# PCA

Naw Varsha Pipada
Asst. Prof, Dept. of CSE
Eng. College Bikaner

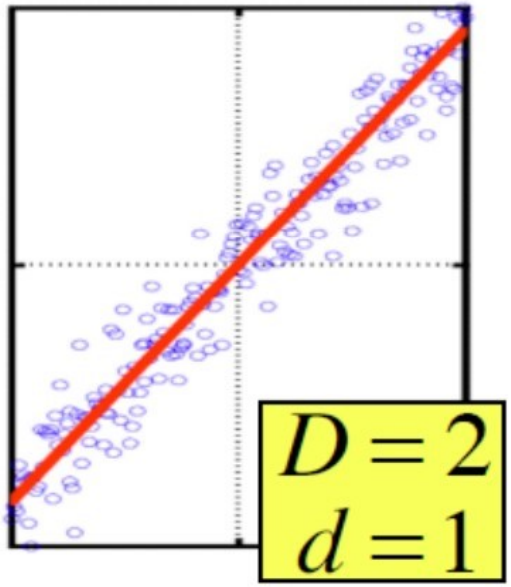
# Principle Component Analysis

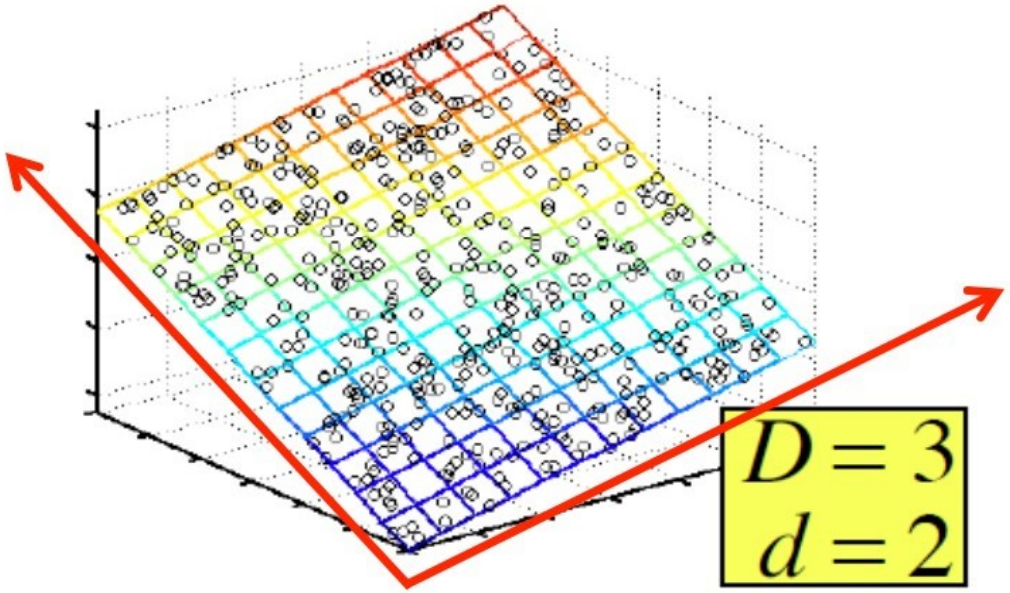
Principal Component Analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

Because smaller data sets are easier to explore and visualize and make analyzing data much easier and faster for machine learning algorithms without extraneous variables to process.

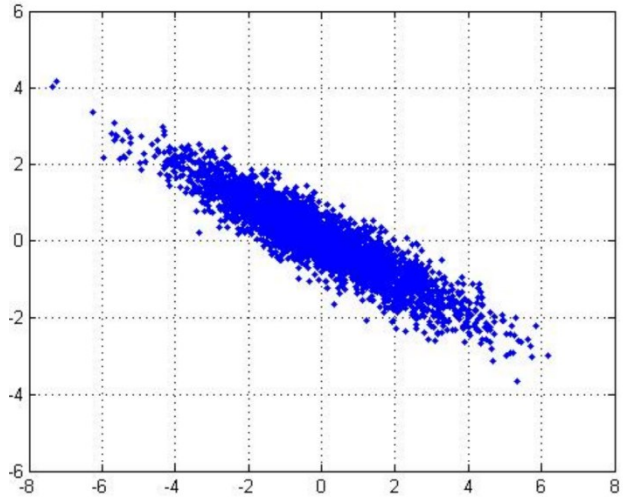
So to sum up, the idea of PCA is simple — reduce the number of variables of a data set, while preserving as much information as possible.

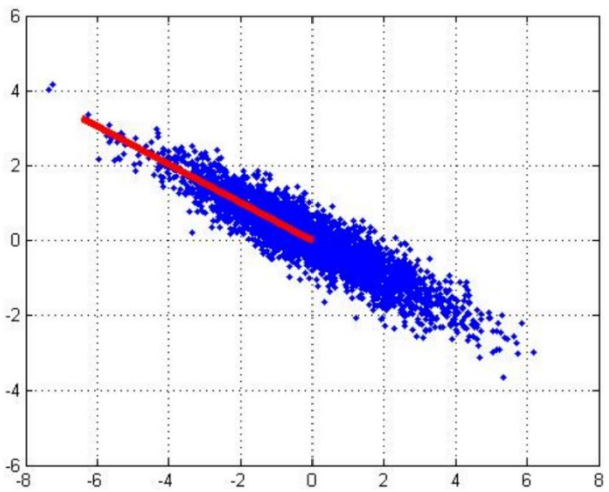
# PCA Example

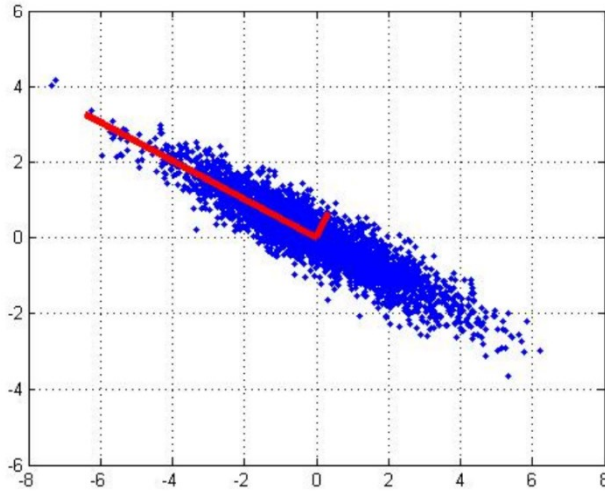


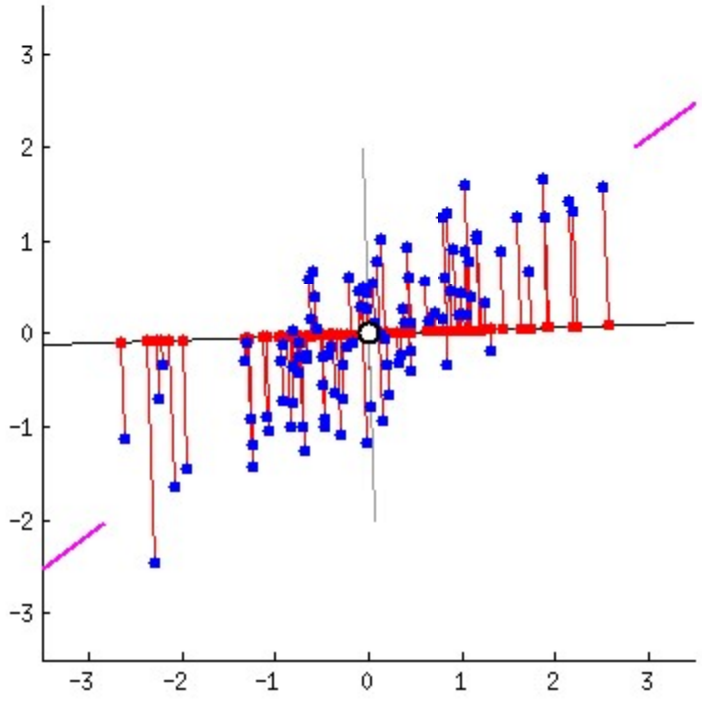
# PCA Example



# PCA Example



# PCA

It is a linear transformation that chooses a new coordinate system for the data set such that

greatest variance by any projection of the data set comes to lie on the first axis (then called the first principal component),

the second greatest variance on the second axis, and so on.

PCA can be used for reducing dimensionality by eliminating the later principal components.

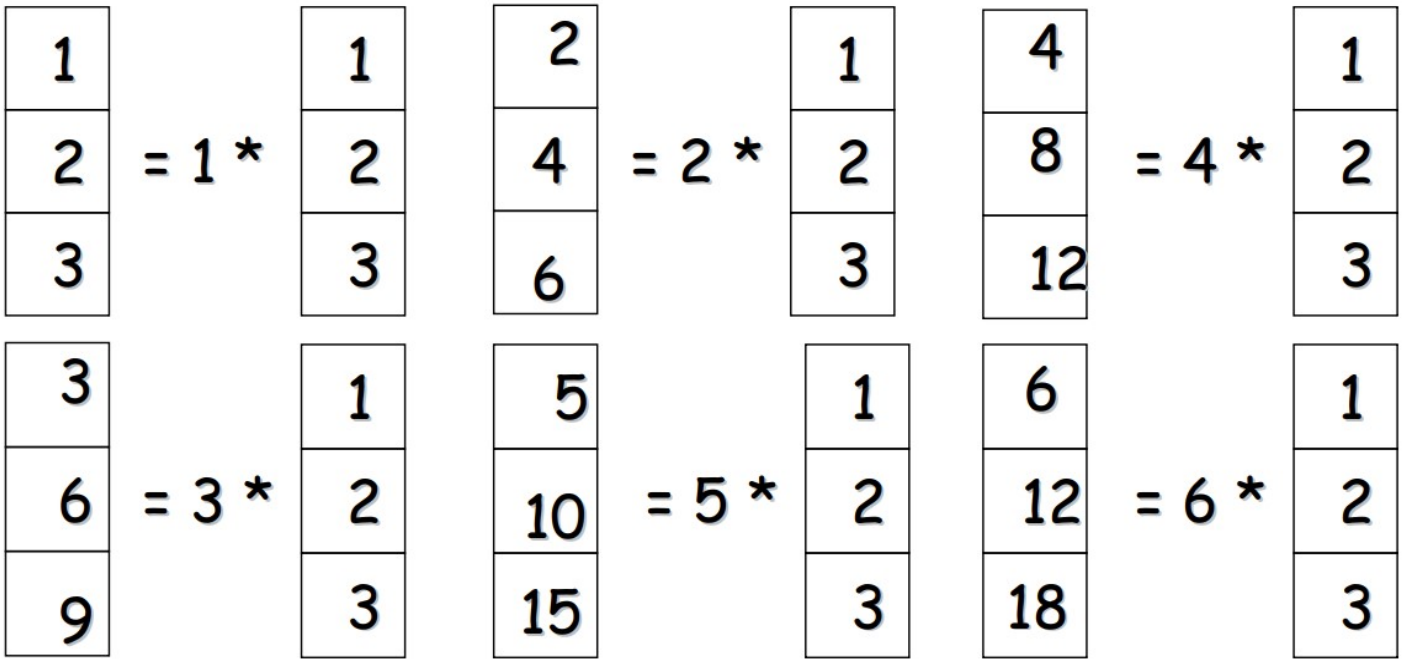
# Why PCA?

Consider the following 3D points

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 4 | 3 | 5 | 6 |
| 2 | 4 | 8 | 6 | 10 | 12 |
| 3 | 6 | 12 | 9 | 15 | 18 |

If each component is stored in a byte,
we need 18 = 3 x 6 bytes

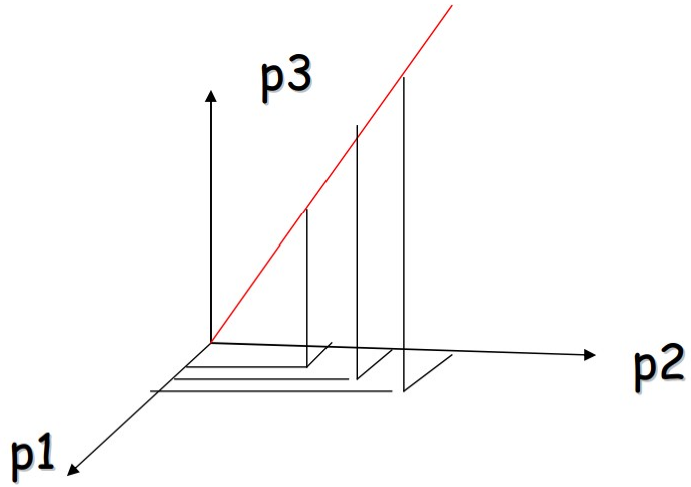
# Why PCA?




They can be stored using only 9 bytes (50% savings!): Store one point Store one point (3 bytes) (3 bytes) + the multiplying constants + the multiplying constants (6 bytes)

# Why PCA: Geometrical Interpretation

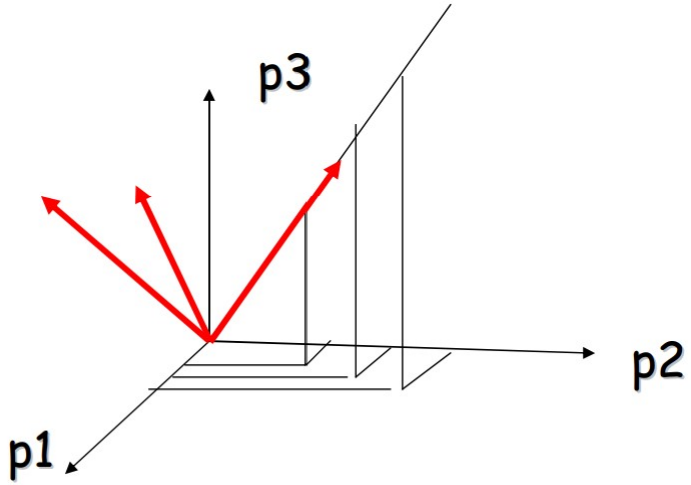
View each point in 3D space.




But in this example, all the points happen to belong to a But in this example, all the points happen to belong to a line: a 1D subspace of the original 3D space.

# Why PCA: Geometrical Interpretation

Consider a new coordinate system where one of the axes is along the direction of the line: is along the direction of the line:




In this coordinate system, every point has only one non-zero coordinate: we only need to store the direction of the line (a 3 bytes image) and the nonzero coordinate for each of the points zero coordinate for each of the points (6 bytes).

# How does PCA work?

We are going to calculate a matrix that summarizes how our variables all relate to one another. (Covariance Matrix)

We'll then break this matrix down into two separate components: direction and magnitude. We can then understand the "directions" of our data and its "magnitude" (or how "important" each direction is).(Eigendecomposition)

We will transform our original data to align with these important directions (which are combinations of our original variables)

By identifying which "directions" are most "important," we can compress or project our data into a smaller space by dropping the "directions" that are the "least important." By projecting our data into a smaller space, we're reducing the dimensionality of our feature space… but because we've transformed our data in these different "directions," we've made sure to keep all original variables in our model!

# Covariance

Variance and Covariance are a measure of the “spread” of a set of points around their center of mass (mean)

Variance – measure of the deviation from the mean for points in one dimension e.g. heights

Covariance as a measure of how much each of the dimensions vary from the mean with respect to each other.

Covariance is measured between 2 dimensions to see if there is a relationship between the 2 dimensions e.g. number of hours studied & marks obtained.

The covariance between one dimension and itself is the variance

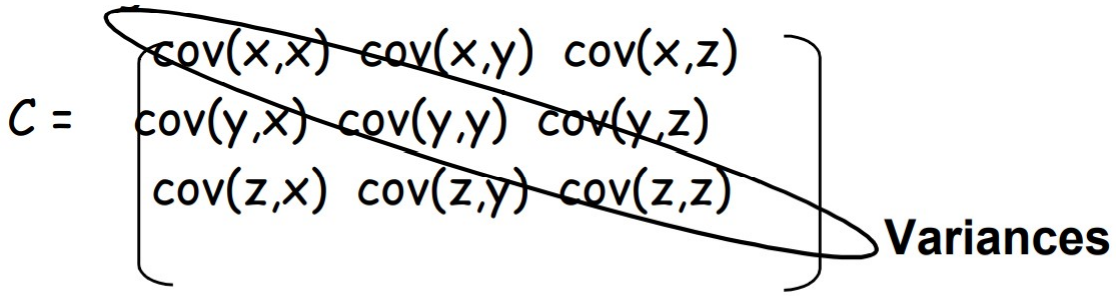
# Covariance

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

So, if you had a 3-dimensional data set (x,y,z), then you could measure the covariance between the x and y dimensions, the y and z dimensions, and the x and z dimensions. Measuring the covariance between x and x , or y and y , or z and z would give you the variance of the x , y and z dimensions respectively.

# Covariance Matrix

Representing Covariance between dimensions as a matrix e.g. for 3 dimensions:




Diagonal is the **variances** of x, y and z

cov(x,y) = cov(y,x) hence matrix is symmetrical about the diagonal

N-dimensional data will result in NxN covariance matrix

# Covariance Matrix

Exact value is not as important as it's sign.

A positive value of covariance indicates both dimensions increase or decrease together

- e.g. as the number of hours studied increases, the marks in that subject increase.

A negative value indicates while one increases the other decreases, o vice-versa

- e.g. active social life vs performance in studies.

If covariance is zero: the two dimensions are independent of each other

- e.g. heights of students vs the marks obtained in a subject

# Why Covariance?

Why bother with calculating covariance when we could just plot the 2 values to see their relationship?

Covariance calculations are used to find relationships between dimensions in high dimensional data sets (usually greater than 3) where visualization is difficult.

# Steps to find PCs

STEP 1: STANDARDIZATION

STEP 2: COVARIANCE MATRIX COMPUTATION

STEP 3: COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

STEP 4: Forming the FEATURE VECTOR

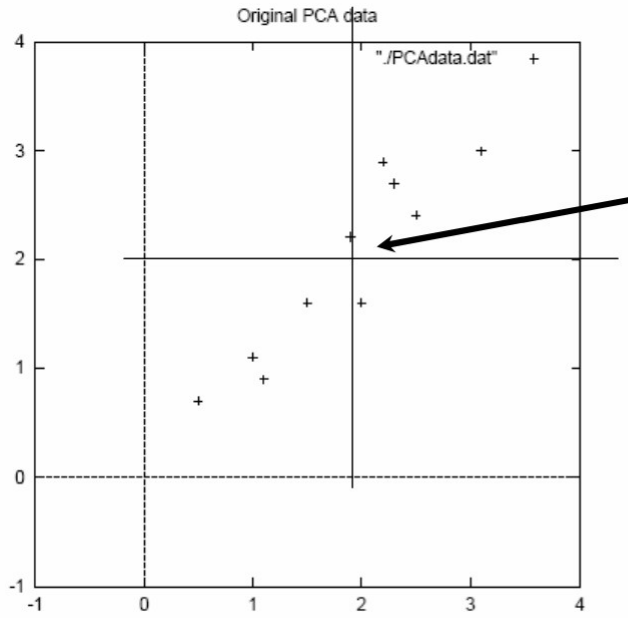
STEP 5: RECAST THE DATA ALONG THE PRINCIPAL COMPONENTS AXES

# Example: STANDARDIZATION

$$z = \frac{value - mean}{standard\ deviation}$$

Once the standardization is done, all the variables will be transformed to the same scale.

# Example: STANDARDIZATION




mean

this becomes the new origin of the data from now on

DATA:

| x | y |
|---|---|
| 2.5 | 2.4 |
| 0.5 | 0.7 |
| 2.2 | 2.9 |
| 1.9 | 2.2 |
| 3.1 | 3.0 |
| 2.3 | 2.7 |
| 2 | 1.6 |
| 1 | 1.1 |
| 1.5 | 1.6 |
| 1.1 | 0.9 |

# Example: COVARIANCE MATRIX COMPUTATION

$$\text{cov} = \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix}$$

since the non-diagonal elements in this covariance matrix are positive, we should expect that both the x and y variable increase together.

# Example: EIGENVECTORS AND EIGENVALUES

$$\text{eigenvalues} = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$\text{eigenvectors} = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -735178656 \end{pmatrix}$$

# Example: FEATURE VECTOR

FeatureVector = ($eig_1$ $eig_2$ $eig_3$ … $eig_n$)

We can either form a feature vector with both of the eigenvectors:

$$\begin{pmatrix} -.677873399 & -.735178656 \\ -.735178656 & .677873399 \end{pmatrix}$$

or, we can choose to leave out the smaller, less significant component and only have a single column:

$$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix}$$

# Example: Transformed Data

In this step, reorient the data from the original axes to the ones represented by the principal components (hence the name Principal Components Analysis).

This can be done by multiplying the transpose of the original data set by the transpose of the feature vector.

$$FinalDataSet = FeatureVector^T * StandardizedOriginalDataSet^T$$

# Pros and Cons

## Pros

- Removes Correlated Features
- Improves Algorithm Performance
- Reduces Overfitting
- Improves Visualization

## Cons

- Independent variables become less interpretable
- Data standardization is must before PCA
- Information Loss

Thanks