GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

LECTURE 12: DIMENSIONALITY REDUCTION

I. DIMENSIONALITY REDUCTION
II. PRINCIPAL COMPONENTS ANALYSIS (PCA)
III. SINGULAR VALUE DECOMPOSITION
IV. KERNEL METHODS IN PCA

EXERCISE:
IV. DIMENSIONALITY REDUCTION IN SCIKIT-LEARN

I. DIMENSIONALITY REDUCTION

Problem: Consider this.

Q: What is dimensionality reduction?

Q: What is dimensionality reduction?

A: A set of techniques for reducing the size (in terms of features, records, and/or bytes) of the dataset under examination.

Q: What is dimensionality reduction?

A: A set of techniques for reducing the size (in terms of features, records, and/or bytes) of the dataset under examination.

In general, the idea is to regard the dataset is a matrix and to decompose the matrix into simpler, meaningful pieces.

Q: What is dimensionality reduction?

A: A set of techniques for reducing the size (in terms of features, records, and/or bytes) of the dataset under examination.

In general, the idea is to regard the dataset is a matrix and to decompose the matrix into simpler, meaningful pieces.

Dimensionality reduction is frequently performed as a pre-processing step before another learning algorithm is applied.

Q: What are the motivations for dimensionality reduction?

Q: What are the motivations for dimensionality reduction?

The number of features in our dataset can be difficult to manage, or even misleading (eg, if the relationships are actually simpler than they appear).

For example, suppose we have a dataset with some features that are related to each other.

For example, suppose we have a dataset with some features that are related to each other.

Ideally, we would like to eliminate this redundancy and consolidate the number of variables we're looking at.

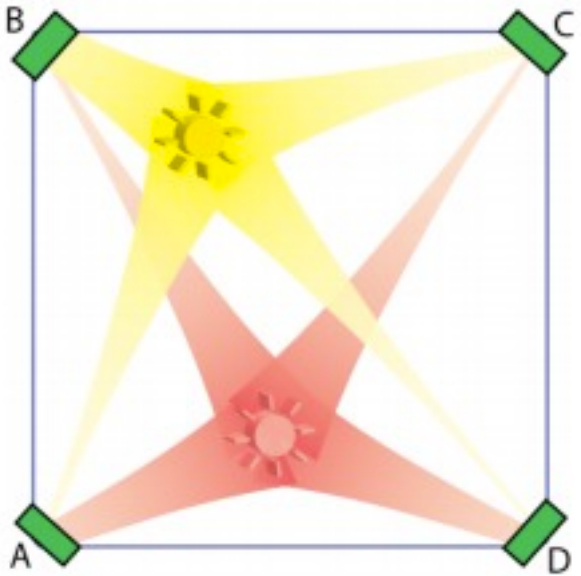
For example, suppose we have a dataset with some features that are related to each other.

Ideally, we would like to eliminate this redundancy and consolidate the number of variables we're looking at.

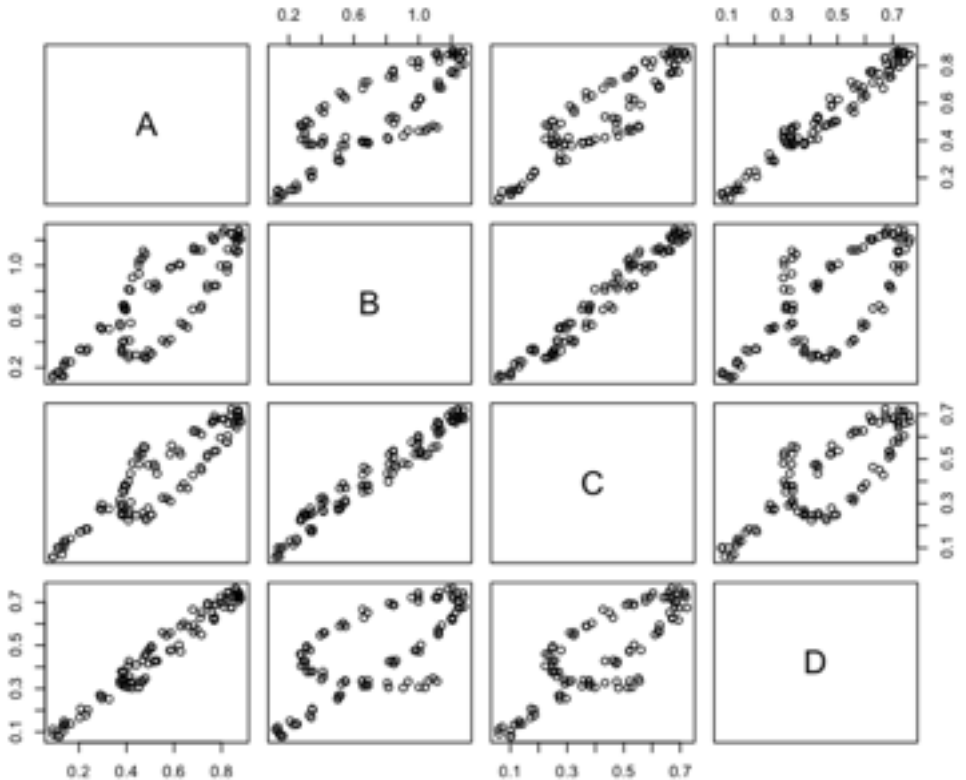
If these relationships are linear, then we can use well-established techniques like PCA/SVD.

Problem: Consider this.

Say we have a large room that contains m lights with unique light patterns and n cameras recording them. Using what the cameras record, how do we determine how many lights there are in the room?



EXAMPLE: COMPARING DATA BETWEEN LIGHT CAPTURE CAMERAS



ASIDE: CURSE OF DIMENSIONALITY

The complexity that comes with a large number of features is due in part to the **curse of dimensionality**.

ASIDE: CURSE OF DIMENSIONALITY

The complexity that comes with a large number of features is due in part to the **curse of dimensionality**.

Namely, the sample size needed to accurately estimate a random variable taking values in a dimensional feature space grows exponentially with the number of features (almost).

ASIDE: CURSE OF DIMENSIONALITY

The complexity that comes with a large number of features is due in part to the **curse of dimensionality**.

Namely, the sample size needed to accurately estimate a random variable taking values in a dimensional feature space grows exponentially with the number of features (almost).

(More precisely, the sample size grows exponentially with $l \leq features$, *the dimension of the manifold embedded in the feature space).*

ASIDE: CURSE OF DIMENSIONALITY

Most of the points in the space are "far" from each other.

Most of the points in the space are "far" from each other.

This illustrates the fact that local methods will break down in these circumstances (eg, in order to collect enough neighbors for a given point, you need to expand the radius of the neighborhood so far that locality is not preserved).

Most of the points in the space are "far" from each other.

This illustrates the fact that local methods will break down in these circumstances (eg, in order to collect enough neighbors for a given point, you need to expand the radius of the neighborhood so far that locality is not preserved).

The bottom line is that high-dimensional spaces can be problematic.

Q: What is the goal of dimensionality reduction?

Q: What is the goal of dimensionality reduction?

We'd like to analyze the data using the most meaningful basis (or **coordinates***) possible.*

Q: What is the goal of dimensionality reduction?

We'd like to analyze the data using the most meaningful basis (or **coordinates***) possible.*

More precisely: given an n **x** d *matrix* A *(encoding* n *observations of a* d*-dimensional random variable), we want to find a* k*-dimensional representation of* A *(*$k < d$*) that (approximately) captures the information in the original data, according to some criterion.*

Q: What is the goal of dimensionality reduction?

- *reduce computational expense*
- *reduce susceptibility to overfitting*
- *reduce noise in the dataset*
- *enhance our intuition*

Q: How is dimensionality reduction performed?

Q: How is dimensionality reduction performed?

A: There are two approaches: feature selection and feature extraction.

Q: How is dimensionality reduction performed?

A: There are two approaches: feature selection and feature extraction.

feature selection – *selecting a subset of features using an external criterion (filter) or the learning algo accuracy itself (wrapper)*

feature extraction – *mapping the features to a lower dimensional space*

Q: How is dimensionality reduction performed?

A: There are two approaches: feature selection and featur

NOTE

We've already seen one example of feature selection for regression: *backward elimination*.

feature selection – *selecting a subset of features using an external criterion (filter) or the learning algo accuracy itself (wrapper)*

feature extraction – *mapping the features to a lower dimensional space*

EXAMPLE: FEATURE SELECTION


```
>>> for i in range(4):
...     print 'P-value for', features[i], ':', feature_selection.f_regression(gas[features].values,
 gas['consumption'].values)[1][i]
...
P-value for tax : 0.00128489067343
P-value for income : 0.0934684297747
P-value for miles : 0.89778460025
P-value for pctlicense : 3.28960494853e-08
```

Feature selection: Removing features with lowest p-values and then refitting model (stepwise regression)

Feature selection is important, but typically when people say dimensionality reduction, they are referring to feature extraction.

Feature selection is important, but typically when people say dimensionality reduction, they are referring to feature extraction.

The goal of feature extraction is to create a new set of coordinates (often in lower dimension) that simplify the representation of the data.

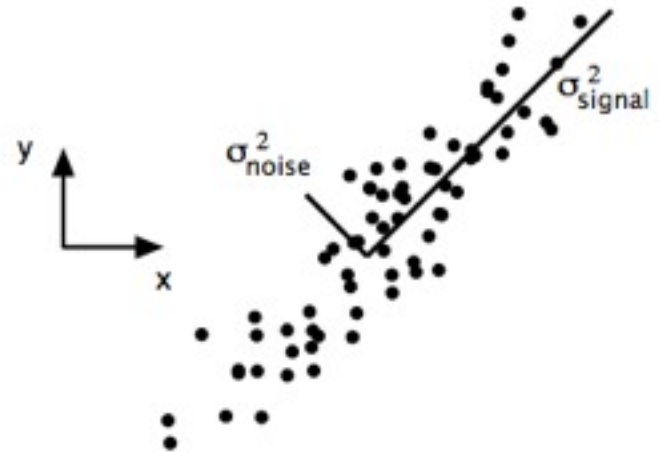


FIG. 2 Simulated data of (x,y) for camera A. The signal and noise variances σ^2_{signal} and σ^2_{noise} are graphically represented by the two lines subtending the cloud of data. Note that the largest direction of variance does not lie along the basis of the recording (x_A, y_A) but rather along the best-fit line.

source: http://www.snl.salk.edu/~shlens/pca.pdf

Q: What are some applications of dimensionality reduction?

Q: What are some applications of dimensionality reduction?

- *topic models (document clustering)*
- *image recognition/computer vision*
- *bioinformatics (microarray analysis)*
- *speech recognition*
- *astronomy (spectral data analysis)*
- *recommender systems*

II. PRINCIPAL COMPONENT ANALYSIS

Principal component analysis is a dimension reduction technique that can be used on a matrix of any dimensions.

Principal component analysis is a dimension reduction technique that can be used on a matrix of any dimensions.

This procedure produces a new basis, each of whose components retain as much variance from the original data as possible.

Principal component analysis is a dimension reduction technique that can be used on a matrix of any dimensions.

This procedure produces a new basis, each of whose components retain as much variance from the original data as possible.

The PCA of a matrix A *boils down to the* **eigenvalue decomposition** *of the* **covariance matrix** *of* A.

The covariance matrix C of a matrix A is always symmetric:

$$C = \begin{bmatrix} \mathrm{E}[(X_1 - \mu_1)(X_1 - \mu_1)] & \mathrm{E}[(X_1 - \mu_1)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_1 - \mu_1)(X_n - \mu_n)] \\ \mathrm{E}[(X_2 - \mu_2)(X_1 - \mu_1)] & \mathrm{E}[(X_2 - \mu_2)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_2 - \mu_2)(X_n - \mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[(X_n - \mu_n)(X_1 - \mu_1)] & \mathrm{E}[(X_n - \mu_n)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_n - \mu_n)(X_n - \mu_n)] \end{bmatrix}.$$

off-diagonal elements C_{ij} give the covariance between X_i, X_j $(i \neq j)$

diagonal elements C_{ii} give the variance of X_i

The eigenvalue decomposition of a symmetric matrix A is given by:

$$A = Q \Lambda Q^T$$

ASIDE: EIGENVALUE DECOMPOSITION

The eigenvalue decomposition of a symmetric matrix A is given by:

$$A = Q \Lambda Q^T$$

The columns of Q are the **eigenvectors** *of A, and the values in Λ are the associated* **eigenvalues** *of A.*

The eigenvalue decomposition of a symmetric matrix A is given by:

$$A = Q \Lambda Q^T$$

The columns of Q are the **eigenvectors** *of A, and the values in Λ are the associated* **eigenvalues** *of A.*

For an eigenvector v of A and its eigenvalue λ, we have the important relation:

$$Av = \lambda v$$

ASIDE: EIGENVALUE DECOMPOSITION

The eigenvalue decomposition of a symmetric matrix A is given by:

$$A = Q \Lambda Q^T$$

The columns of Q are the **eigenvectors** *of A, and the values in Λ are t*

eigenvalues *of A.*

NOTE

This relationship *defines* what it means to be an eigenvector of *A*.

For an eigenvector v of A and its eigenvalue λ, we have the important relation:

$$Av = \lambda v$$

The eigenvectors form a basis of the vector space on which A acts (eg, they are orthogonal).

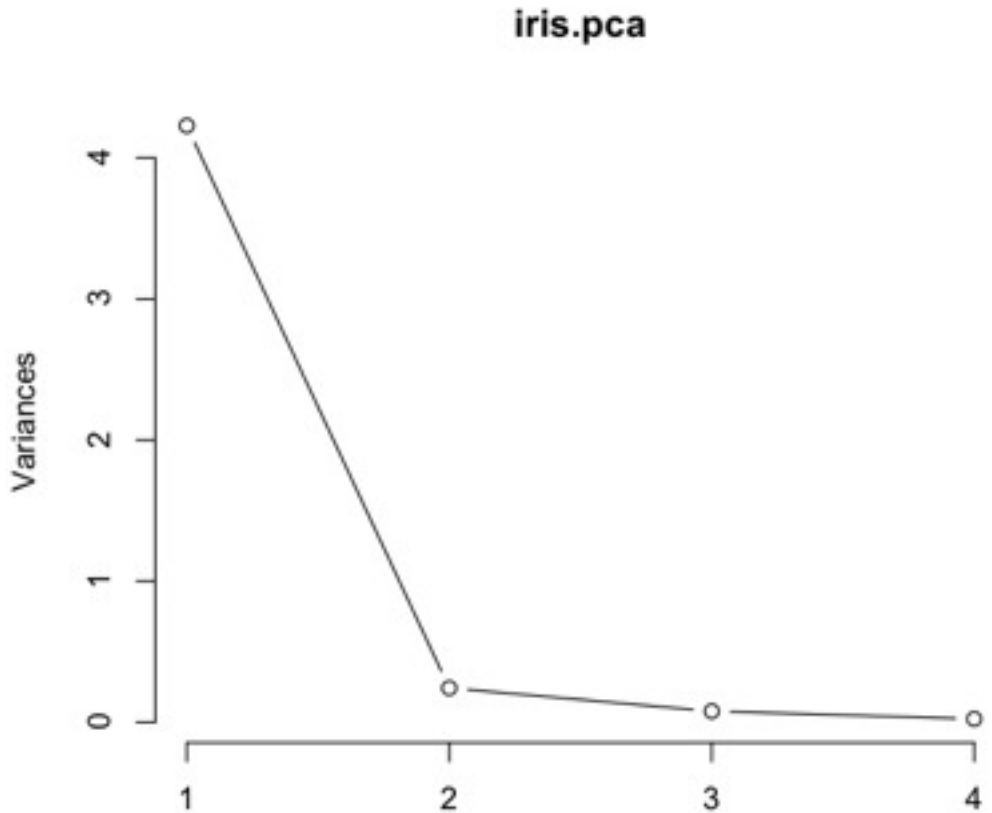
The eigenvectors form a basis of the vector space on which A acts (eg, they are orthogonal).

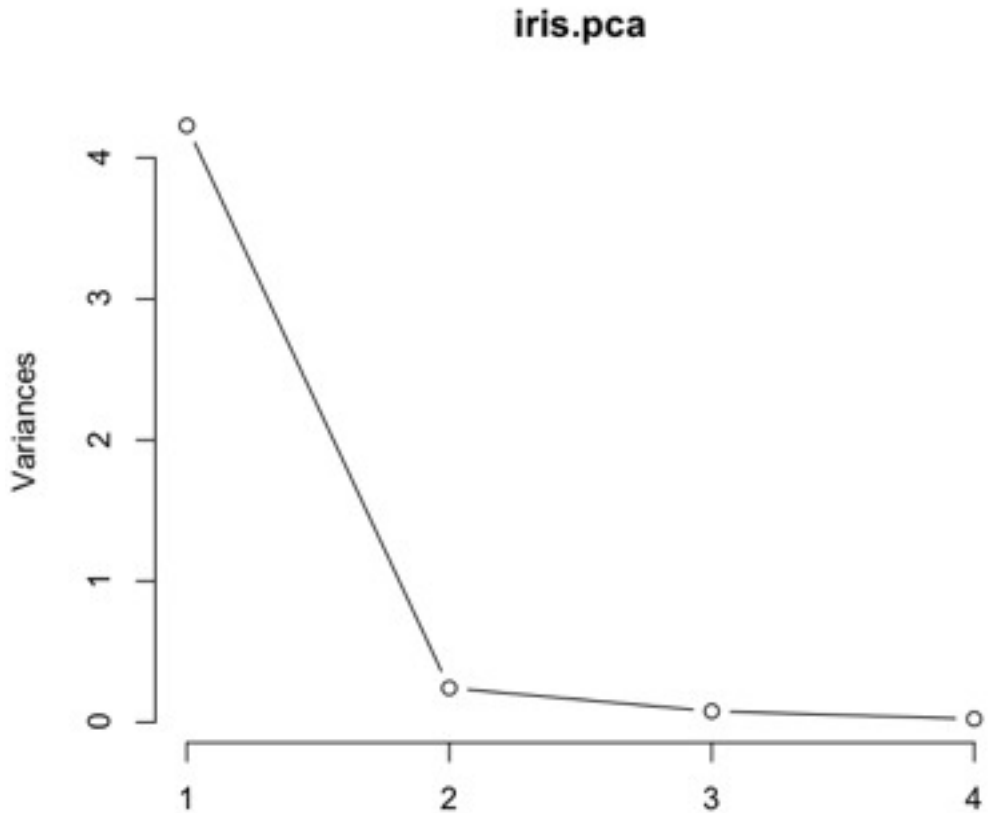
Furthermore the basis elements are ordered by their eigenvalues (from largest to smallest), and these eigenvalues represent the amount of variance explained by each basis element.

The eigenvectors form a basis of the vector space on which A acts (eg, they are orthogonal).

Furthermore the basis elements are ordered by their eigenvalues (from largest to smallest), and these eigenvalues represent the amount of variance explained by each basis element.

This can be visualized in a **scree plot**, *which shows the amount of variance explained by each basis vector.*

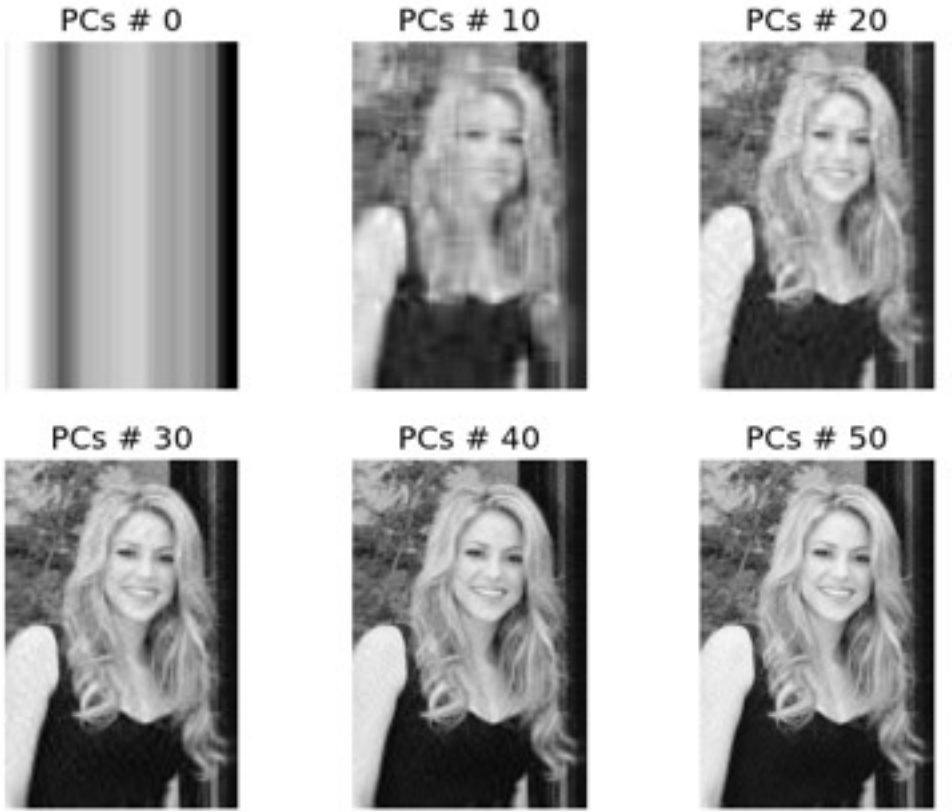

iris.pca
Variances
0
1
2
3
4
1
2
3
4



NOTE

Looking at this plot also gives you an idea of how many principal components to keep.

Apply the *elbow test*: keep only those pc's that appear to the left of the elbow in the graph.



source: http://glowingpython.blogspot.it/2011/07/pca-and-image-compression-with-numpy.html

III. SINGULAR VALUE DECOMPOSITION

Consider a matrix A with n rows and d features.

Consider a matrix A with n rows and d features.

The **singular value decomposition** *of A is given by:*

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

Consider a matrix A with n rows and d features.

The **singular value decomposition** *of A is given by:*

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

(n x d) (n x n) (n x d) (d x d)

Consider a matrix A with n rows and d features.

The **singular value decomposition** *of A is given by:*

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \; \underset{(n \times d)}{\Sigma} \; \underset{(d \times d)}{V^T}$$

st. U, V are **orthogonal** *matrices and Σ is a* **diagonal** *matrix.*

Consider a matrix A with n rows and d features.

The **singular value decomposition** *of A is given by:*

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

$$(n \times d) \qquad (n \times n) \quad (n \times d) \quad (d \times d)$$

st. U, V are **orthogonal** *matrices and Σ is a* **diagonal** *matrix.*

→ $U^T U = U U^T = I_n$, $V^T V = V V^T = I_d$ → $\Sigma_{ij} = 0 \ (i \neq j)$

The **singular value decomposition** *of* A *is given by:*

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \; \underset{(n \times d)}{\Sigma} \; \underset{(d \times d)}{V^T}$$

The columns of U *&* V *are the (left- and right-)* **singular vectors** *of* A.

The **singular value decomposition** *of* A *is given by:*

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \; \underset{(n \times d)}{\Sigma} \; \underset{(d \times d)}{V^T}$$

The columns of U *&* V *are the (left- and right-)* **singular vectors** *of* A*.*

These singular vectors provide **orthonormal bases** *for the spaces* K_n *&* K_d *(columns of* U *&* V*, respectively).*

The **singular value decomposition** *of* A *is given by:*

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \; \underset{(n \times d)}{\Sigma} \; \underset{(d \times d)}{V^T}$$

The nonzero entries of Σ *are the* **singular values** *of* A*. These are real, nonnegative, and rank-ordered (decreasing from left to right).*

The **singular value decomposition** *of* A *is given by:*

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \underset{(n \times d)}{\Sigma} \underset{(d \times d)}{V^T}$$

NOTE

The number of singular values is equal to the *rank* of A.

The rank of a matrix measures its *non-degeneracy.*

The nonzero entries of Σ *are the* **singular values** *of* A*. These are real, nonnegative, and rank-ordered (decreasing from left to right).*

For a general SVD, the columns of U are the eigenvectors of AA^T, and the columns of V are the eigenvectors of A^TA.

Also, the singular values of A are the square roots of the eigenvalues of AA^T and A^TA.

III. KERNEL PCA

Review:

Review:

With support vector machines, we covered three kernels:

Review:

With support vector machines, we covered three kernels:

linear: $K(x,x') = x^T x$

Review:

With support vector machines, we covered three kernels:

linear: $x^T x$

polynomial: $(x^T x' + 1)^d$

Review:

With support vector machines, we covered three kernels:

linear

polynomial: $(x^T x' + 1)^d$

gaussian (rdf): $exp(-\gamma||x - x'||^2)$

Likewise, PCA can also use kernels methods to produce new clarity around the structure of the data.

Likewise, PCA can also use kernels methods to produce new clarity around the structure of the data.

In particular, KPCA is most often used for image de-noising and pattern recognition (or commonly novelty detection).

Likewise, PCA can also use kernels methods to produce new clarity around the structure of the data.

In particular, KPCA is most often used for image de-noising and pattern recognition (or commonly novelty detection).

KPCA is particularly useful for extracting nonlinear features, though like standard PCA, the interpretation is not always straightforward!

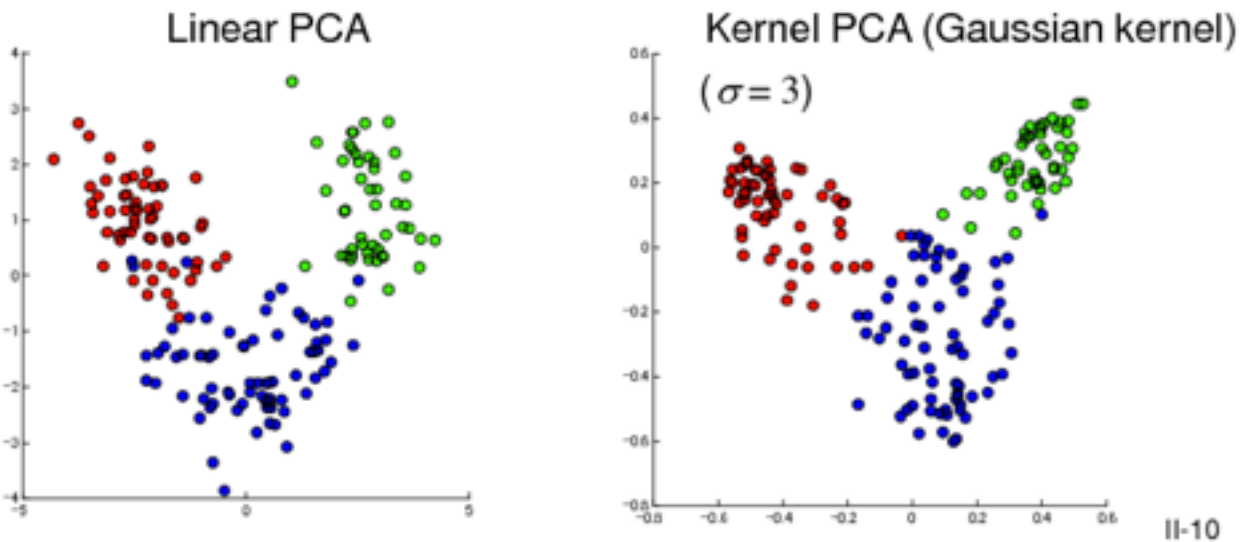
KERNEL PCA - EXAMPLE

- **Wine data** (from UCI repository)

  13 dim. chemical measurements of for three types of wine. 178 data.

  Class labels are NOT used in PCA, but shown in the figures.

  First two principal components:



http://yosinski.com/mlss12/media/slides/MLSS-2012-Fukumizu-Kernel-Methods-for-Statistical-Learning_034.png