ESS2222

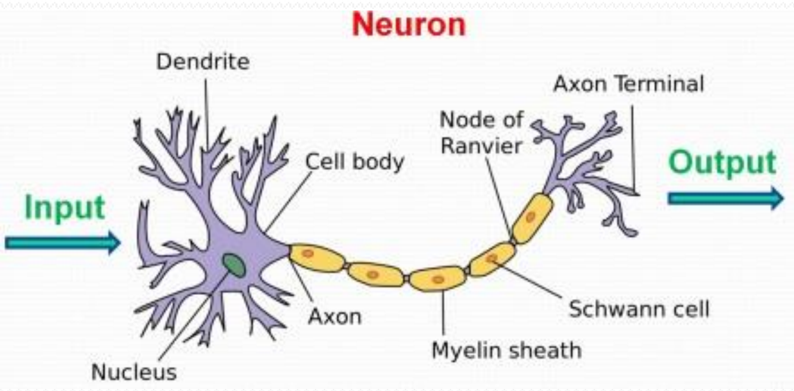
Lecture 8 – Mining of Massive Data - Dimensionality Reduction

Hosein Shahnas

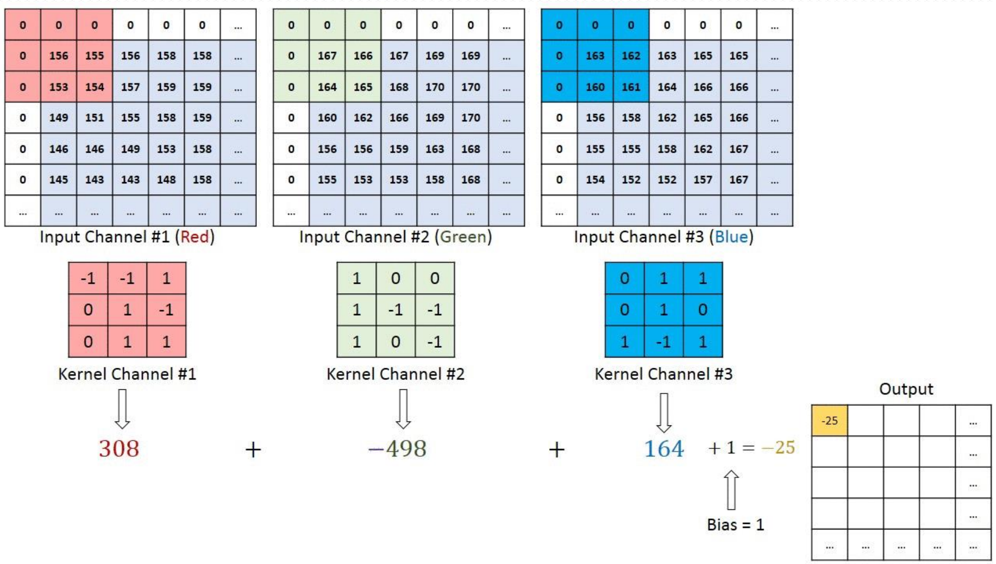
University of Toronto, Department of Earth Sciences,

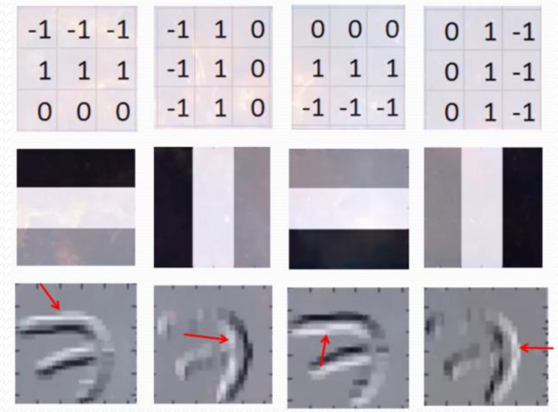
Outline

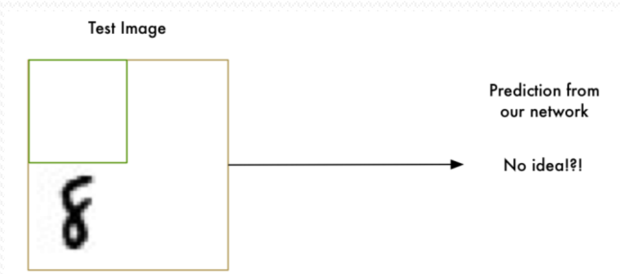
- ❑ **Principal Component Analysis**
- ❑ **Dimensionality Reduction**

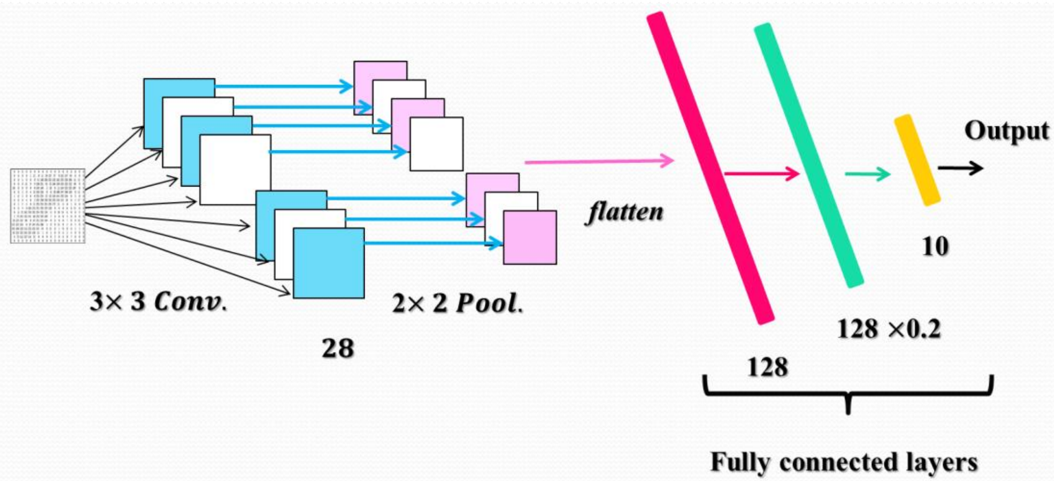


Review of Lecture 7







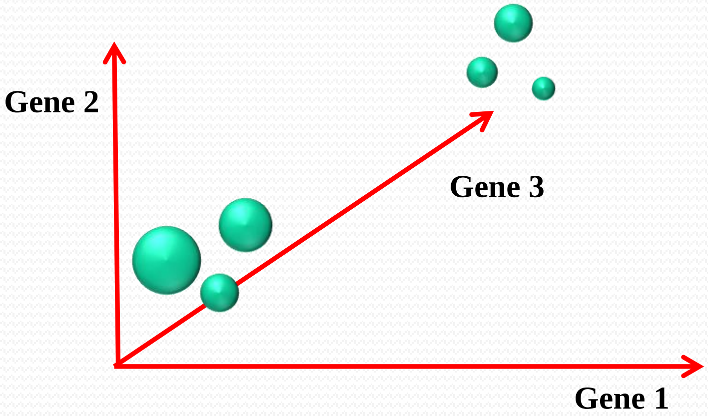


Principal Component Analysis

Principal component analysis (PCA)

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.

		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1
Feature 3	Gene 3	12	9	10	2.5	1.3	2

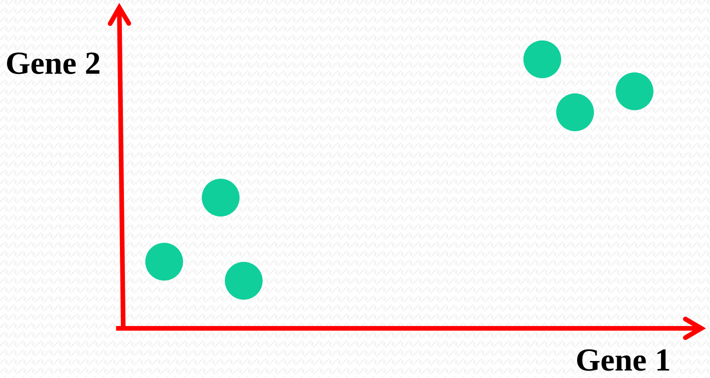


How can we take three or more features (three or more dimensional feature data) and make a lower PC representation (lower dimension)?

Principal Component Analysis

Let’s assume we have only two genes (two features):

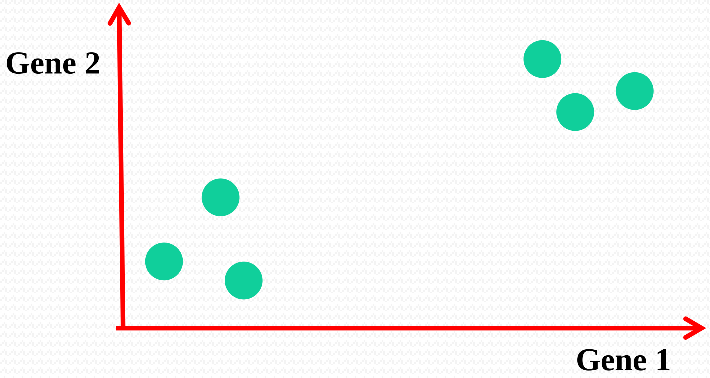
		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1

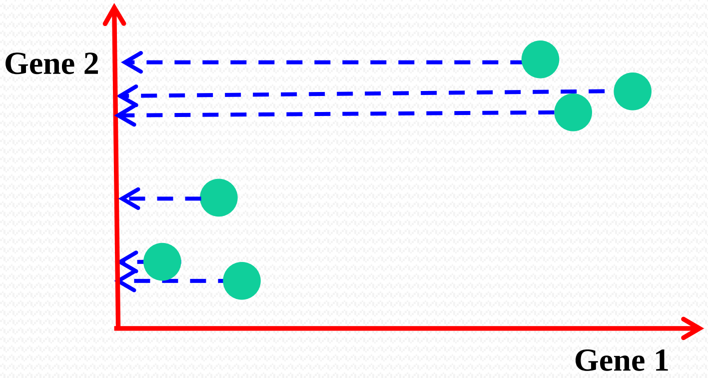


Principal Component Analysis

Let's assume we have only two genes (two features):

		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1

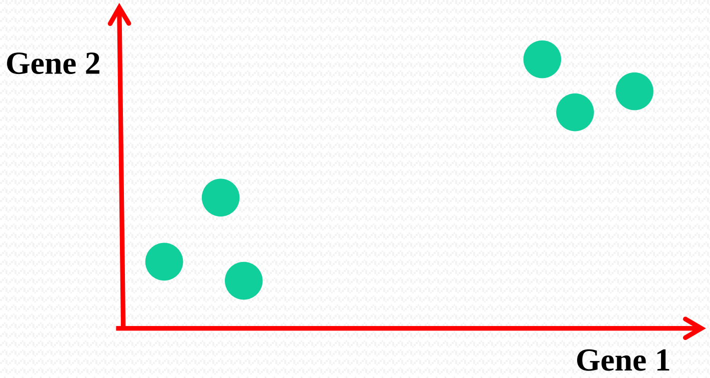


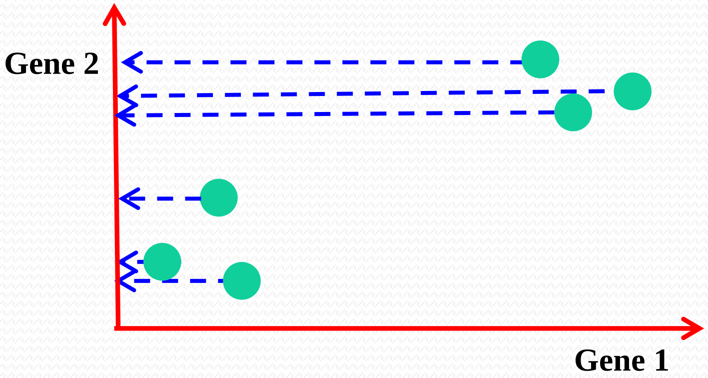


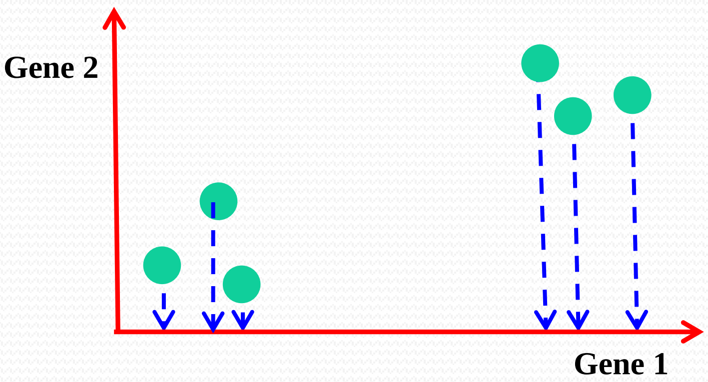
Principal Component Analysis

Let's assume we have only two genes (two features):

		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1



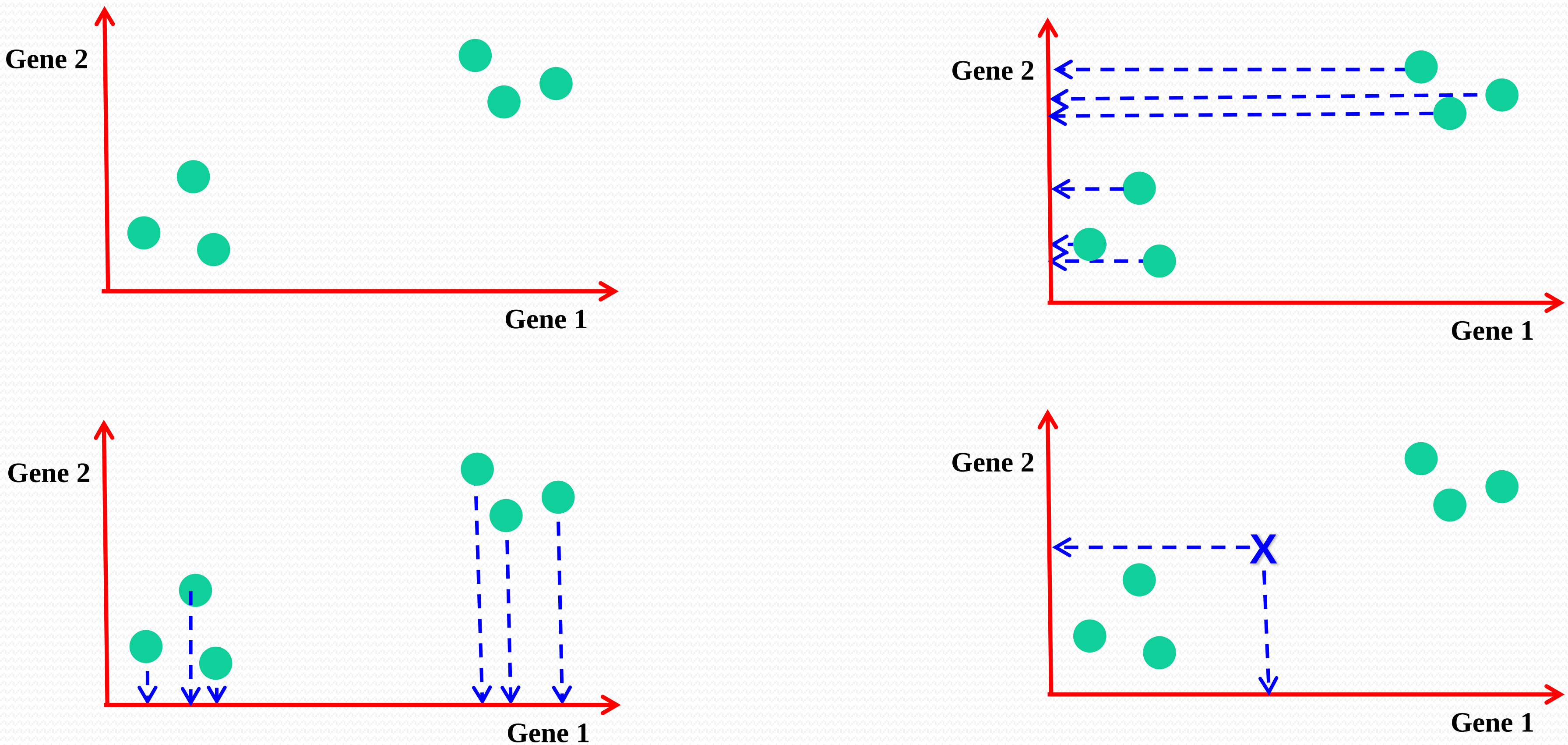




Principal Component Analysis

Let's assume we have only two genes (two features):

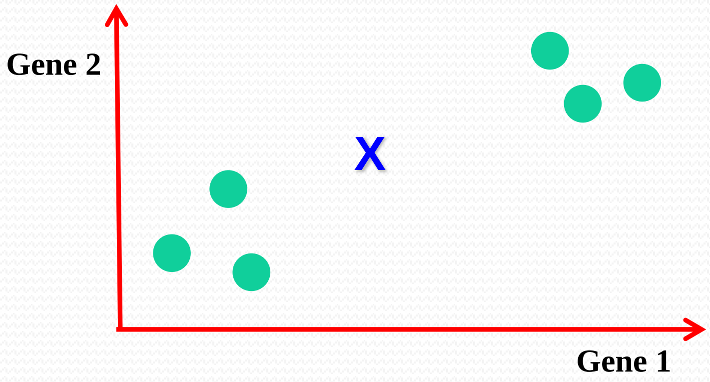
		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1



Principal Component Analysis

Let's assume we have only two genes (two features):

		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1



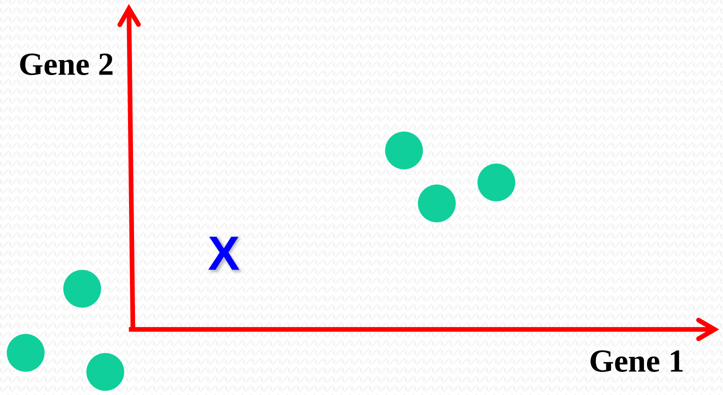
a) Find the center of mean

Principal Component Analysis

Let's assume we have only two genes (two features):

		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1

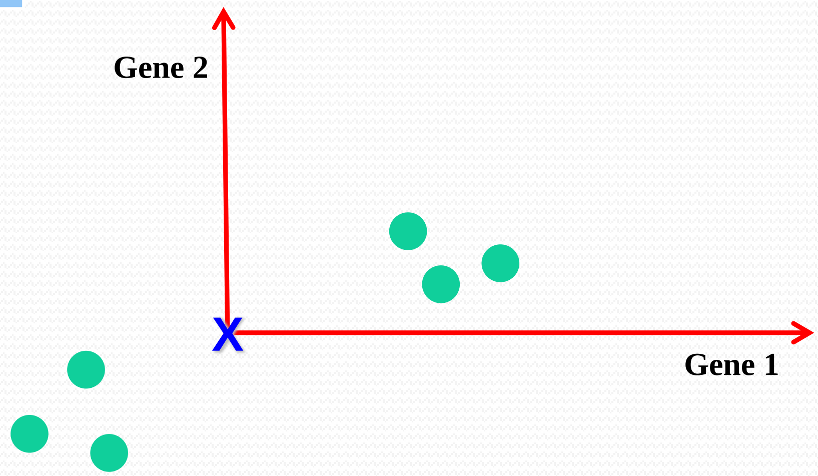
a) Find the center of mean
b) Move the center to the origin



Principal Component Analysis

Let's assume we have only two genes (two features):

		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1

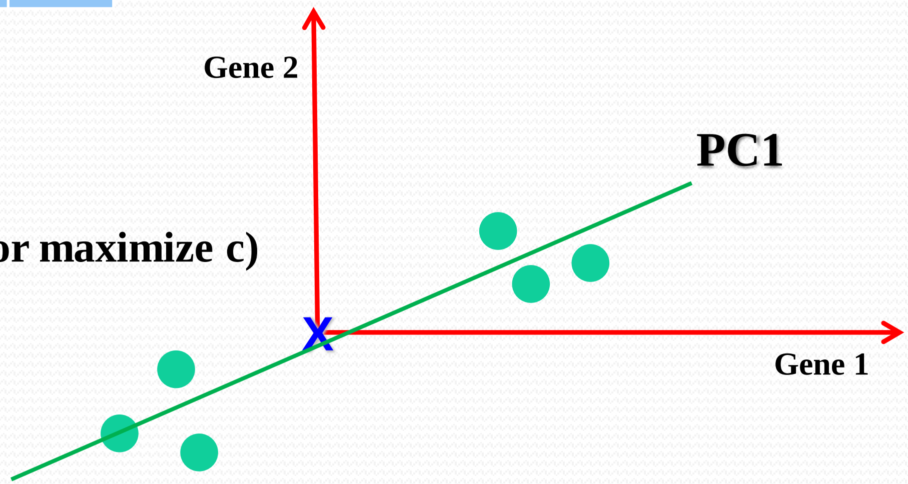


a) Find the center of mean
b) Move the center to the origin

Principal Component Analysis

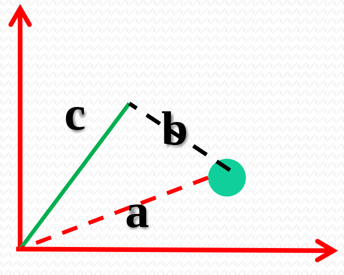
Let's assume we have only two genes (two features):

		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1



a) Find the center of mean
b) Move the center to the origin
c) Fit the data points to a line (minimize b or maximize c)

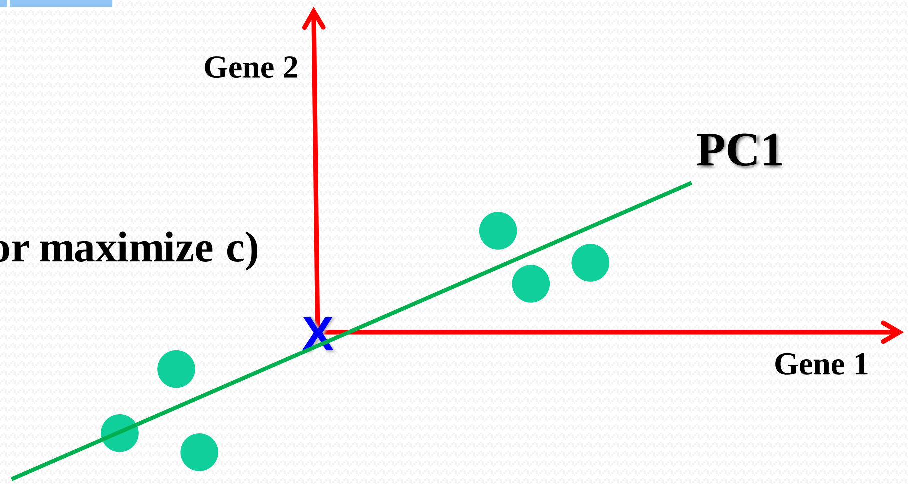
Max $\sum_i \; c_i^2$



Principal Component Analysis

Let's assume we have only two genes (two features):

		Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Feature 1	Gene 1	10	11	8	3	2	1
Feature 2	Gene 2	6	4	5	3	2.8	1



a) Find the center of mean

b) Move the center to the origin

c) Fit the data points to a line (minimize b or maximize c)

Max $\sum_i c_i^2$

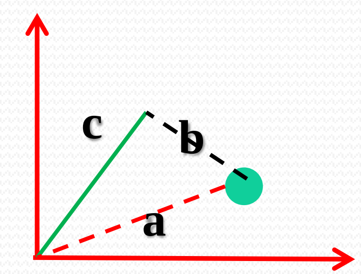
PC1: principal component 1, which is a linear combination of feature 1 and 2.

A unit vector along PC1 is an Eigenvector for PC1.

SSD = Max $\sum_i c_i^2$ Eigenvalue for PC1

$\sqrt{SSD}$ Singular value for PC1

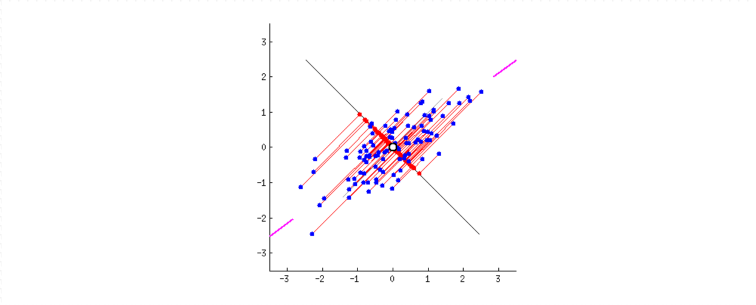
SSD: Sum of squares od distances

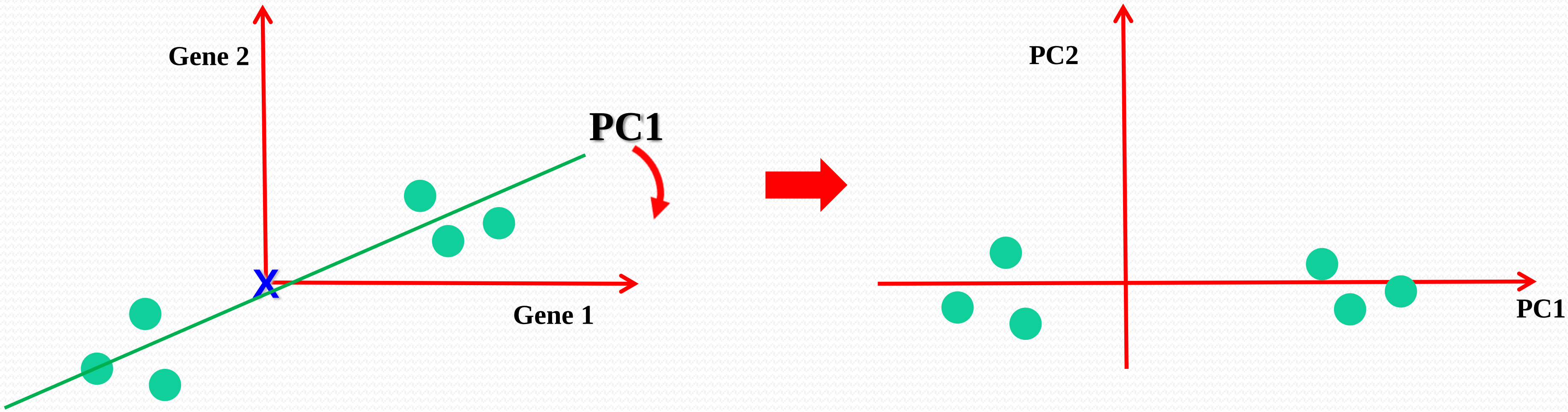


Principal Component Analysis

$\frac{SSD_{PC1}}{n-1}$ variation for PC1

$\frac{SSD_{PC2}}{n-1}$ variation for PC2



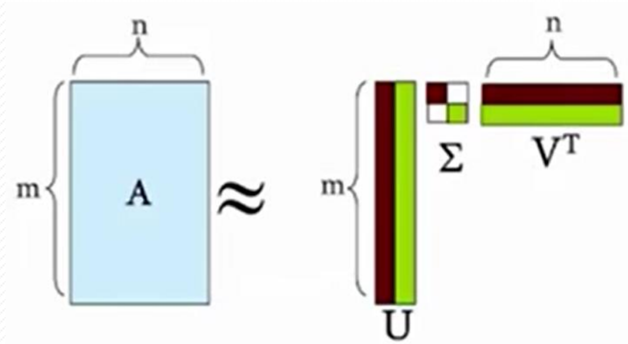


For example if:

$\frac{SSD_{PC1}}{n-1} = 16$, $\frac{SSD_{PC2}}{n-1} = 2$ Var(PC1) + Var(PC2) = 18,

PC1 accounts for 16/18 = 89% of variations (information) around PC's .

Singular Value Decomposition (SVD)

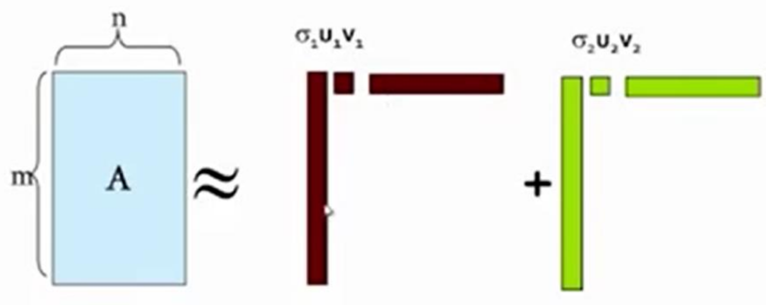


$$A_{[mn]} = U_{[mr]}\, \Sigma_{[rr]}\, V_{[nr]}^{T}$$

m: number of rows
n: number of columns

A: Input data
U: The left-singular vectors of A are a set of orthonormal eigenvectors of AA^T.
V: The right-singular vectors of A are a set of orthonormal eigenvectors of A^TA.
Σ : Singular values

The non-zero singular values of A (found on the diagonal entries of Σ) are the square roots of the non-zero eigenvalues of both AA^T and A^TA.

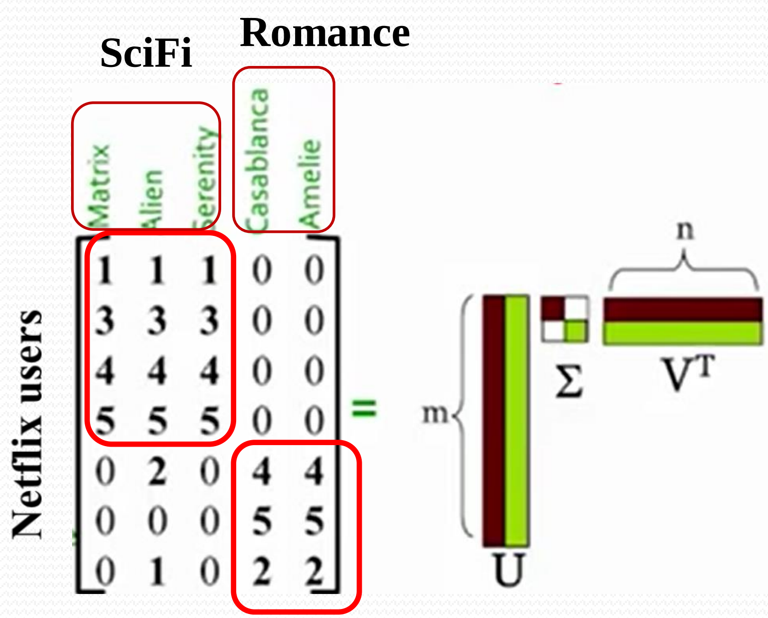


Dimensionality Reduction

It is always possible to decompose a real matrix A into **A = U Σ V^T**,where

U, **Σ**, V: unique (decomposition is unique)
U,V: orthonormal ($U^TU = I$; $V^TV = I$)
Σ : diagonal
Entries are (singular values) are positive ($\sigma_1 \geq \sigma_2 \geq \ldots. \sigma_r \geq 0$).

Example:
Netflix users ranking the movies



Dimensionality Reduction

What do we learn from SVD decomposition?

Columns; Concept

Strength of SciFi lovers

$$
\underset{A}{\text{Users}\;\overset{\text{movies}}{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}}}
=
\underset{U}{\overset{\text{concepts}}{\begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix}}}
\times
\underset{\Sigma}{\begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix}}
\times
\underset{V^T}{\overset{\text{movies}}{\begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}}}
\begin{matrix} \text{SciFi-concept} \\ \text{Romance-concept} \\ \text{Remaining} \end{matrix}
$$

Noise

SciFi – concept

Romance – concept

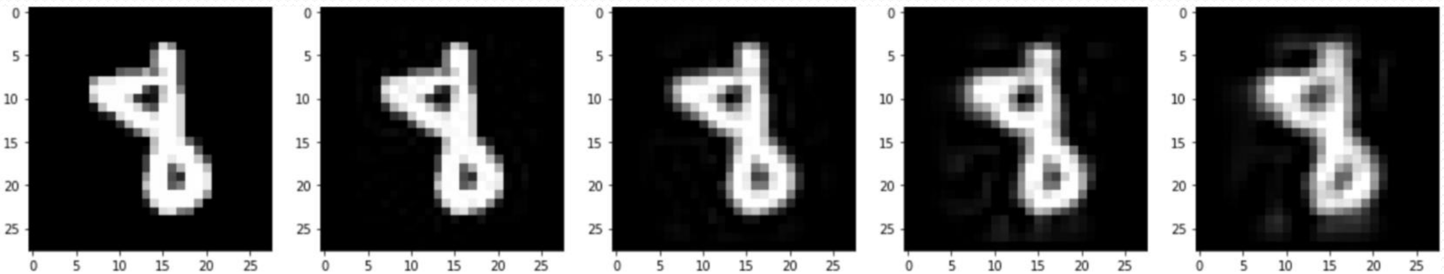
Remaining– concept

The first four users strongly correspond to SciFi – concept,
The last three users heavily correspond to Romance – concept,

U: "user-to-concept" similarity matrix
V: "movie-to-concept" similarity matrix

Principal Component Analysis

784	331	154	87	59	Components



original	99%	95%	90%	85%	Explained Variance

PCA for Visualization

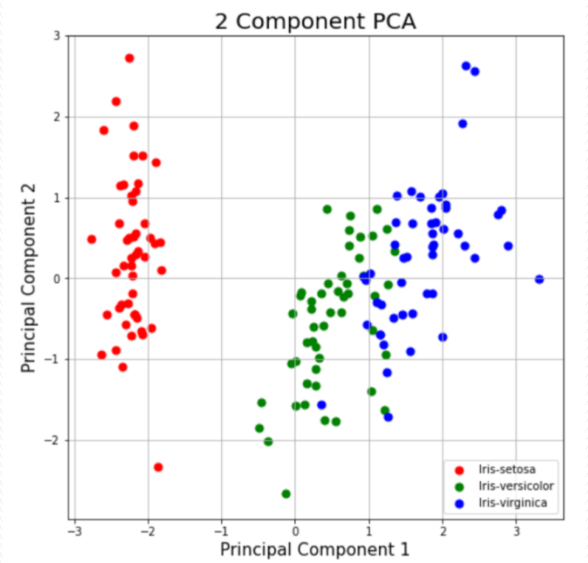
Example 1: Reduction of the number of features in Iris-problem from 4 to 2:

Standardization

	sepal length	sepal width	petal length	petal width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Standardization →

	sepal length	sepal width	petal length	petal width
0	-0.900681	1.032057	-1.341272	-1.312977
1	-1.143017	-0.124958	-1.341272	-1.312977
2	-1.385353	0.337848	-1.398138	-1.312977
3	-1.506521	0.106445	-1.284407	-1.312977
4	-1.021849	1.263460	-1.341272	-1.312977



2 Component PCA Graph

PCA

	sepal length	sepal width	petal length	petal width
0	-0.900681	1.032057	-1.341272	-1.312977
1	-1.143017	-0.124958	-1.341272	-1.312977
2	-1.385353	0.337848	-1.398138	-1.312977
3	-1.506521	0.106445	-1.284407	-1.312977
4	-1.021849	1.263460	-1.341272	-1.312977

PCA
(2 components) →

	principal component 1	princial component 2
0	-2.264542	0.505704
1	-2.086426	-0.655405
2	-2.367950	-0.318477
3	-2.304197	-0.575368
4	-2.388777	0.674767

Concatenating

	principal component 1	principal component 2
0	-2.264542	0.505704
1	-2.086426	-0.655405
2	-2.367950	-0.318477
3	-2.304197	-0.575368
4	-2.388777	0.674767

principalDf

	target
0	Iris-setosa
1	Iris-setosa
2	Iris-setosa
3	Iris-setosa
4	Iris-setosa

df[['target']]

pd.concat(axis = 1) →

	principal component 1	principal component 2	target
0	-2.264542	0.505704	Iris-setosa
1	-2.086426	-0.655405	Iris-setosa
2	-2.367950	-0.318477	Iris-setosa
3	-2.304197	-0.575368	Iris-setosa
4	-2.388777	0.674767	Iris-setosa

finalDf

PCA for Visualization

```
#https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60
#from sklearn.datasets import fetch_mldata
#mnist = fetch_mldata('MNIST original')
import sys
#============================================ PCA for Data Visualization
#============================================ import data
print('========================================= 1')
import pandas as pd
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
# load dataset into Pandas DataFrame
df = pd.read_csv(url, names=['sepal length','sepal width','petal length','petal width','target'])
#df = pd.read_csv(url, names=['sepal length','sepal width','target'])

print ('df = ', df)
print('========================================= 1')
#============================================ import data
#============================================ set x and y and Standardize
print('========================================= 2')
from sklearn.preprocessing import StandardScaler
features = ['sepal length', 'sepal width', 'petal length', 'petal width']
# Separating out the features
x = df.loc[:, features].values
# Separating out the target
y = df.loc[:,['target']].values
# Standardizing the features
x = StandardScaler().fit_transform(x)
#print ('y = ', y)
#print ('x = ', x)
print ('y.shape = ', y.shape)
print ('x.shape = ', x.shape)
print('========================================= 2')
#============================================ set X and y and Standardize

#========================================================== Principal Component Analysis 1

print('========================================= 3')
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
print('pca.explained_variance_ratio_ = ', pca.explained_variance_ratio_ )
```

PCA for Visualization

```python
principalDf = pd.DataFrame(data = principalComponents
             , columns = ['principal component 1', 'principal component 2'])
#print('principalComponents = ',principalComponents)
print ('principalComponents.shape = ', principalComponents.shape)
print ('principalDf.shape = ', principalDf.shape)

#print ('principalComponents = ', principalComponents)
print ('principalDf.head(3) = ', principalDf.head(3))

print('========================================= 3')

#=========================================================== Principal Component Analysis 1

#=========================================================== Principal Component Analysis 2
'''
print('========================================= 4')
# scikit-learn choose the minimum number of principal components such that 95% of the variance is retained.
from sklearn.decomposition import PCA
pca = PCA(0.98)
principalComponents = pca.fit_transform(x)
n_dim = principalComponents[1].shape
print('pca.explained_variance_ratio_ = ', pca.explained_variance_ratio_ )
Feature_size = int(principalComponents.size/y.size)
print('Feature_size = ', Feature_size)

if (Feature_size==1):
    columns0 = ['principal component 1']
if (Feature_size==2):
    columns0 = ['principal component 1', 'principal component 2']
if (Feature_size==3):
    columns0 = ['principal component 1', 'principal component 2', 'principal component 3']
if (Feature_size==4):
    columns0 = ['principal component 1', 'principal component 2', 'principal component 3', 'principal component 4']

principalDf = pd.DataFrame(data = principalComponents
             , columns = columns0)
#print('principalComponents = ',principalComponents)
print ('principalComponents.shape = ', principalComponents.shape)
print ('principalDf.shape = ', principalDf.shape)
```

PCA for Visualization

```

#print ('principalComponents = ', principalComponents)
print ('principalDf.head(3) = ', principalDf.head(3))
print('principalComponents.shape = ', len(principalDf.columns))
print('========================================= 4')
'''
#=========================================================== Principal Component Analysis 2

#============================================= Concatenating along axis = 1 (column)
print('========================================= 5')
 # combine reduced features and targets (3 columns: 2 reduced features + 1 taget)
finalDf = pd.concat([principalDf, df[['target']]], axis = 1)
#print('finalDf = ', finalDf)
print('finalDf.shape = ', finalDf.shape)
print('========================================= 5')
#============================================= Concatenating along axis = 1 (column)
#============================================= Visualize 2D Projection
import matplotlib.pyplot as plt
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('Principal Component 1', fontsize = 15)
ax.set_ylabel('Principal Component 2', fontsize = 15)
ax.set_title('2 component PCA', fontsize = 20)
targets = ['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
colors = ['r', 'g', 'b']
for target, color in zip(targets,colors):
    indicesToKeep = finalDf['target'] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'principal component 1']
               , finalDf.loc[indicesToKeep, 'principal component 2']
               , c = color
               , s = 50)
ax.legend(targets)
ax.grid()
#============================================= Visualize 2D Projection
explained_var = pca.explained_variance_ratio_
print('explained_var = ', explained_var)


```