Dimension Reduction Techniques

Unsupervised Learning

Agenda

- Curse of Dimensionality
- Dimension Reduction
- Principal Component Analysis (PCA)
    - Procedure
    - Terminologies
    - Selecting Principal Components
- Case Study

Agenda

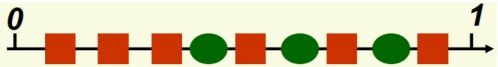
- Signal to noise ratio (SNR)
- Kernel PCA
- Multiple Correspondence Analysis (MCA)
  - Example
  - Explanation

Curse of dimensionality

1. If the number of features ‘n’ is large, the number of samples m, may be too small for accurate parameter estimation
2. covariance matrix has n^2 parameters $\boldsymbol{\Sigma} = \begin{bmatrix} \Sigma_{1,1} & \cdots & \Sigma_{1,n} \\ \vdots & \ddots & \vdots \\ \Sigma_{n,1} & \cdots & \Sigma_{n,n} \end{bmatrix}$
3. For accurate estimation, sample size should be much bigger than n^2, to be able to accurately represent the covariance, otherwise the model may become too complicated for the data, overfitting
4. If m < n^2 we assume that features are uncorrelated (because we cannot represent it accurately with the given m points), even if we know this assumption is wrong. Doing so, we do not represent the covariance as parameters in our model

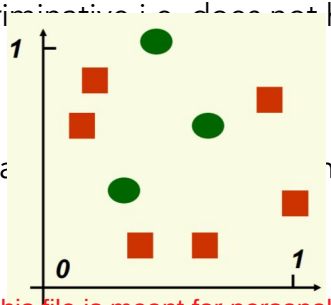
Curse of dimensionality

1. Suppose we have nine data points on a single dimension as shown



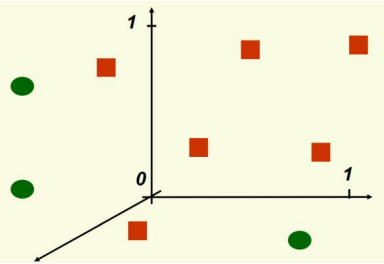
2. The feature is no very discriminative i.e. does not help in segregating these points clearly

3. Then we represent these da nsions…



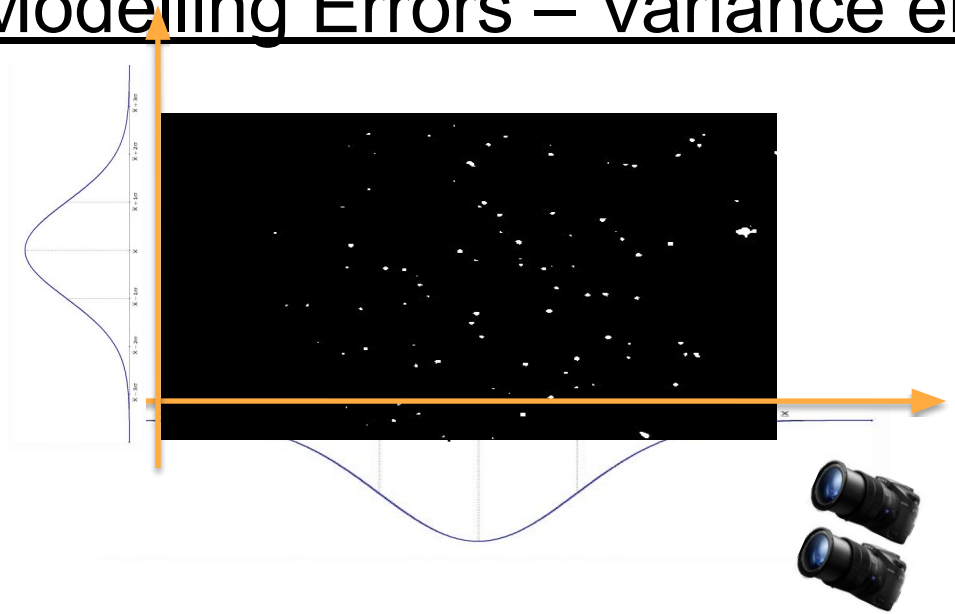
Curse of dimensionality

4. Yet again the two features are not helping in cleanly separating the data points. So we add another feature, a third dimension



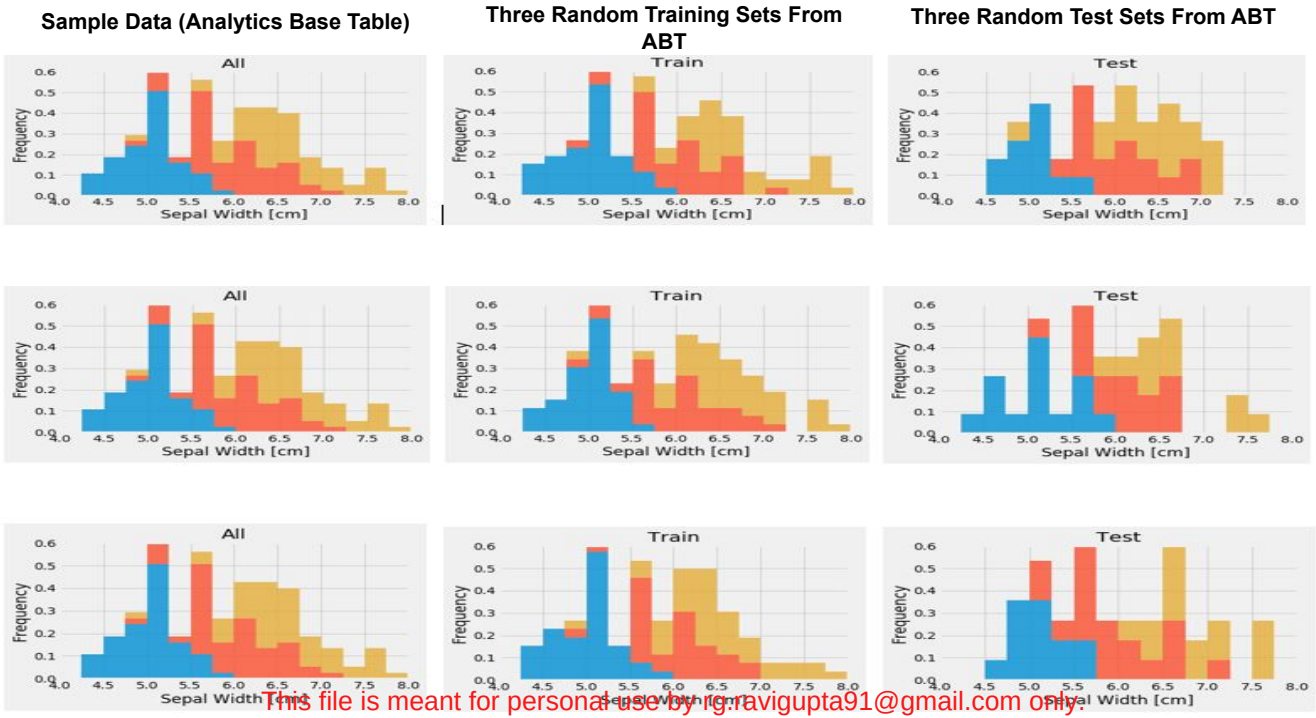
5. The data points get spread far and wide. The problem with this state is, we do not now how the data would have been spread (if we had more) in these empty spaces

Modelling Errors – Variance error



Sample / snapshot

Modelling Errors Visual demo of variance in training and test data



Curse of dimensionality

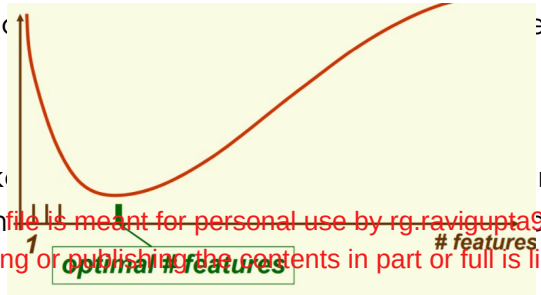
8. In general, if n samples is dense enough in 1D

9. Then in d dimensions we need n^d samples and n^d grows as a function of d

10. For a given sample size, If we can't solve a problem with a few features, adding more features seems like a good idea

11. As we add more features, the d e (sparse feature space)

12. Any model in such a case is lik nce errors. This is curse of dimensionality and very co is ng



Dimension Reduction

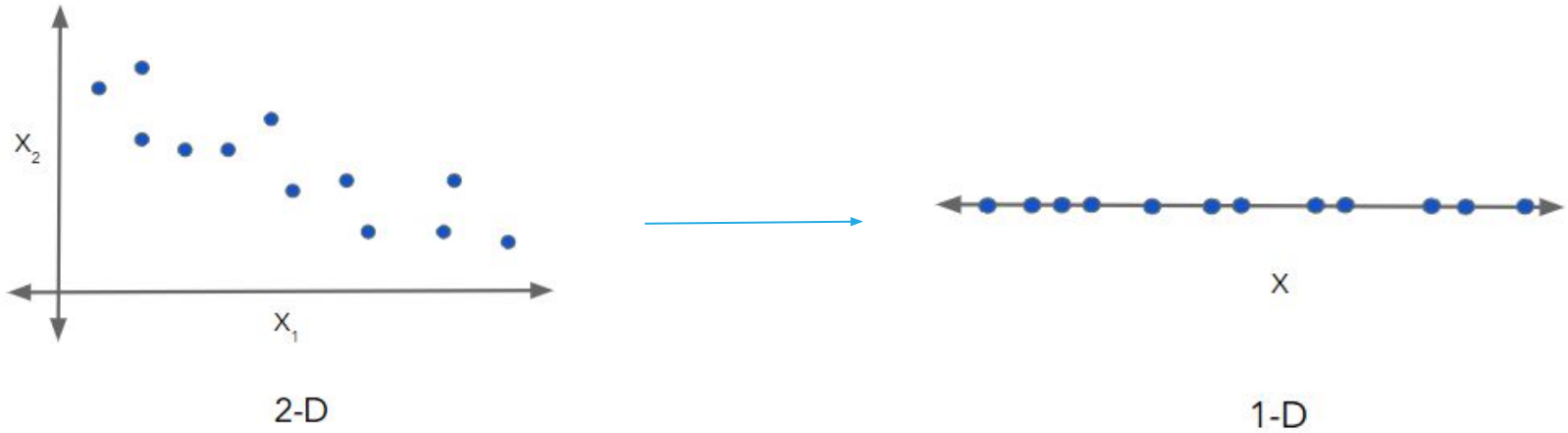
Dimension reduction

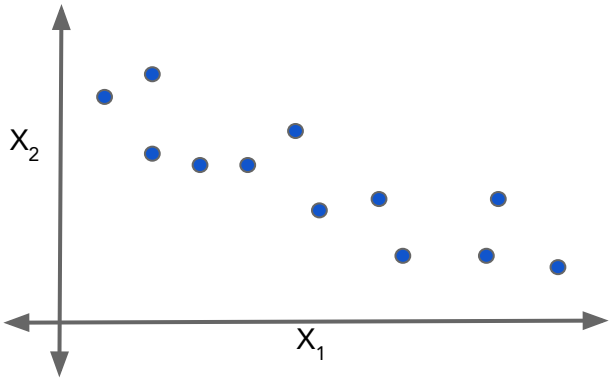
- The real-life dataset may contain a large number of features under study
- For example, while estimating the price of a mobile phone we need to consider various features like screen size, internal storage, camera quality, battery backup and so on
- The dataset with a large number of features needs more time for training the model. Also, it can cause the overfitting

Dimension reduction

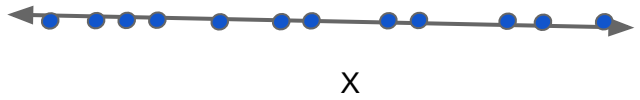
- To avoid such issues, one can reduce the dimension of the dataset
- The dimension reduction techniques remove the redundant variables/ noise in the original data, which reduces the training time
- Reducing the dataset to 2 or 3 dimensions helps in visualization of the data
- Various dimension reduction techniques:
  - Principal Component Analysis (PCA)
  - Linear Discriminant Analysis (LDA)
  - Factor Analysis

Dimension reduction





2-D



1-D

Approaches for dimension reduction

- Two different approaches can be used for dimension reduction: Projection, Manifold learning
- In the projection approach, the original dataset is projected onto the lower-dimensional plane
- PCA uses the projection approach for dimension reduction
- This method is not effective if the dataset has different layers in the higher dimensions
- In manifold learning, a manifold is created on which the dataset lies

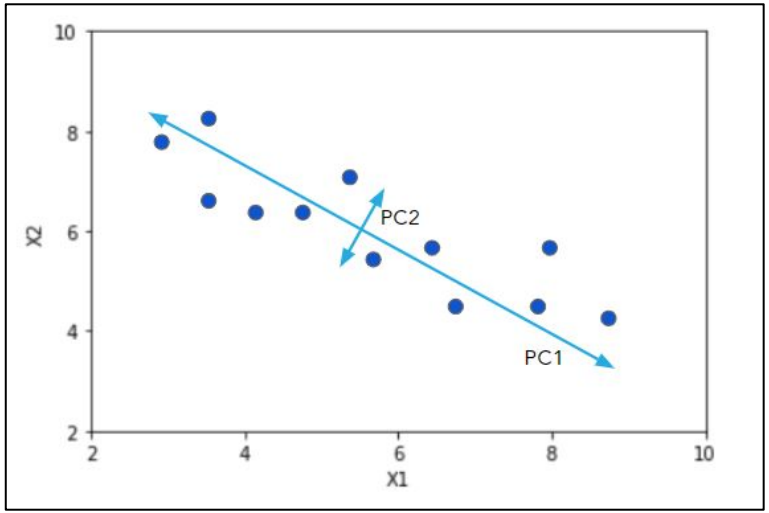
Principal Component Analysis (PCA)

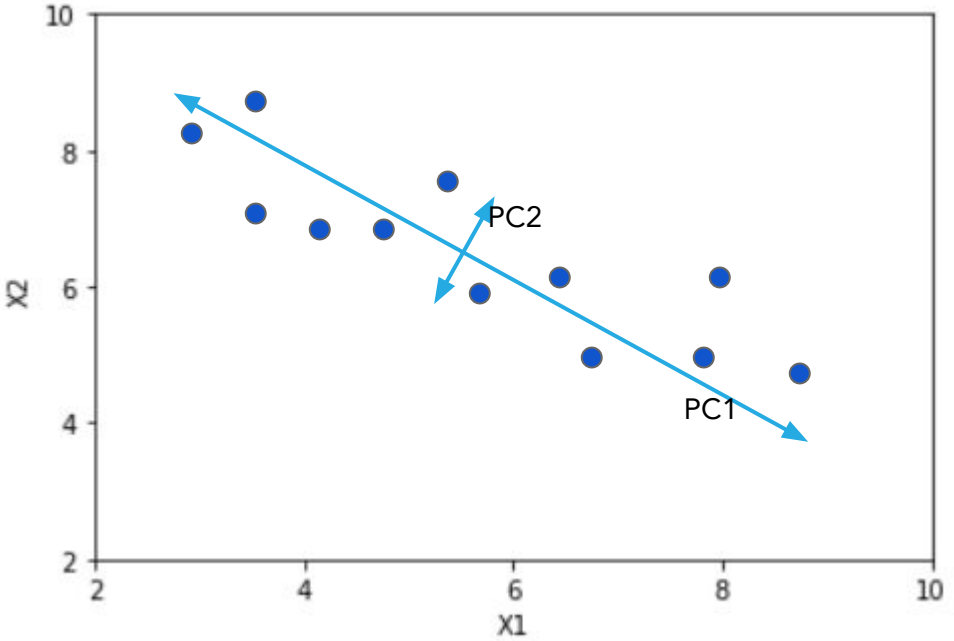
PCA

- It is one of the dimensionality reduction techniques that is used to reduce the dimensions of the large datasets
- It transforms the large set of features into a small set such that it will contain the maximum information in the original data
- The number of components is less than or equal to the number of independent variables
- PCA projects the original dataset on the lower dimensional plane
- It transoms the original data to a new set of uncorrelated variables

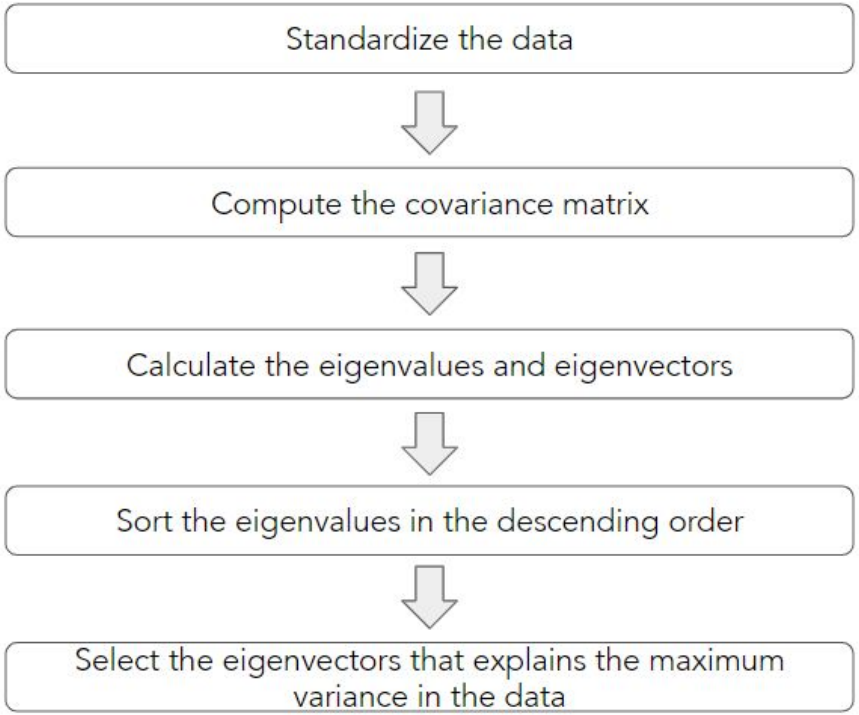
PCA

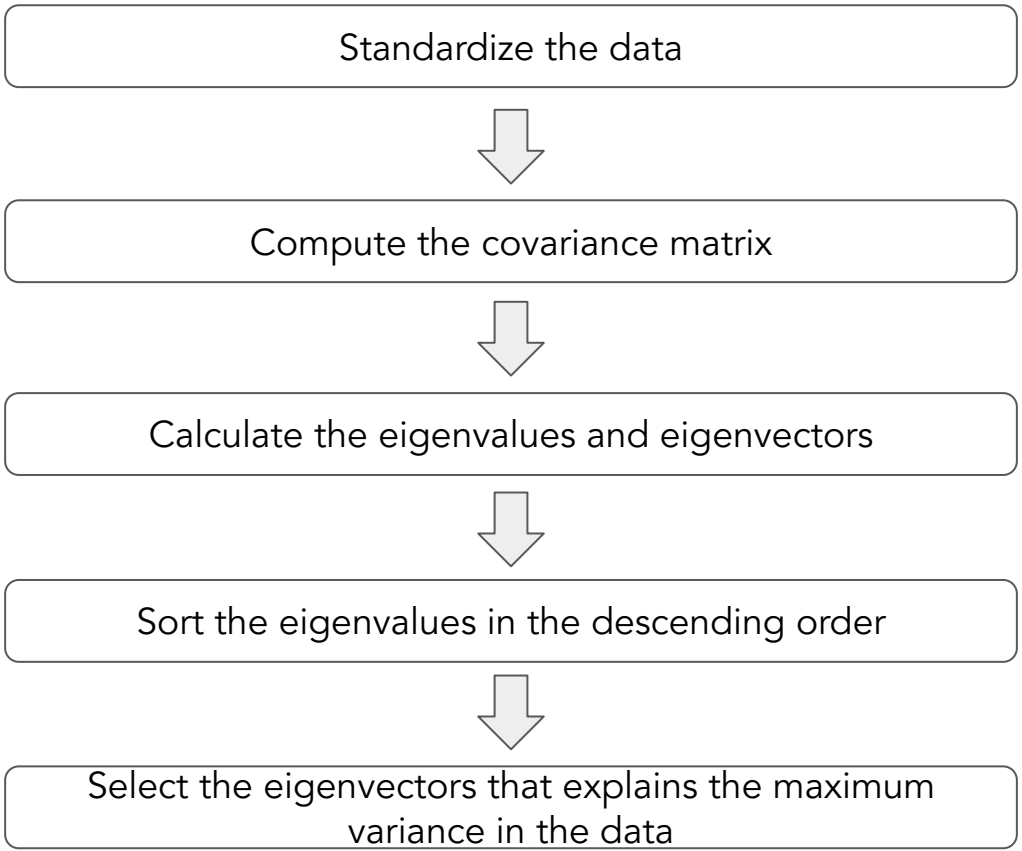
- The first principal component (PC1) exhibits the direction of maximum variance in the data
- It is used to remove the redundancy in the data
- PCA reduces the multicollinearity (if present) in the original data
- Principal components are always orthogonal to each other





PCA - procedure





Application

- PCA is mainly used in image compression, facial recognition models
- It is also used in the exploratory analysis to reduce the dimension of data before applying machine learning methods
- Used in the field of psychology, finance to identify the patterns high dimensional data

Python code

In python, we use the following code to perform PCA:

```
# import the function
from sklearn.decomposition import PCA

# specify required no of components to 'n_components'
pca = PCA(n_components = k)

# fit_transform() fits the model and transforms the original data
# pass the standardized data to fit PCA
pca.fit_transform(standardized_data)
```

Terminologies

Covariance

- The covariance measures how co-dependent two variables are
- Positive covariance value means that the two variables are directly proportional to each other
- Negative covariance value means that the two variables are inversely proportional to each other
- It is similar to variance, but the variance illustrates the variation of the single variable and covariance explains how two variables vary together

Covariance

The covariance between two variables X and Y is given by

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

Xi = values taken by variable X , $\forall$ X $\in$ [1, n]

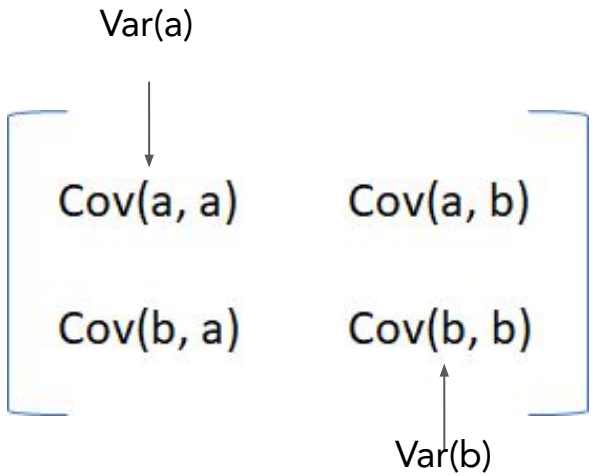
Yi = values taken by variable Y , $\forall$ Y $\in$ [1, n]

$\overline{X}$ = mean of Xi

$\overline{Y}$ = mean of Yi

Covariance matrix

- The covariance matrix explains the covariance between the pair of variables
- The diagonal entries represent the variance of the variable, as it is the covariance of the variable with itself
- The diagonal matrix is always symmetric
- The off-diagonal entries are covariance between the variables that represent the distortions (redundancy) in the data

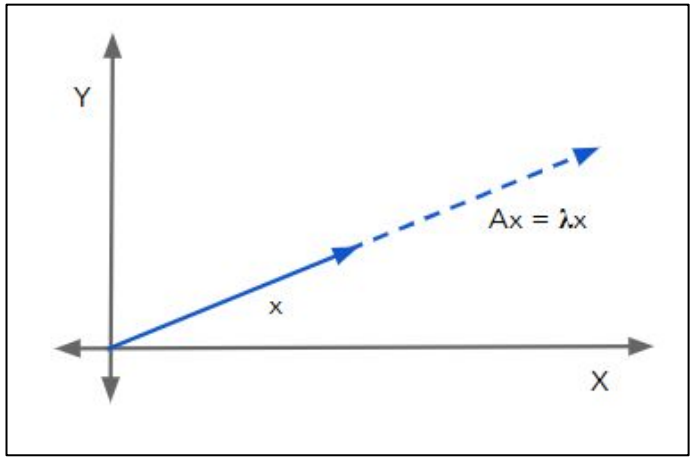


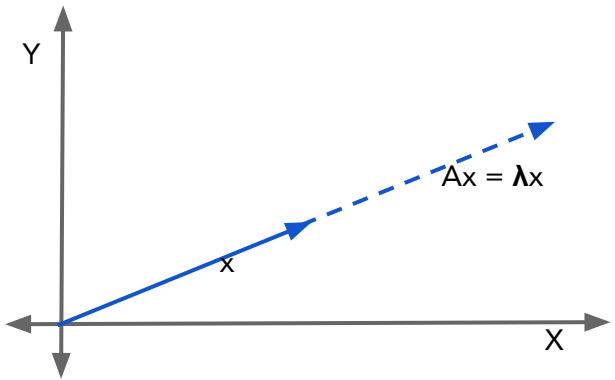
Eigenvalue

- For any nxn matrix A, we can find n eigenvalues that satisfy the characteristic equation
- A characteristic equation is defined as: $|A - \lambda I| = 0$ i.e. $\det(A - \lambda I) = 0$
- The characteristic polynomial for matrix A given as $|A - \lambda I|$
- The scalar value λ is known as the eigenvalue of the matrix A
- Eigenvalues can be real/ complex in nature

Eigenvector

- For each eigenvalue λ of a matrix A, there exist a non-zero vector x, which satisfy the equation: $(A - \lambda I)x = 0$ i.e. $Ax = \lambda x$
- The vector x is known as the eigenvector corresponding to the eigenvalue λ
- Eigenvectors are always orthogonal to each other
- The eigenvector is a vector that does not changes its direction, after transformation by matrix A





Calculate the eigenvalues and eigenvectors of the given matrix.

$$A = \begin{bmatrix} 2 & -3 \\ 1 & 6 \end{bmatrix}$$

Eigenvalues and eigenvectors

- In PCA, the coefficients of the principal components are the eigenvectors of the covariance matrix
- The corresponding eigenvalue represents the total variance explained by that principal component
- The percentage of variation explained by the i^{th} component is given as

$$\left(\frac{\lambda_i}{\sum_{i=1}^{n} \lambda_i}\right) * 100$$

where λ_i is the i^{th} eigenvalue.

Selecting Principal Components

Selecting principal components

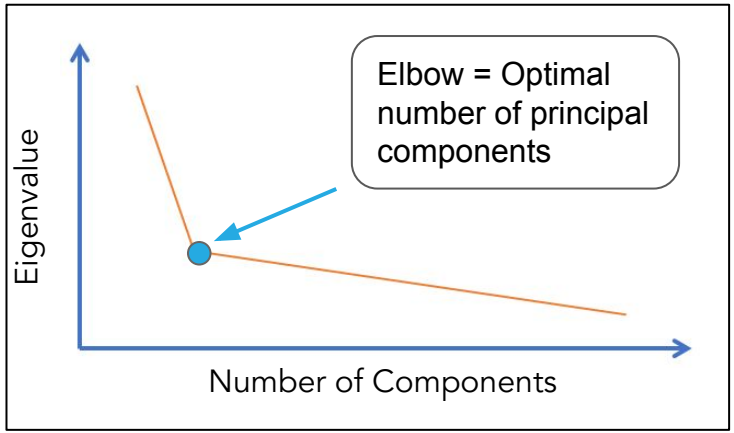
- After calculating the eigenvalues and eigenvectors of the covariance matrix, we need to select the principal components which retain the maximum variance in the data
- The first principal component (PC1) is the eigenvector associated with the highest eigenvalue. PC1 indicates the direction of maximum variation in the data
- Different criteria to select the principal components:
  - Kaiser Criterion
  - Scree Plot

Kaiser criterion

- It is an easy criterion to choose the principal components
- It selects the eigenvector as a principal component for which the corresponding eigenvalue is greater than 1
- Higher eigenvalues correspond to the principal components that explain most of the variance in the data

Scree plot

- This method plots the eigenvalues (Y-axis) against the number of principal components (X-axis)
- The elbow point in the scree plot corresponds to the optimal number of components
- After the elbow point, the components do not contribute much to the variance in the data
- This method fails if there is no explicit elbow point in the scree plot



Percentage of total variance

- One can decide the number of principal components based on the percentage of variance explained by all the variables
- In most of the cases, the components explaining 70-80% of the variance can be considered as the principal components
- On the other hand, in some examples, the first few components explain only 50-60% of the total variance

Case Study

Business example

The Department of Social Welfare in Canada has collected data on the various factors that influence the crime rate (per 1,00,000 individuals) in different cities.

Three different factors are considered for the study: Total population, Unemployed individuals (between age 18-65) and Average annual income of the individual.

Let us reduce the 3-D data to 2-D to make the data more interpretable. In order to do so, we try to obtain 2 principal components which preserve the maximum information in the original data.

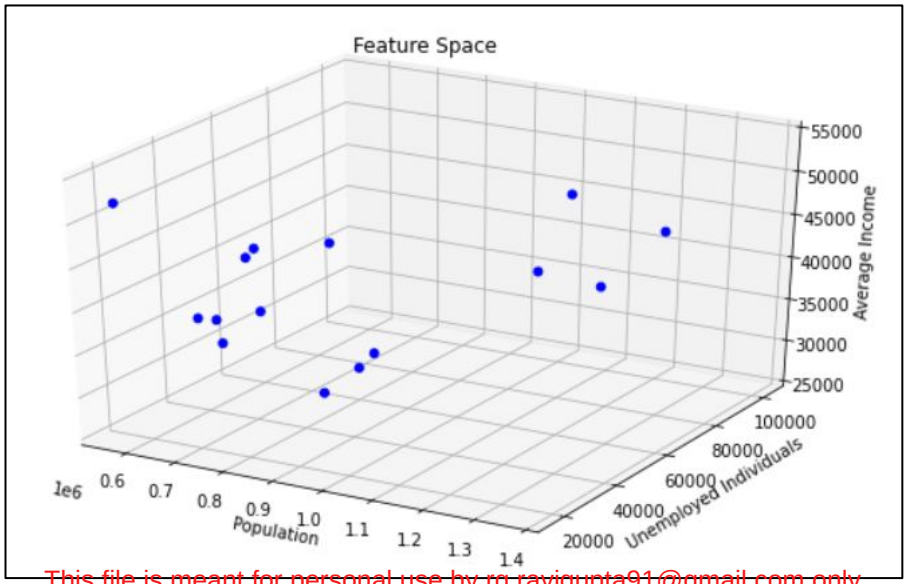
Data

Consider the independent features affecting the crime rate in Canada.

	Population	Unemployed Individuals	Average Income
0	598442	45521	31600
1	1365213	67741	48654
2	857120	36859	29800
3	685742	86100	24510
4	985303	26753	35850
5	620000	54000	41740
6	1052369	94023	46080
7	565412	16401	52100
8	674268	24758	38740
9	856200	39865	46800
10	785411	27568	40200
11	641220	36520	36305
12	1074000	102400	34000
13	654210	45214	42350
14	974100	96520	35800

Data

The original feature space in 3-D.



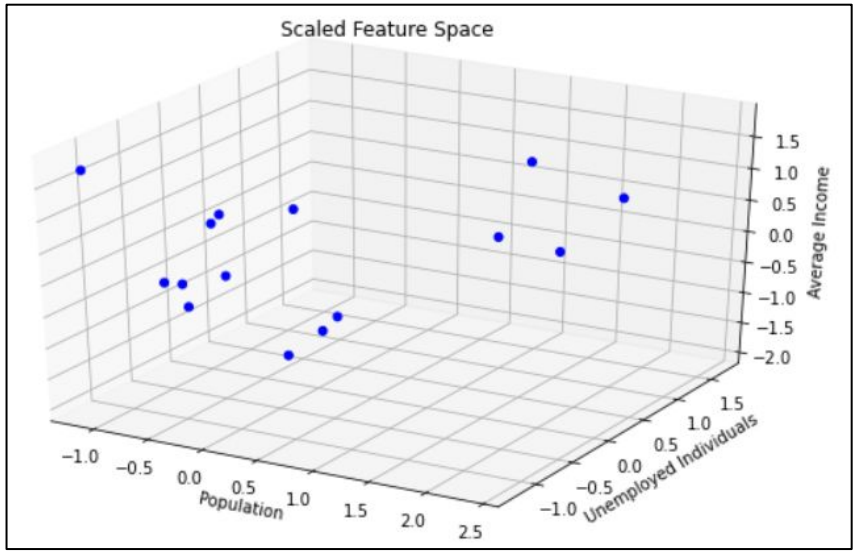
Step 1: standardize the data

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

Where, μ: Mean of the variable

σ: Standard deviation of the variable

x: Original data points



Step 2: covariance matrix

- The diagonal entries of the matrix represents the variance of each variable
- Here, for the standardized data, the variance is 1
- The off-diagonal values exhibits the relation between pair of variables

```
[[ 1.          0.51213849  0.17461469]
 [ 0.51213849  1.         -0.23987409]
 [ 0.17461469 -0.23987409  1.        ]]
```

Step 3: eigenvalues and eigenvectors

Let A be a covariance matrix and λ be the eigenvalue of A.

Eigenvalues are the roots of the equation:

$$det(A - \lambda I) = 0$$

$$det\left(\begin{bmatrix} 1 & 0.51213849 & 0.17461469 \\ 0.51213849 & 1 & -0.23987409 \\ 0.17461469 & -0.23987409 & 1 \end{bmatrix} - \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix}\right) = 0$$

$$det\left(\begin{bmatrix} 1 & 0.51213849 & 0.17461469 \\ 0.51213849 & 1 & -0.23987409 \\ 0.17461469 & -0.23987409 & 1 \end{bmatrix} - \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix}\right) = 0$$

Step 3: eigenvalues and eigenvectors

After solving the equation, we get

$\lambda_1 = 0.35442161,\ \lambda_2 = 1.51704963,\ \lambda_3 = 1.12852876$

Now find the eigenvectors by solving the following equation:

$$(A - \lambda I)x = 0$$

After solving the equation, we get the eigenvectors as:

```
[[ 0.63178091 -0.68151582  0.36930891]
[-0.65513505 -0.72411768 -0.21552643]
[-0.41430778  0.10578172  0.90396863]]
```

Step 4: sort the eigenvalues

Sort the eigenvalues in the descending order.

$\lambda_2 = 1.51704963$,

$\lambda_3 = 1.12852876$,

$\lambda_1 = 0.35442161$

Since there are only 3 eigenvalues, we use the Kaiser criterion to decide the number of principal components.

Here, λ_2 and λ_3 are greater than 1. Thus we select the eigenvectors corresponding to λ_2 and λ_3 as principal components.

Step 5: select the components

- Consider the eigenvectors corresponding to λ_2 and λ_3 as principal components
- These two principal components explain most of the variance in the data

Coefficients of the first two principal components:

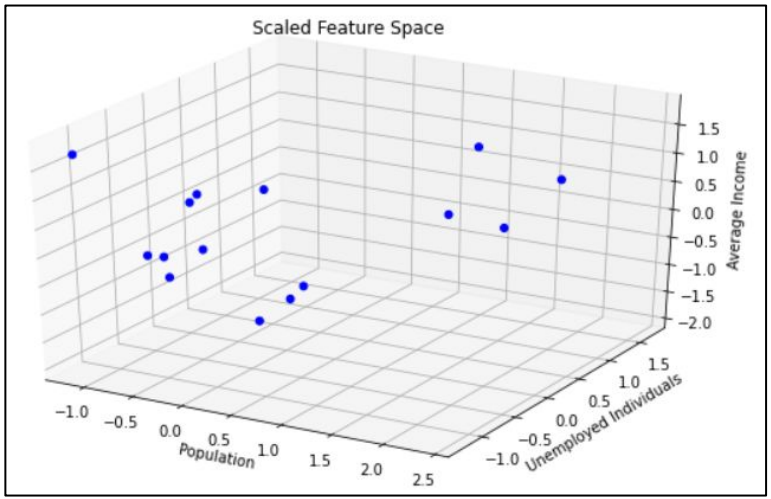
```
[[-0.68151582,  0.36930891],
 [-0.72411768, -0.21552643],
 [ 0.10578172,  0.90396863]]
```

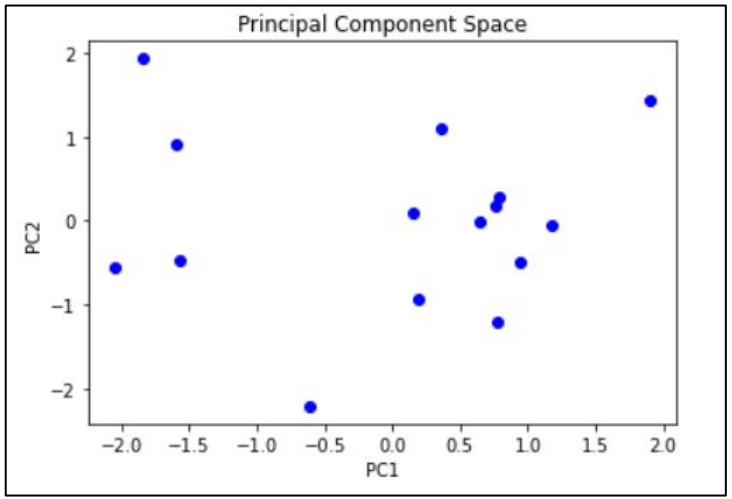
Transform the data

Transform the original data by taking the dot product of the original data with the principal components.

	PC1	PC2
0	0.775810	-1.193580
1	-1.843436	1.928682
2	0.192405	-0.923901
3	-0.605173	-2.204244
4	0.146691	0.084517
5	0.640157	-0.007702
6	-1.601667	0.915383
7	1.893958	1.425421
8	1.170228	-0.060018
9	0.357842	1.087864
10	0.786712	0.274442
11	0.939639	-0.493302
12	-2.046345	-0.558696
13	0.766898	0.186667
14	-1.573717	-0.461534

Summary





Signal to noise ratio(SNR)

Principal Component Analysis - Repeat

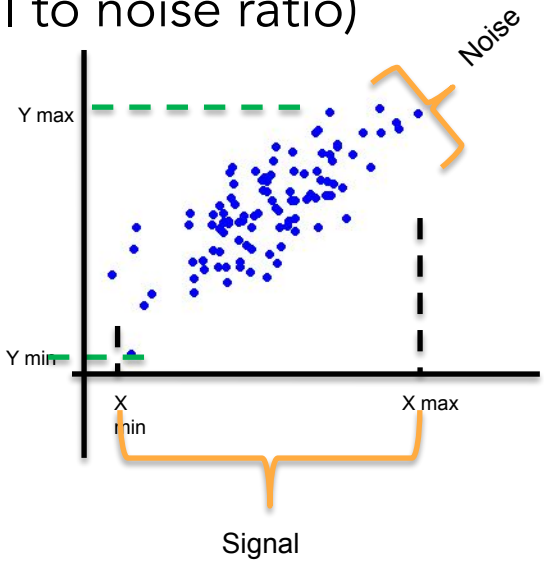
Principal Component Analysis attempts to hit two birds with same stone –

1. It transform existing dimensions to increase the SNR. It creates a new dimension out of the original two

2. Helps Remove redundancy by eliminating attributes /dimensions which contain same information as another attribute.

Principal Component Analysis

(Signal to noise ratio)



- Signal – all valid values for a variable (show between max and min values for x axis and y axis). Represents a valid data
- Noise – The spread of data points across the best fit line. For a given value of x, there are multiple values of y (some on line and some around the line). This spread is due to random factors
- Signal to Noise Ratio – Variance of signal / variance in noise.

```
X_std_df = pd.DataFrame(X_std)
axes = pd.plotting.scatter_matrix(X_std_df)
plt.tight_layout()
```

- Greater the SNR the better the model will be

$$\frac{\sigma^2_{signal}}{\sigma^2_{noise}}$$

Principal Component Covariance Matrix

1. Variance is measured within the dimensions and co-variance is among the dimensions

$$\text{var}(X) = \frac{\sum_{i=1}^{n} (X_i - \overline{X})(X_i - \overline{X})}{(n-1)}$$

2. Express total variance (variance and cross variance between dimensions as a matrix (variance matrix)

$$\text{cov}(X, Y) = \frac{\sum_{i=1}^{n} (X_i - \overline{X})(Y_i - \overline{Y})}{(n-1)}$$

3. Covariance matrix is a mathematical representation of the total variance of individual dimension and across dimensions .

$$C = \begin{bmatrix} \text{cov}(X,X) & \text{cov}(X,Y) & \text{cov}(X,Z) \\ \text{cov}(Y,X) & \text{cov}(Y,Y) & \text{cov}(Y,Z) \\ \text{cov}(Z,X) & \text{cov}(Z,Y) & \text{cov}(Z,Z) \end{bmatrix}$$

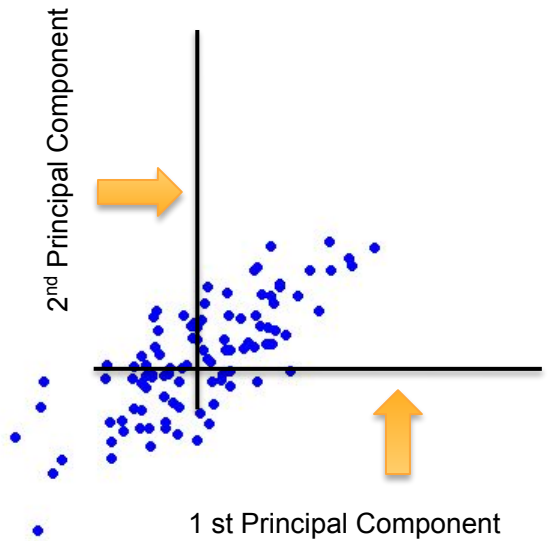
Covariance matrix for three dimensions x,y and z

```
eig_vals, eig_vecs = np.linalg.eig(cov_matrix)
```

Improving SNR through PCA

(Scaling the dimensions)

1. The mean is subtracted from all the points on both dimensions i.e. (xi – xbar) and (yi – ybar)
2. The dimensions are transformed using algebra into new set of dimensions
3. The transformation is a rotation of axes in mathematical space




```
X_std = StandardScaler().fit_transform(X)
eig_vals, eig_vecs = np.linalg.eig(cov_matrix)
```

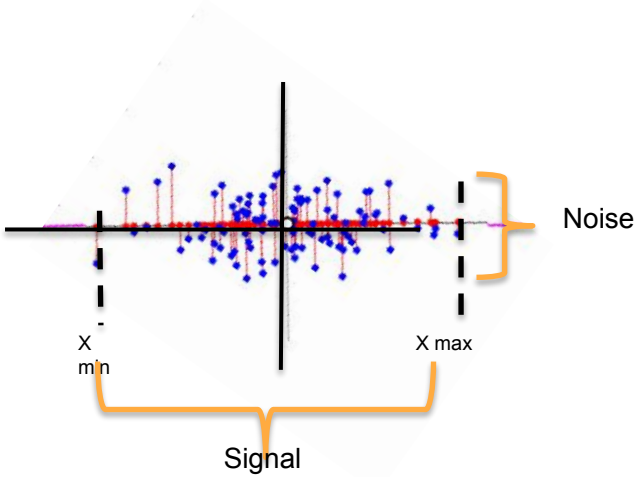
Improving SNR through PCA

(Principal components)

4. The original data points are now represented by the red dots on new dimensions

5. It also introduces error of representation (vertical red lines from the blue dots to corresponding red dots on the new dimension)

6. The axis rotation is done such that the new dimension captures max variance in the data points and also reduces total error of representation




```
print('Eigen Vectors \n%s', eig_vecs)
print('\n Eigen Values \n%s', eig_vals)
```

Improving SNR through PCA

(Principal components)

7. The first principal component is thus a new dimension representing the two original dimensions. We do away with the original two dimensions

Kernel PCA

Limitation of PCA

- PCA is a linear method.

- That is it can only be applied to datasets which are linearly separable.

- But, if we use it to non-linear datasets, we might get a result which may not be the optimal dimensionality reduction.

Overcome PCA limitation

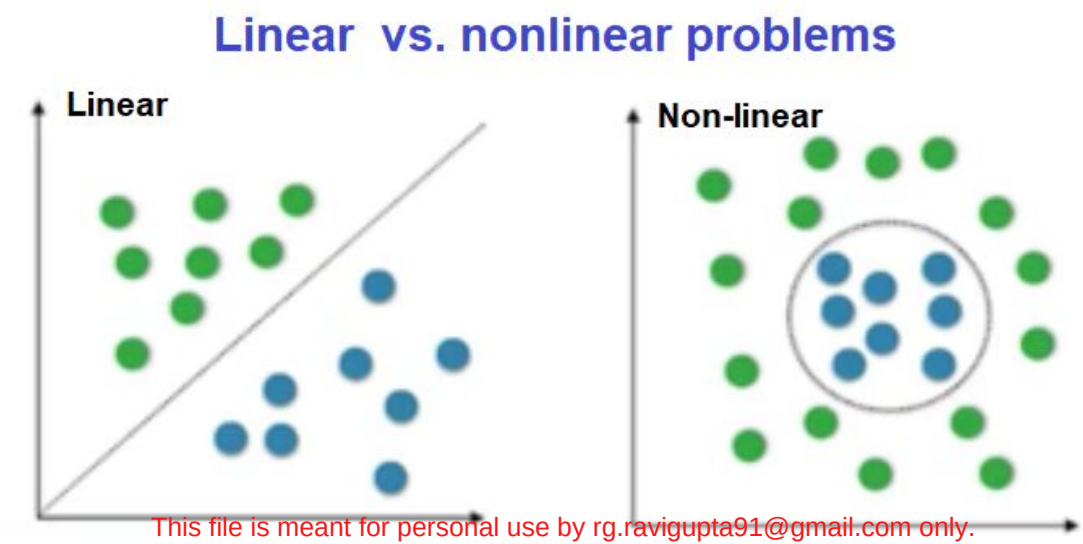
- Kernel PCA uses a kernel function to project dataset into a higher dimensional feature space, where it is linearly separable.

- It is similar to the idea of Support Vector Machines.

- There are various kernel methods like linear, polynomial, and gaussian.

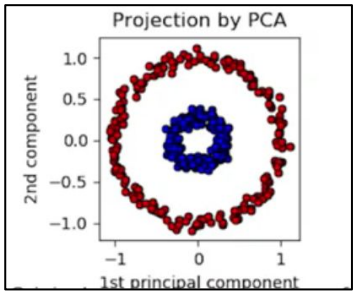
What is Kernel?

- A similarity function, which takes two inputs and returns the similarity between the two

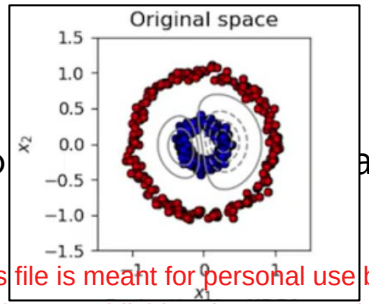


Why is Kernel PCA used ?

- PCA is linear and cannot classify non-linear data effectively.



- We need a method to ar data" pattern.



High – Dimensional Mapping

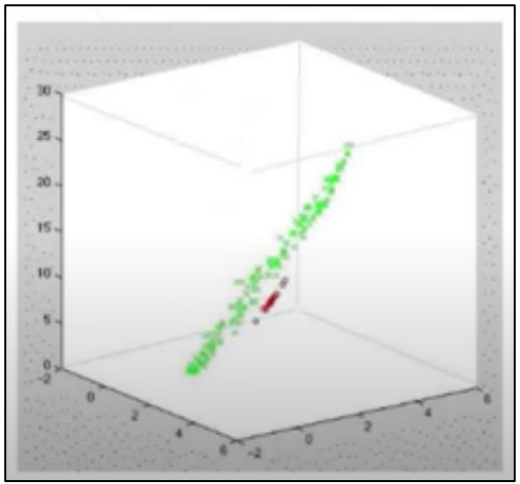
- Mapping data of lower dimension into a high dimension space makes it linear separable.

- $\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$

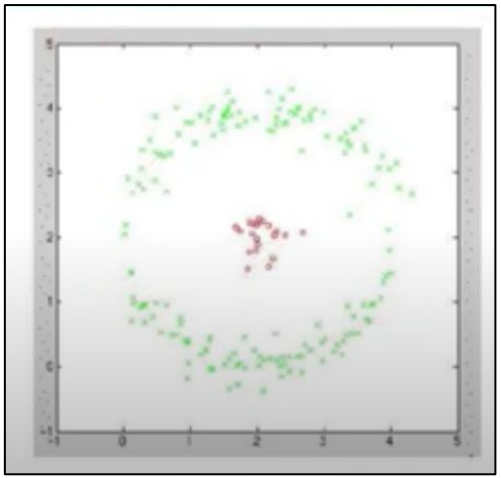
- $(X_1 , X_2) \mapsto (X_1 , X_2 , X_1^2 + X_2^2)$

PCA vs Kernel PCA

- These classes are now linearly separable by simply mapping them.



- These classes are not linearly separable.



How does Kernel PCA work?

- Kernel PCA works on the principle of converting low dimension space to high dimension space

- This is done by using the " Kernel Trick"

Kernel Trick

- $K(x_i, x_j) = x_i \cdot x_j = x_i^T x_j$

  - If we apply high-dimensional mapping on all the data points using a transformation function :

    $\Phi : x \rightarrow \varphi(x)$

  - Then the dot product becomes :

    $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$

Kernel Trick in PCA

- First, We plot the data points in a higher dimensional space
- Extract the Principal Component in that space (PCA) using the Kernel Trick.
- The Result will be nonlinear in the original data space.

Limitation of PCA/Kernel PCA

- Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components (from Wikipedia).

- PCA assumes that the data contains continuous values only and contains no categorical variables.

- It is not possible to apply PCA techniques for dimensionality reduction when the data is composed of categorical variables.

Multiple Correspondence Analysis (MCA)

Overcome PCA/Kernel PCA limitation

- Luckily there exists Multiple Correspondence Analysis (MCA), a PCA-like technique developed for categorical data.

- MCA has been successfully applied for clustering in genomic data or population surveys.

Correspondence Analysis (CA)

- Correspondence analysis is developed for categorical variables that take values of either 0 or 1.

- A categorical variable is binarized in cases where it can take more than 2 values. For example, a categorical variable that varies between [1,2,3] can be categorized such that 1 becomes [1,0,0], 2 becomes [0,1,0] and 3 becomes [0,0,1]. Then we call method as Multiple correspondence analysis.

- The matrix consisting of data transformed using this process is referred as indicator matrix.

Multiple Correspondence Analysis (MCA)

- Multiple correspondence analysis is a simple correspondence analysis carried out on an indicator (or design) matrix with cases as rows and categories of variables as columns.

- MCA is obtained by applying a standard correspondence analysis on this indicator matrix.

- The result is a linear combination of rows (also referred as factors or factor scores) that best describe the data.

Multiple Correspondence Analysis (MCA)

- As several variables that represent the same quantity are introduced, the variance explained by the components is severely underestimated.

- Benzécri and then Greenacre proposed corrections to better estimate the explained variances.

Example

- Let us assume that there are `I` participants on a survey with `J` multiple choices questions with `4` answers. Now let's consider a table and denote it by `X`. So X will have I rows and 4J columns.
- X is the completely disjunctive table of I observations of K categorical variables.
- Assume k-th variable have J_k different levels (categories) and set J = $\sum_{k=1}^{K} J_k$.
- The table X is then a I X J matrix with all coefficient being 0 or 1. Set the sum of all entries of X to be N and introduce Z = X/N.

Explanation

- In an MCA, there are also two special vectors: first , that contains the sums along the rows of Z, and c , that contains the sums along the columns of Z .

- Note D_r = diag(r) and D_c = diag(c) ,
  where the diagonal matrices containing r and c respectively as diagonal.

- With these notations, computing an MCA consists essentially in the singular value decomposition of the matrix:

$$M = D_r^{-1/2}(Z - rc^t)\, D_c^{-1/2}$$

Explanation

- The decomposition of M gives you P, Δ and Q such that M = $P\Delta Q^t$ where P, Q two unitary matrices and Δ is the generalized diagonal matrix of the singular values (with the same shape as Z).

- The positive coefficients of Δ^2 are the eigenvalues of Z.

- The interest of MCA comes from the way observations (rows) and variables (columns) in Z can be decomposed. This decomposition is called a factor decomposition. The coordinates of the observations in the factor space are given by

$$F = D_r^{-1/2} P\Delta$$

Explanation

- The i-th rows of F represent the i -th observation in the factor space. And similarly, the coordinates of the variables (in the same factor space as observations!) are given by

$$G = D_c^{-1/2} Q\Delta$$

Thank You