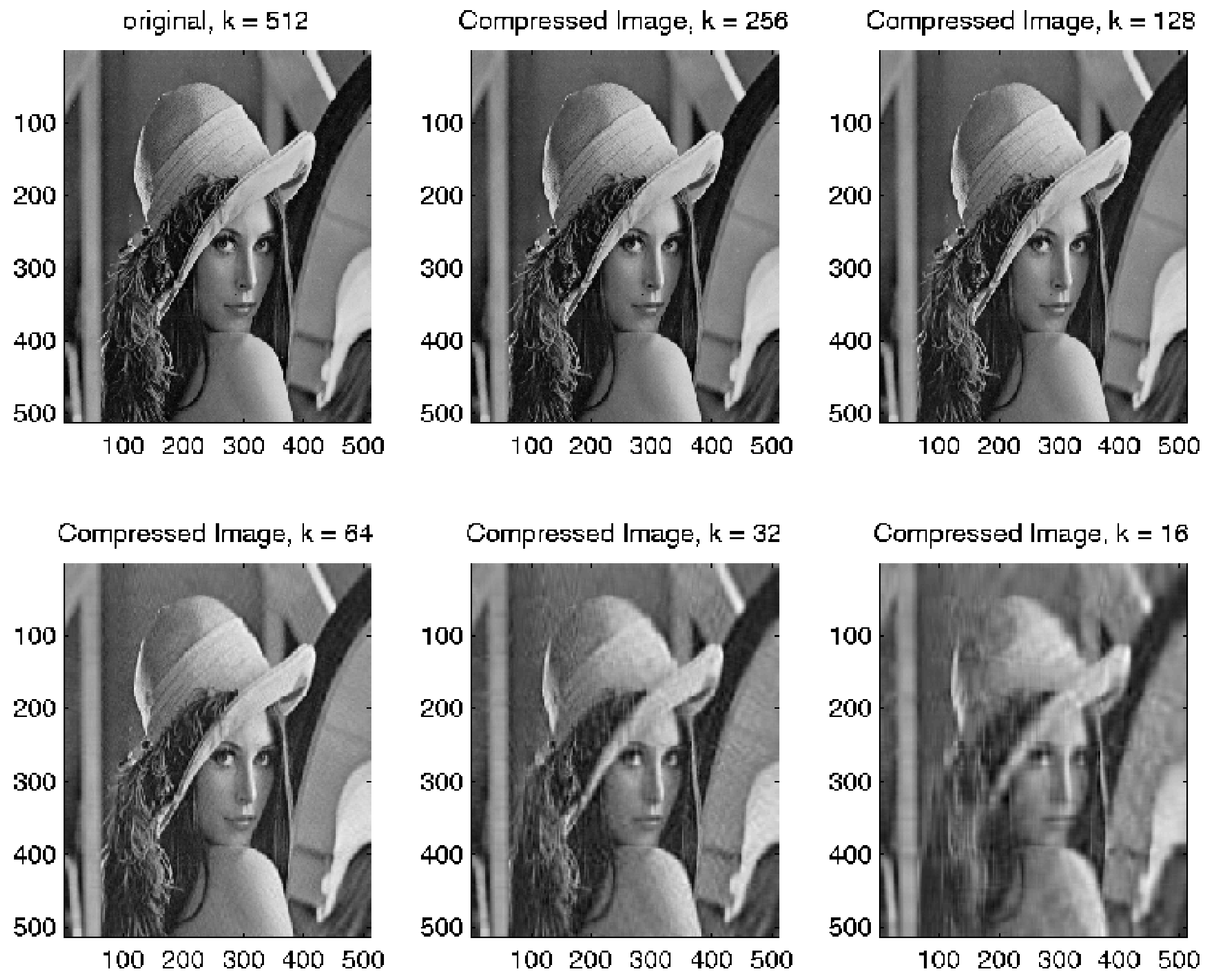
$$\underset{\mathbf{mXn}}{A} \approx \underset{\mathbf{mXk}}{\mathbf{U}}\ \underset{\mathbf{kXk}}{\mathbf{S}}\ \underset{\mathbf{kXn}}{\mathbf{V^T}}$$

‘users’, ‘books’ and **themes**:

U: user to theme
V: book to theme
Σ: diagonal elements are strengths of themes

$$\begin{array}{ccccc}
\text{I too had a love story} & \text{Five point someone} & \text{Half Girlfriend} & \text{Metamorphosis} & \text{The Wanderer} \\
\hline
1 & 1 & 0 & 0 & 1 \\
4 & 5 & 4 & 0 & 0 \\
5 & 4 & 4 & 0 & 0 \\
5 & 3 & 3 & 0 & 1 \\
1 & 1 & 0 & 5 & 4 \\
0 & 1 & 0 & 4 & 5 \\
0 & 0 & 2 & 2 & 3
\end{array}
\approx
\underset{}{\overset{\mathbf{U}}{\begin{bmatrix}
\mathbf{-0.11} & -0.04 & 0.23 \\
\mathbf{-0.56} & 0.22 & -0.09 \\
\mathbf{-0.56} & 0.23 & -0.02 \\
\mathbf{-0.50} & 0.12 & 0.14 \\
-0.23 & \mathbf{-0.61} & 0.42 \\
-0.18 & \mathbf{-0.64} & 0.02 \\
-0.16 & \mathbf{-0.33} & -0.86
\end{bmatrix}}}
\overset{\mathbf{S}}{\begin{bmatrix}
\mathbf{12.84} & 0 & 0 \\
0 & \mathbf{9.43} & 0 \\
0 & 0 & \mathbf{2.13}
\end{bmatrix}}
\overset{\mathbf{V_T}}{\begin{bmatrix}
\mathbf{-0.61} & \mathbf{-0.55} & \mathbf{-0.49} & -0.17 & -0.23 \\
0.21 & 0.11 & 0.16 & \mathbf{-0.66} & \mathbf{-0.69} \\
0.42 & 0.27 & -0.81 & 0.21 & -0.21
\end{bmatrix}}$$

Dimensionality reduction with SVD



Choosing k < n

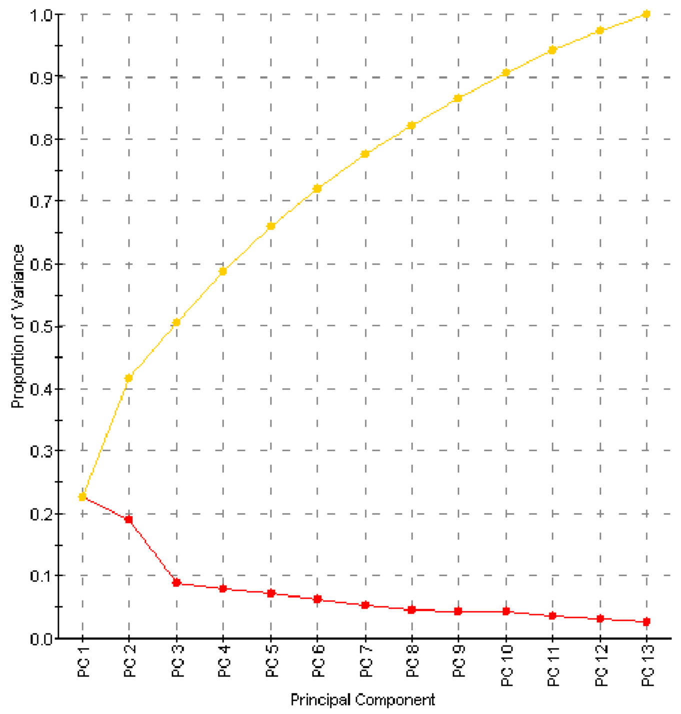
- Widespread use image compression and data transmission
- Depending on the application, we choose the number of components such that the essence of the data (that is essential to the task) is retained

Choosing number of Principal components

Choosing number of principal components

To reduce the dimensionality, the number of components are lower than the original columns
Optimal ‘k’:

- Top ‘k’ components capture what proportion of the variance?
- We look at a scree plot, similar to the plot we used to choose number of clusters in k-means clustering
- Against the #principal components, we look at the cumulative proportion of variance captured
- Choose k after which incremental benefit negligible
- Common choices: 80%, 90%, 95% variance



Demo with real data

Endnotes, practical considerations

Some practical considerations

Most packages use SVD for finding the Principal Components

- PCA using this method assumes that the data is centered and scaled
- In such implementations, PCA is scale sensitive - don’t forget to centre and scale the data!

PCA is inherently a linear transformation method

- While PCA can significantly boost performance when followed by linear models (like logistic regression),
- For non-linear methods, it can still give significant speed boosts be reducing the number of variables

Dimensionality reduction

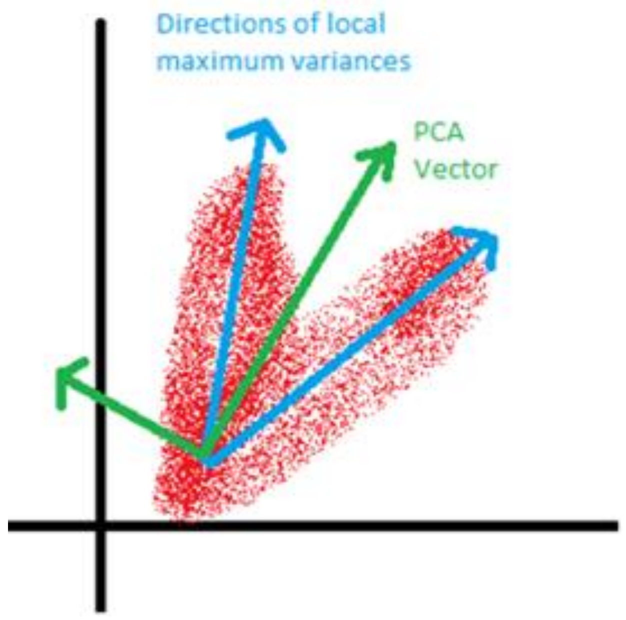
- In cases where you already have largely uncorrelated features, a forced reduction in dimensionality can lead to severe loss of information
- Be sure to be careful that you’re capturing significant variance as well

Shortcomings of PCA, and alternatives

PCA has 3 major assumptions/simplifications embedded –

1. The PCs have to be linear combinations of the original columns
   - Why limit ourselves to linearity when we can go non-linear?
   - t-SNE is an alternative, although computationally very expensive
2. PCA requires the PCs to be uncorrelated/orthogonal/perpendicular
   - Sometimes the data demands that correlated components to represent the data
   - ICA (Independent Component Analysis) overcomes this drawback, but is several times slower than PCA
3. PCA assumes low variance components are not very useful -
   - In supervised learning situations, this can lead to loss of valuable information. This is especially true for highly imbalanced classes/variables.

Despite some drawbacks, PCA is a very efficient and powerful technique to reduce complexity in data and discover patterns.



Summarizing our learning

Principal Component Analysis, finds uncorrelated components that are linear combinations of the original variables and capture the variance/ information in the data. Another interpretation is that it finds the best lower dimensional approximation of the data.

A very powerful technique that has several uses –

1. Dimensionality reduction
   - Widespread use in several applications, we saw image compression example
2. Data visualization and Exploratory Data Analysis
   - Reduce to <3D and we can now visualize the dataset with dozens of features
3. Create uncorrelated features/variables that can be an input to a prediction model
   - Doing a PCA will reduce the need for any stepwise selection in a regression model
   - For non-linear methods (e.g. Neural Networks), it will still give a significant performance boost by reducing the number of variables
4. Uncovering latent variables/themes/concepts
   - ‘Hidden’/’latent’ factors in recommender systems
5. Noise reduction in the dataset
   - Re-construct data with 95% variance components to clean up data

While it does have some drawbacks, the idea is pretty powerful, and PCA is pretty much a ‘default’ method for the applications mentioned above owing to its efficiency, elegant geometrical interpretation, and simplicity.

Appendix