Section 19. Principal Components Analysis

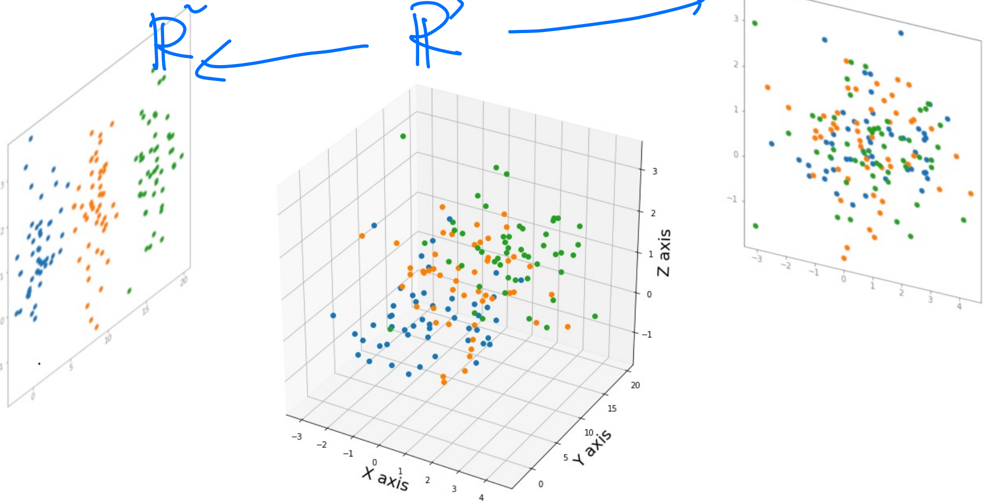
- Dimension Reduction
- Singular value decomposition
- Principal Components
- PCA regression
- PCA classification
- Non-linear PCA

$$\mathbb{R}^2 \quad \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \quad \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Dimensional Reduction

Data: $(\vec{x}^{(i)}, y^{(i)})$ *or* $(\vec{x}^{(i)})$

$$\begin{tikzcd}
\begin{bmatrix} \cancel{x_1} \\ x_2 \\ x_3 \end{bmatrix} & \vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \arrow[l] \arrow[r] & \begin{bmatrix} x_1 \\ x_3 \end{bmatrix} \\
\mathbb{R}^2 & \mathbb{R}^3 \arrow[l] \arrow[r] & \mathbb{R}^2
\end{tikzcd}$$



Dimensional Reduction is the process of combining **high** dimensional information into **low** dimensional representations. Low dimensional representations can be computationally easier to work with, while hopefully throwing away only extraneous information.

Low dimensional projections are meaningful. In fact, all of data visualization can be said to be the result careful low dimensional projections. One goal is to discover the small set of variables that controls the larger phenomena.

➢ Feature engineering

Constructed features play an important role in terms of subset selection. Using our naive subset selection for a linear model, we would quickly throw away some features. However, when we pick the right **combination of the features** suddenly we're given one our best predictors.

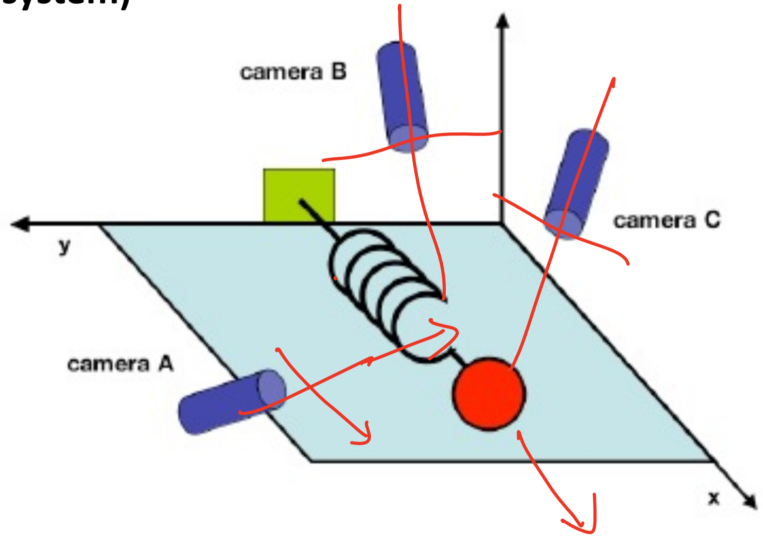
The process of **constructing meaningful features** is called **feature engineering**. Feature engineering is useful both in terms of constructing better models, and communicating results in a human readable format. One of the project of statistics, and by extension machine learning, is to construct a minimal set of mean rich features that explain all of the **variance** in the data.

To this end, feature engineering often uses the techniques of dimensional reduction, and visa versa.

Another motivation example (mass–spring system)

Suppose we do not know what are the real x, y and z axes, so we choose three camera positions A, B, C at some arbitrary angles with respect to the system. The angles between our measurements might not even be 90^o. Now, we record with the cameras for several minutes. Each camera produces a two-dimensional representation of the data. $t=0 \quad t=0.1 \quad t=0.2 \ldots\ldots$

$\vec{x}^{(0)}, \quad \vec{x}^{(1)}, \quad \vec{x}^{(2)}, \ldots\ldots$

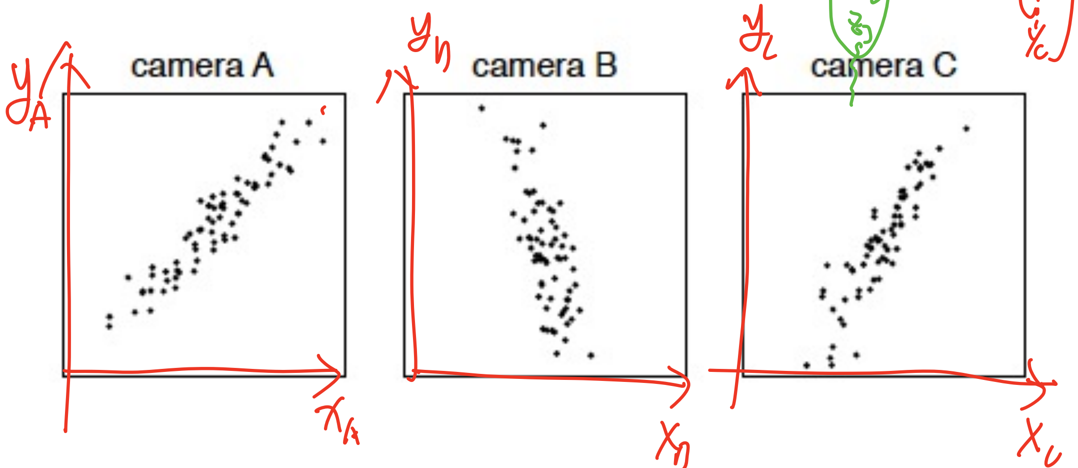


$$\vec{X} = \begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix} \in \mathbb{R}^6$$

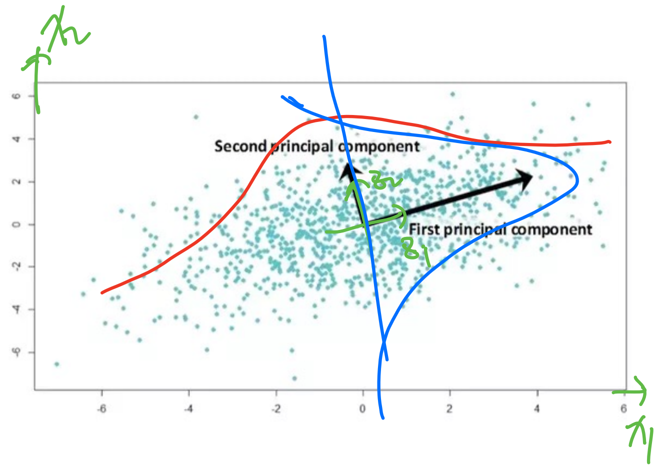
$\vec{z} = (\ldots\ldots) \begin{pmatrix} x_A \\ y_A \\ \vdots \\ y_C \end{pmatrix}$

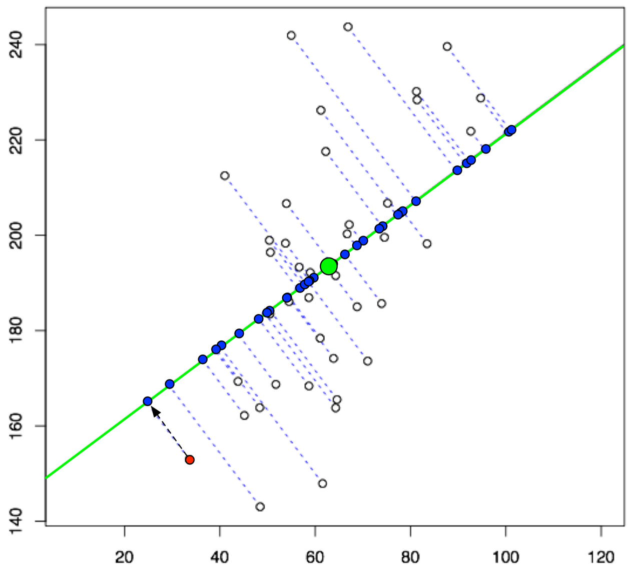
z_2

z_3

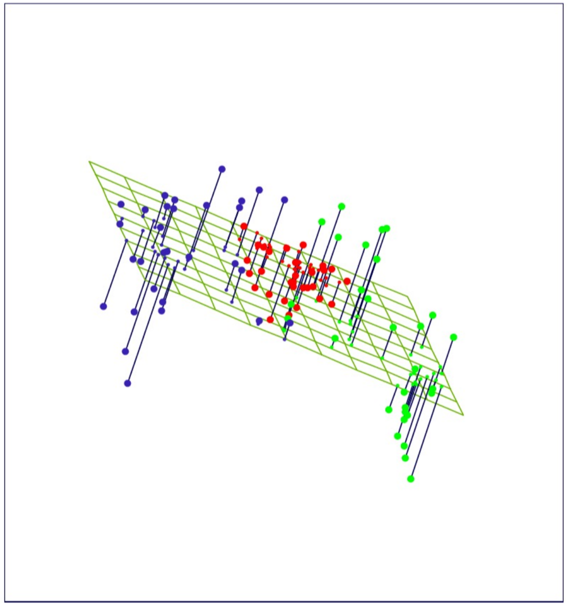


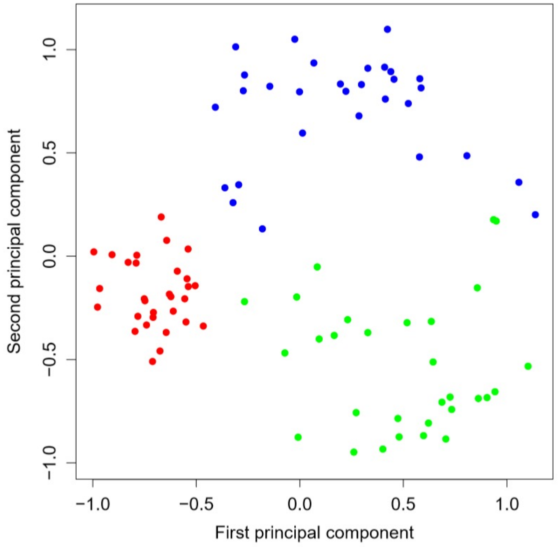
➢ **Principal Components Analysis**



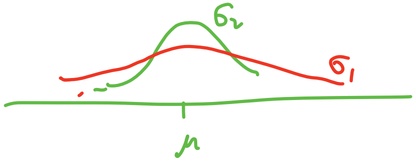


There are two main techniques in dimensional reduction: **projection onto linear subspaces and manifold learning**. In projection onto linear subspaces, we try to discover the **linear combination of features** the provide the clearest coordinate system for the dataset, i.e., the **component factors**.





In an alternative view, factor analysis tries to fit a **linear subspace** to a data set in such a way as to **maximize the projected variance**, that is maximize the amount of data variance captured in orthogonal project to the new space.



$\sigma_2 < \sigma_1$

$\mu = E(X)$

- **Linearly combine** the variables to create the new variables, called **principal components.**

X : r.v.

- The first few explain most of the **variation.**

$Var(X) = E[(X-\mu)^2]$

- Achieve data reduction, without much loss of information.

- It's important to note though that PCA doesn't know anything about the labels y. This can be an **advantage**. Unlabeled data can import the PCA (an unlabeled data is often cheap).

PCA is a greedy algorithm with a beautiful mathematical interpretation.

The idea is to proceed iteratively:

- Find principal component (direction) that accounts for the largest possible variance.
- Project onto the subspace orthogonal to that vector.
- Repeat, collecting the orthogonal vectors into the rows of U until U is a $p \times p$ orthogonal matrix.

$A \in R^{n \times n}$

☆ • A symmetric $\overset{def}{\Longleftrightarrow} A = A^T$

Spectral Decomposition $\Longleftrightarrow A = PDP^{-1}$ and $P^T = P^{-1}$

P is orthogonal

$$D = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix}$$

$E_{\lambda_1} \perp E_{\lambda_2} \cdots \perp E_{\lambda_s}$

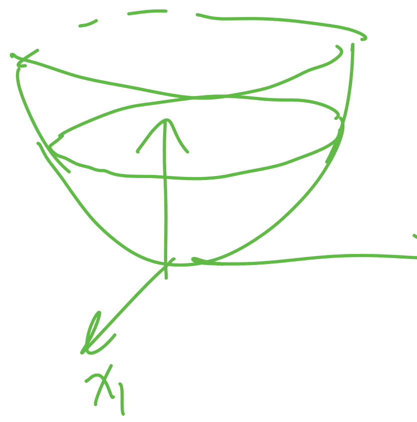
• Ex: $f(x_1, x_2) = x_1^2 + 3x_2^2 + 4x_1x_2 = [x_1 \ x_2]\begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

• quadratic form $P(\vec{x}) = \vec{x}^T A \vec{x}$ $\quad A = A^T$.

• $P(\vec{x})$ or A is positive (semi) definite $\overset{def}{\Longleftrightarrow}$

$P(\vec{x}) = \vec{x}^T A \vec{x} \geq 0$ for any $\vec{x} \neq \vec{0} \in \mathbb{R}^n$

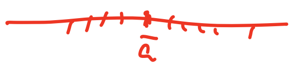
$\Longleftrightarrow$ (all eigenvalues of A) ≥ 0



☆ • Let X be any $m \times n$ matrix, then

• $\underbrace{X^T X}_{n \times n}$ is positive semi-definite.

• If rank $X = n$, then $X^T X$ is positive definite.



$A = a_1, a_2, \cdots, a_n$

➢ Singular Value Decomposition

Data Matrix $X = \begin{bmatrix} \vec{x}^{(1)T} \\ \vdots \\ \vdots \end{bmatrix}_{n \times d}$

Suppose $E(X) = \vec{0}$

• $Var(A) := \frac{1}{n-1} \sum_{i=1}^{n} (a_i - \bar{a})^2$

☆ • Def: Sample Covariance

$Cov(X) := \frac{1}{n-1} X^T X = \begin{bmatrix} Var(X_1) & Cov(X_1, X_2) & \cdots \\ \vdots & & \\ \vdots & & \\ \vdots & & \end{bmatrix}_{d \times d}$

• Ex: $X = \begin{array}{cc} \text{ages} & \text{height} \\ \left[\begin{array}{c} 3 \\ -4 \\ 7 \\ 1 \\ -4 \\ -3 \end{array}\right. & \left.\begin{array}{c} 7 \\ -6 \\ 8 \\ -1 \\ -1 \\ -7 \end{array}\right] \end{array}$

$Cov(X) := \frac{1}{5} X^T X = \begin{bmatrix} 20 & 25 \\ 25 & 40 \end{bmatrix}$

• $\boxed{A = X^T X}$ $d \times d$.

$r = \text{rank } A = \text{rank}(X)$

① A is symmetric $\Longrightarrow$ A has d real eigenvalues

② A is positive-semidefinite $\Longrightarrow$ $\lambda_1 \geq \lambda_2 \geq \lambda_3 \cdots \geq \lambda_r > 0$

$$X^T X = A = VDV^{-1} = \boxed{VDV^T}$$

$$D = \begin{bmatrix} \lambda_1 & & & & & 0 \\ & \lambda_2 & & & & \\ & & \ddots & & & \\ & & & \lambda_r & & \\ & & & & 0 & \\ 0 & & & & & \ddots \; 0 \end{bmatrix}$$

$$= \lambda_1 \vec{v}_1 \vec{v}_1^T + \lambda_2 \vec{v}_2 \vec{v}_2^T + \cdots + \lambda_r \vec{v}_r \vec{v}_r^T$$

$$V = [\vec{v}_1 \cdots \vec{v}_d]$$

$$U = [u_1 \cdots u_r \cdots u_n]$$

Goal 2: $X = U \Sigma V^T$

$$= \sigma_1 \vec{u}_1 \vec{v}_1^T + \cdots + \sigma_r \vec{u}_r \vec{v}_r^T$$

E.g. $A^T A =$

$$\begin{bmatrix} \vec{a}_1^T \\ \vec{a}_2^T \end{bmatrix} [\vec{a}_1 \; \vec{a}_2]$$

$$= \begin{bmatrix} a_1 \cdot a_1 & a_1 \cdot a_2 \\ a_2 \cdot a_1 & a_2 \cdot a_2 \end{bmatrix}$$

$$X^T X = V D V^T \iff V^T X^T X V = D \iff (XV)^T XV = D$$

- $$XV = [X\vec{v}_1 \cdots X\vec{v}_d] \longrightarrow$$

$$\iff \begin{bmatrix} \ddots & \cdots & & \\ \cdots & X\vec{v}_i \cdot X\vec{v}_j & \cdots & \\ \cdots & & & \ddots \end{bmatrix} = \begin{bmatrix} \lambda_1 & & & & & 0 \\ & \lambda_2 & & & & \\ & & \ddots & & & \\ & & & \lambda_r & & \\ & & & & 0 & \\ 0 & & & & & \ddots 0 \end{bmatrix}$$

$$\iff \begin{cases} \|X\vec{v}_i\|^2 = \lambda_i & \text{for } i = 1, 2, \cdots r \\ X\vec{v}_i \cdot X\vec{v}_j = 0 & \text{for } i \neq j \iff \text{all orthogonal} \\ X\vec{v}_i = 0 & \text{for } i > r \end{cases}$$

singular values $\swarrow$

· Denote : $\vec{u}_i := \frac{X\vec{v}_i}{\|X\vec{v}_i\|}$ for $i=1,\cdots,r$. Denote : $\sigma_i = \|X\vec{v}_i\| = \sqrt{\lambda_i}$

$$\underset{n\times d}{XV} = [\underset{d\times d}{X\vec{v}_1} \cdots X\vec{v}_r \; \underset{n\times n}{X\vec{v}_{r+1}} \cdots \underset{n\times d}{X\vec{v}_d}] = [\sigma_1\vec{u}_1 \; \sigma_2\vec{u}_2 \cdots \sigma_r\vec{u}_r \; 0 \; \underline{0 \cdots 0}]$$

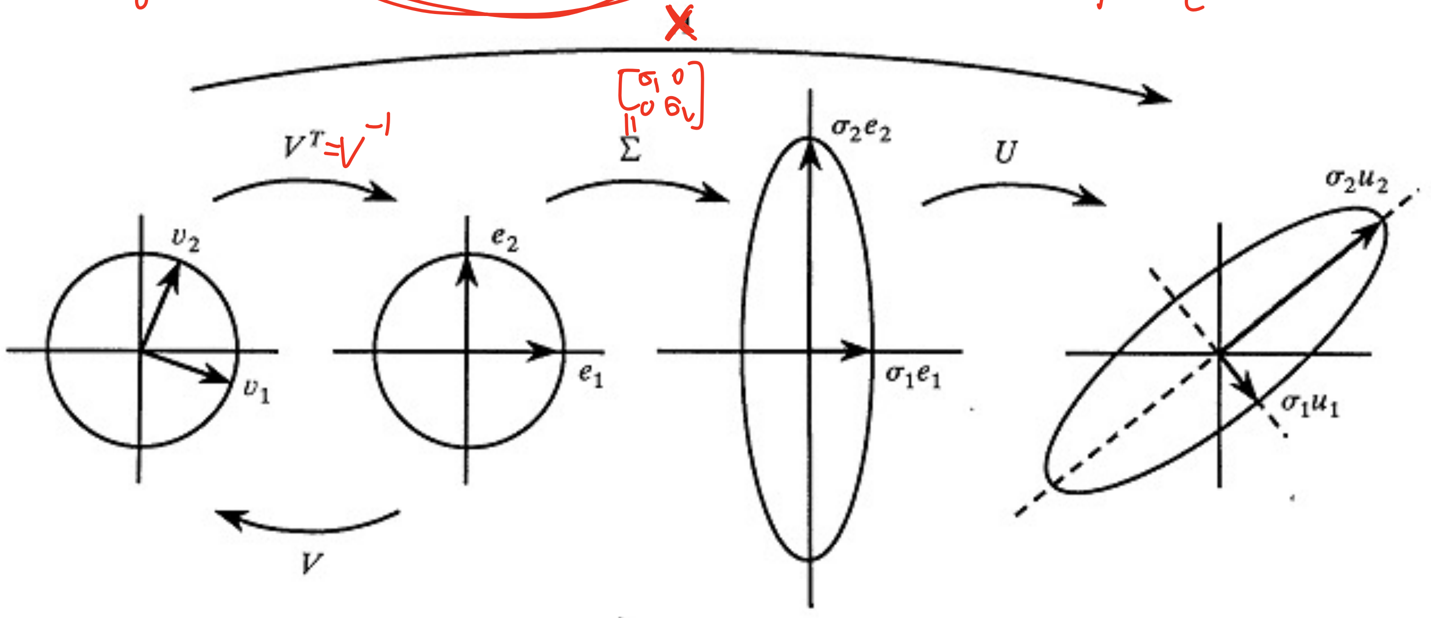
$$= [\underline{\vec{u}_1 \; \vec{u}_2 \cdots \vec{u}_r} \; ? \; ? \cdots ?] \underset{n\times d}{\begin{bmatrix} \sigma_1 & & & & & \\ & \sigma_2 & & & O & \\ & & \ddots & & & \\ & & & \sigma_r & & \\ & O & & & 0 & \\ & & & & & \ddots \; 0 \end{bmatrix}}$$

orthogonal complement

$$\boxed{XV = U\Sigma}$$

orthogonal

$$\boxed{X = U\Sigma V^T}$$



Principal Components

$\sigma_i = \sqrt{\lambda_i}$

Data $X_{n \times d}$

$$X = U \Sigma V^T = \underline{\underline{\sigma_1}} \vec{u_1} \vec{v_1}^T + \dots + \underline{\underline{\sigma_r}} \vec{u_r} \vec{v_r}^T$$

Def: $\vec{v_1} \dots \vec{v_r}$ are called the principal components of X.

first

$\boxed{X^T X} = V D V^T$

Def New Data Matrix

$$\underset{n \times d}{Z} = \underset{n \times d}{X} \ \underset{d \times d}{V}$$

$$\mathrm{cov}(Z) = \frac{1}{n-1} Z^T Z = \frac{1}{n-1} V^T X^T X V$$

$$= \frac{1}{n-1} \cancel{V^T V} D \cancel{V^T V}$$

$$= \frac{1}{n-1} \begin{bmatrix} \lambda_1 & & & & & 0 \\ & \lambda_2 & & & & \\ & & \ddots & & & \\ & & & \lambda_r & & \\ & & & & 0 & \\ 0 & & & & & \ddots \; 0 \end{bmatrix}$$

$$\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_d \end{bmatrix} = \boxed{\vec{z} = V^T \vec{x}} = \begin{bmatrix} \vec{v}_1^T \\ \vec{v}_2^T \\ \vdots \\ \vec{v}_d^T \end{bmatrix} \vec{x} = \begin{bmatrix} \vec{v}_1^T \vec{x} \\ \vdots \\ \vec{v}_d^T \vec{x} \end{bmatrix}$$

- The new features z_i, z_j has correlation 0.

$$\underline{X = VDV^T} \qquad D = \begin{bmatrix} 100 & & & & & & \\ & 90 & & & & & \\ & & 10 & & & & \\ & & & 1 & & & \\ & & & & 0 & & \\ & & & & & \ddots & \\ & & & & & & 0 \end{bmatrix}$$

$$\boxed{\underline{Z = XV}}$$

$$Z^T Z = \frac{1}{n-1} \begin{bmatrix} 100 & & & & & & \\ & 90 & & & & 0 & \\ & & 10 & & & & \\ & & & 1 & & & \\ & 0 & & & 1 & & \\ & & & & & \ddots & \\ & & & & & & 0 \end{bmatrix}$$

$$\tilde{Z}^T \tilde{Z} = \frac{1}{n-1} \begin{bmatrix} 100 & & & & \\ & 90 & & & \\ & & 10 & & \\ & & & 0 & \\ & & & & 0 \; 0 \end{bmatrix}$$

It is often desired to reduce the number of variables, especially when the number of variables in concern are too many. But the reduction must be done without much loss of information.

Principal components provide an ideal way of such reduction. One may retain the first k principal components , which altogether explains of the total variation.

$$\frac{\lambda_1 + \cdots + \lambda_k}{\lambda_1 + \cdots + \lambda_p}$$

➢ **Principal Components Regression**

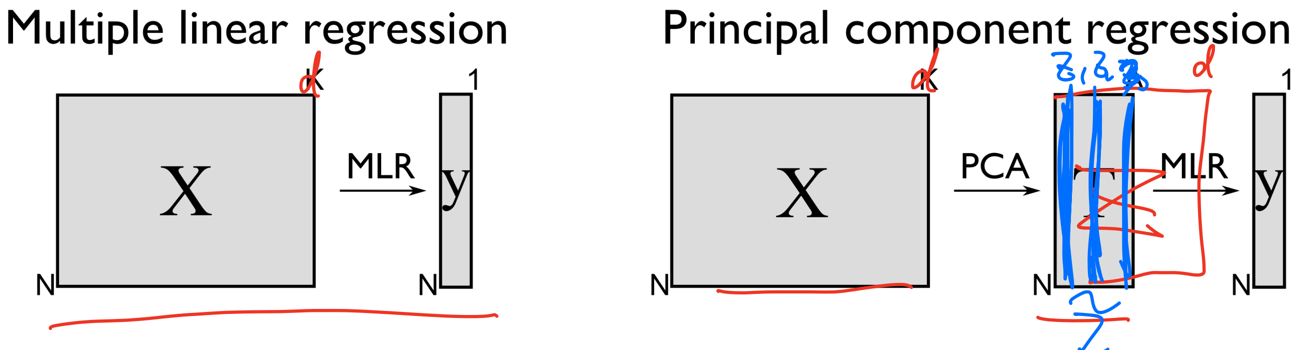
$n \times d$ $n \times d$

Old data X New data $Z = XV =$

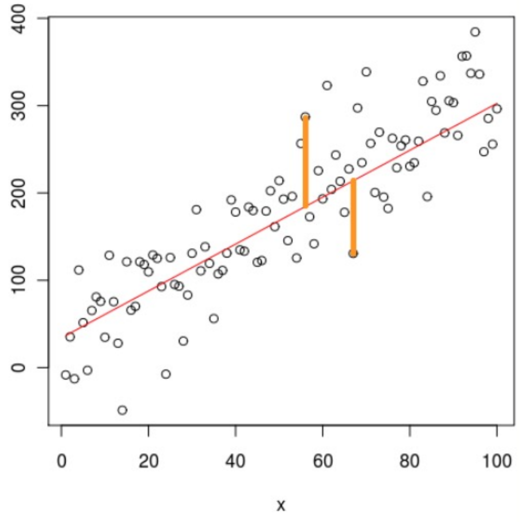
1. perform principal components analysis (PCA) on the original data,
2. perform dimension reduction by selecting the number of principal components (m) using cross-validation or test set error,
3. Conduct regression using the first m dimension reduced principal components.

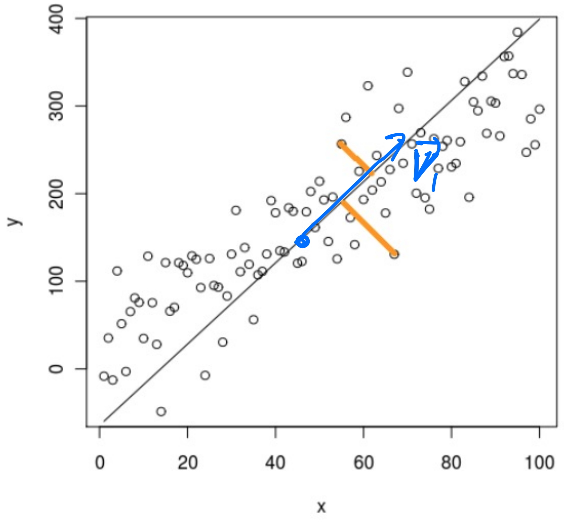
Principal components regression forms the derived input columns $z_m = Xv_m$ and then regresses *on* $z_1, z_2, \cdots, z_m$ for some $m \leq p$.

Principal components regression discards the $p - m$ smallest eigenvalue components.



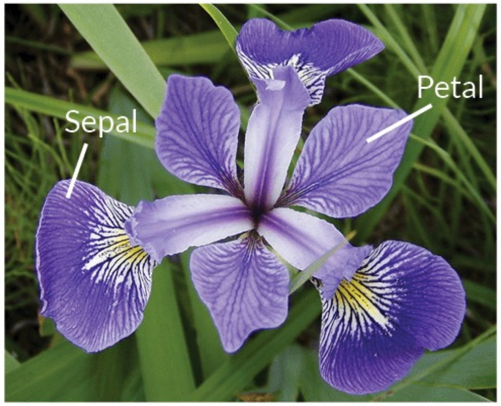
PCA v.s. least square regression





➢ Principal Components for classification

Take as an example projection onto the first two principal components in the Iris data set.

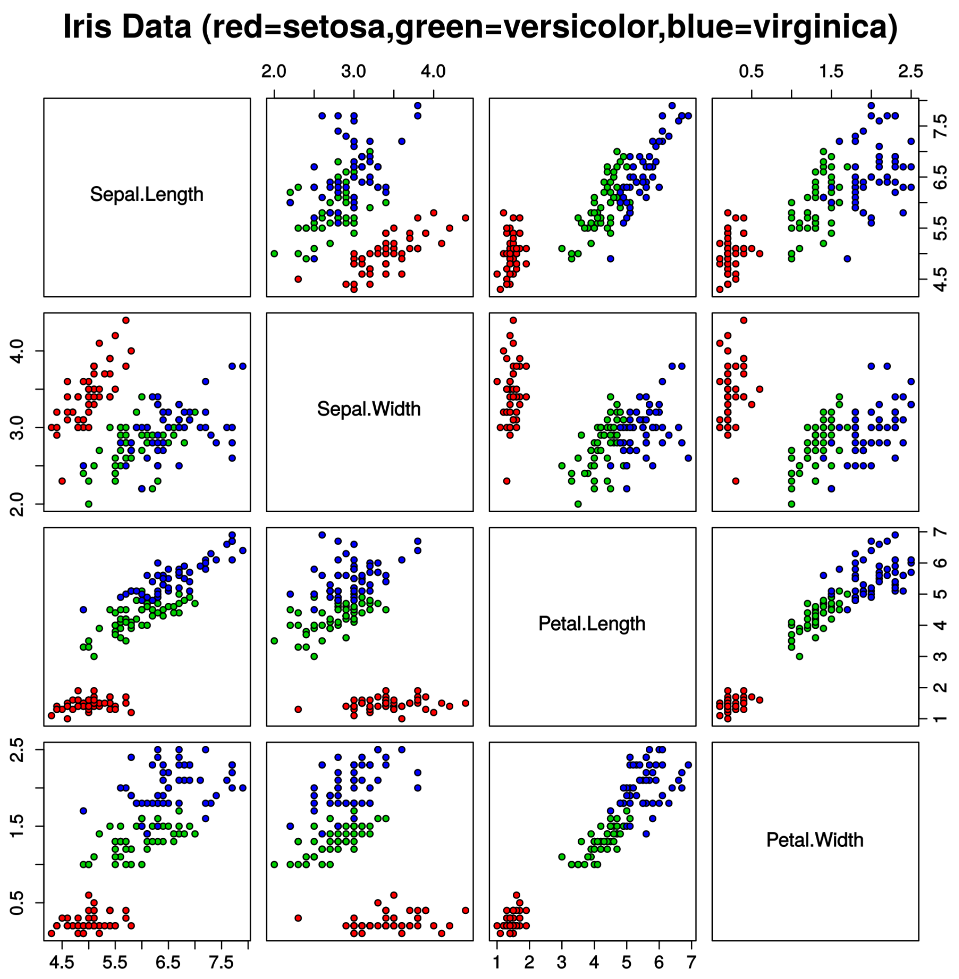


Iris Versicolor

Iris Setosa

Iris Virginica

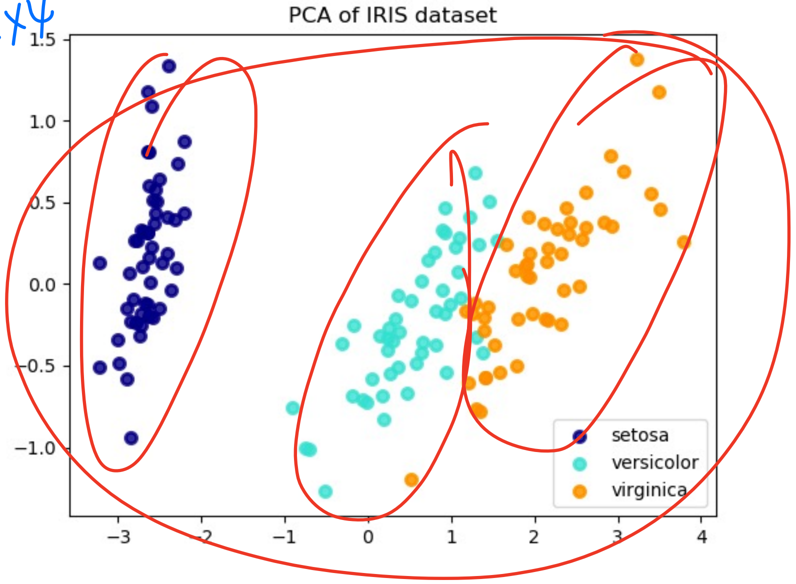
Sepal Length	Sepal Width	Petal Length	Petal Width	Species
5.1	3.5	1.4	0.2	Setosa
7.0	3.2	4.7	1.4	Versicolor

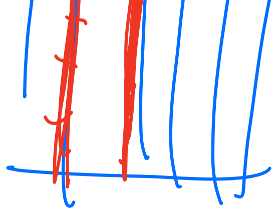


There are four variables, with ~~cor~~ my relations given above.

$Z = XV$

Data $X_{n\times 4}$



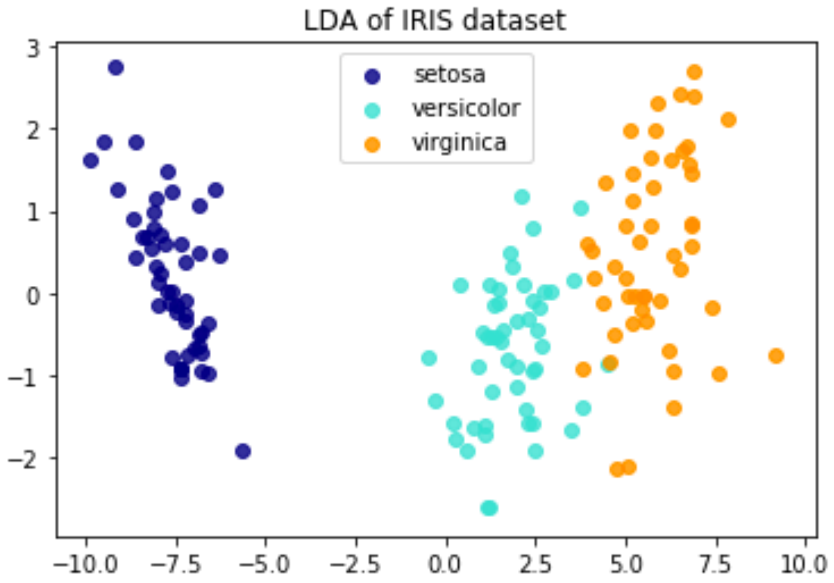


Using PCA to project onto the first two principal components yields a projection onto

$$v_1 = w(1) = (0.36, -0.08, 0.86, 0.36)\ ;\ v_2 = w(2) = (0.66, 0.73, -0.18, -0.07)$$

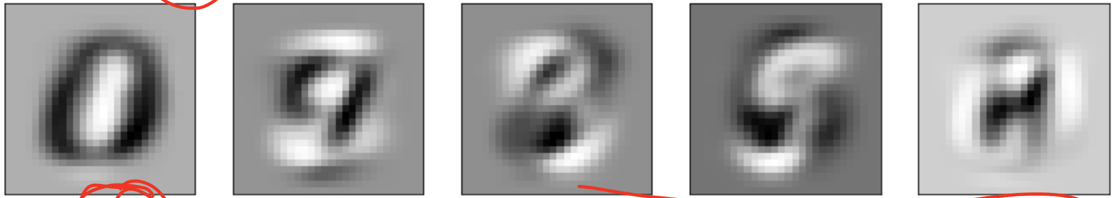
We see the species are separable knowing only their features, and furthermore we have concrete measurement ratios to determine species.

Compare for a moment with the LDA projection. LDA requires knowing the labels and requires significantly more computational time for a similar fit.



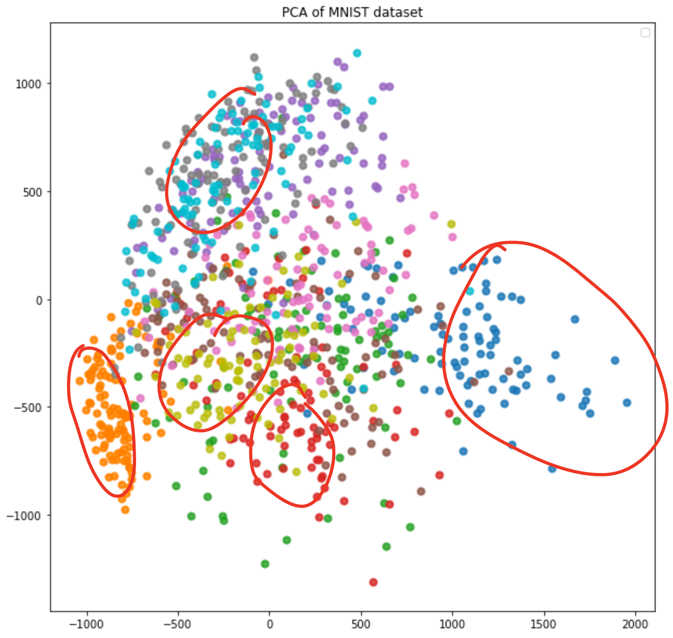
PCA MNIST

$q = \boxed{28\times28}$



$\boxed{28\times28}$

The MNIST data set may provide one of the most startling examples. A simple PCA can be computed quite quickly and yields a wealth of information about the structure of the data set. Projecting onto the first two components shows a lot of structure on the MNIST dataset.



Key assumption: a small number of principal components suffice to explain most of the variability in the data, as well as the relationship with the response.

Choose $z_1, \ldots, z_M$ as the first M principal components.

This assumption may not hold!

For example, performing PCA on the fashion MNIST data set doesn't yield as strong of forms as on the digits MNIST.

$\boxed{\vec{X}}$ $\boxed{\vec{x}}$

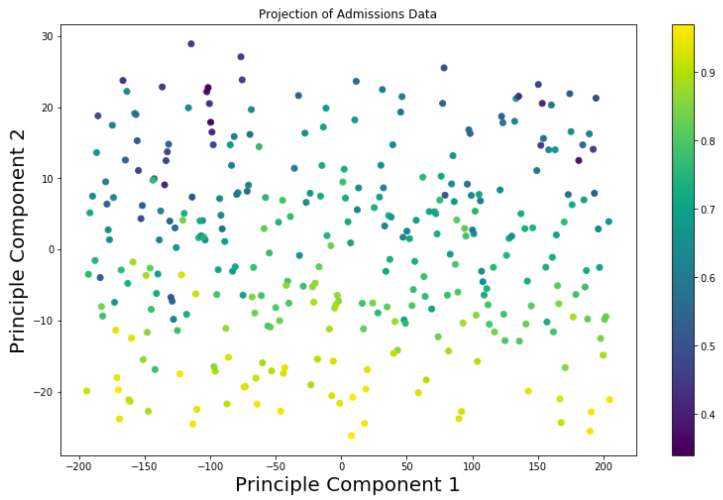
$$\boxed{\text{Data matrix}} \rightarrow \underline{\boxed{X^T X}} = VDV^T$$

$\underline{\text{"PCA"}}$

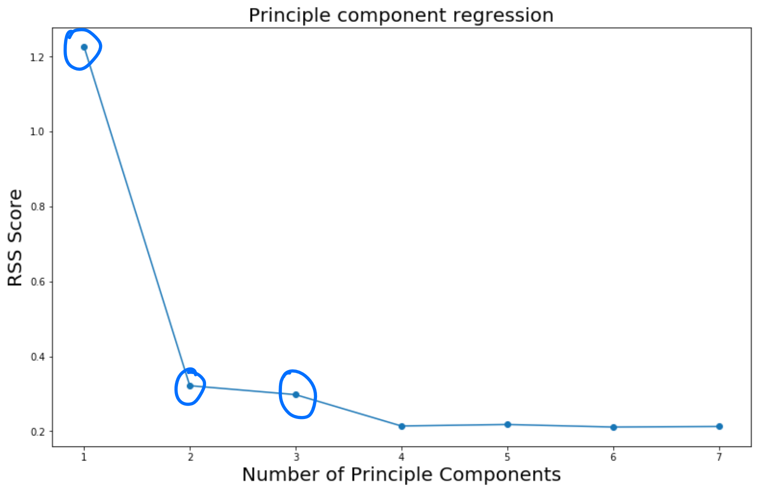
$$\underline{\text{New Data}}\ \underset{n\times d}{Z} = \underset{n\times d}{X}\ \underset{d\times d}{V}$$

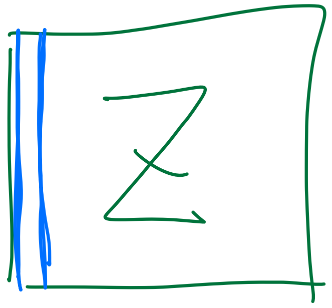
$\underline{\text{"RISE"}}$

Example.



In the context of regression, PCA can perform subset selection for us. In the graph above, we used PCA to project the features of the dataset onto the first two principal components.

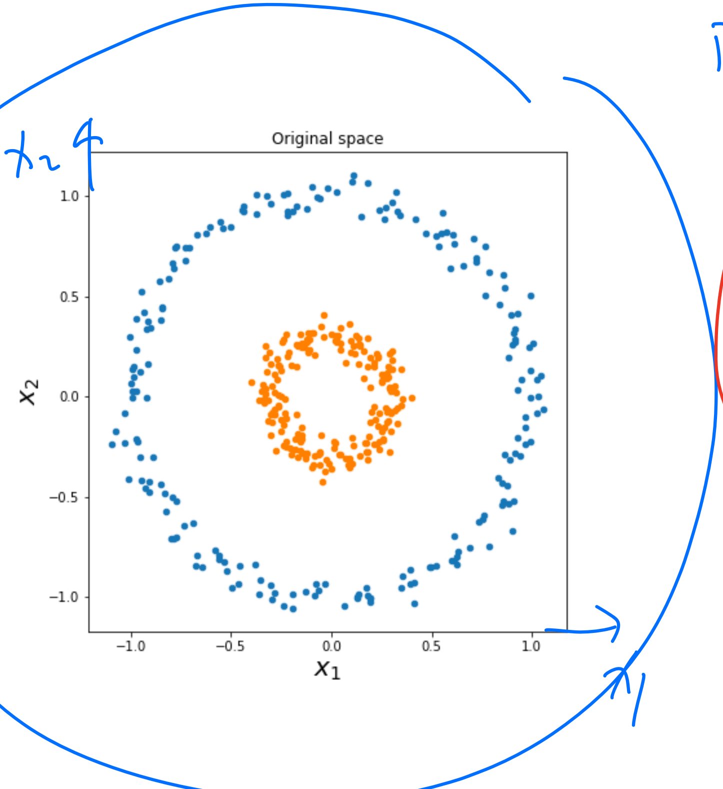


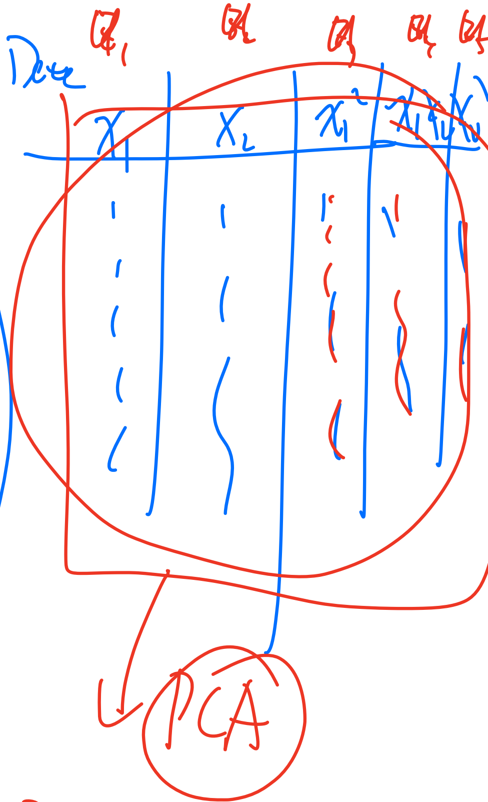


Nonlinear PCA

kernel PCA

Of course, linear methods may be almost useless if the underlying structure isn't linear. For truly unknown structure we must construct smoothing maps using splines, random graphs, or other high complexity techniques. For example, in the following distribution, linear PCA will do nothing to reduce the complexity of the data set.



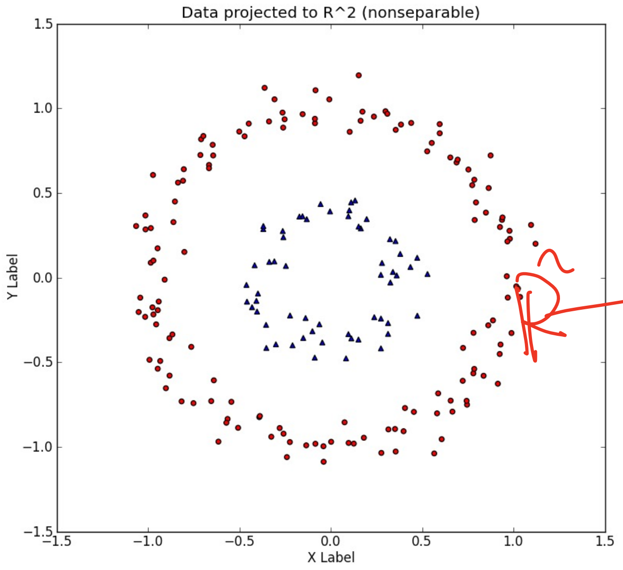


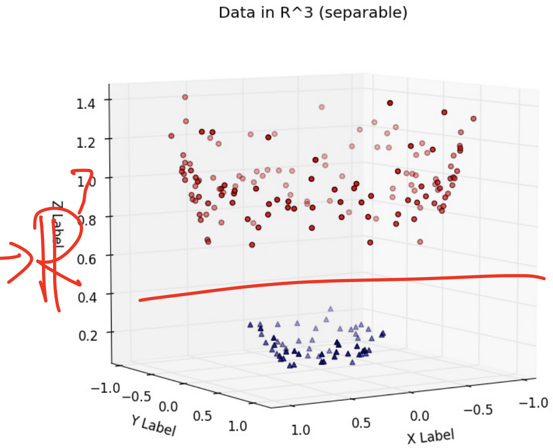
$z_1 =$

$z_2 =$

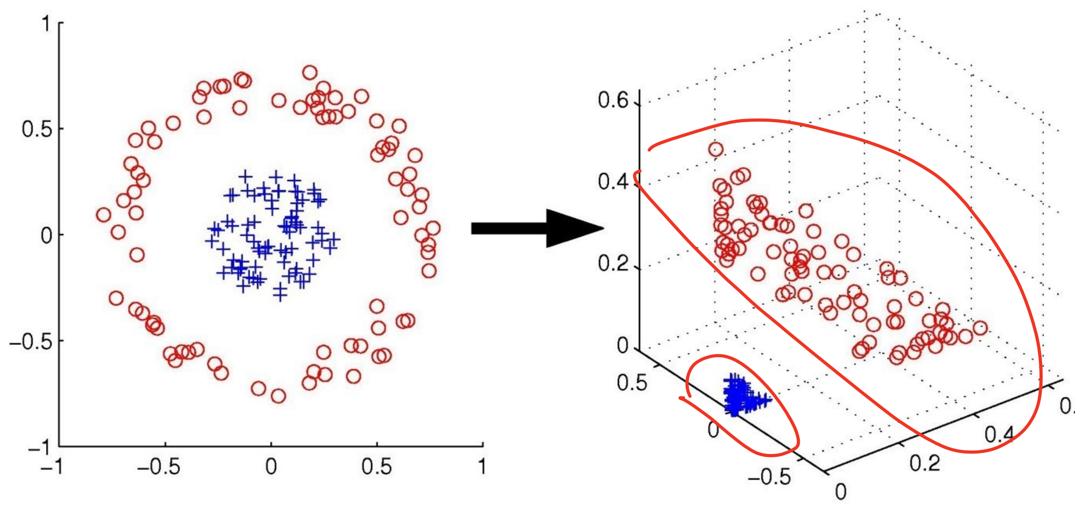
$z_3 =$

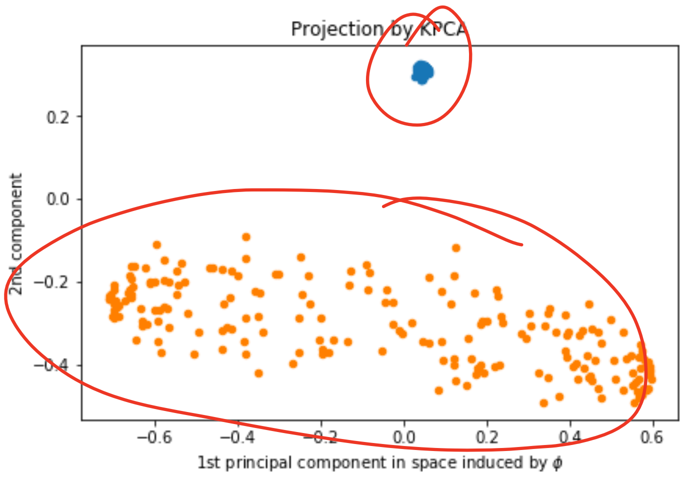
However, whenever we have linear methods there is the hope that we can apply the linear methods to generated features and get a a better fit. For nonlinear PCA, this means that we generate a high dimensional feature space, say $(1, \vec{x}, \vec{x}^2)$, and then project down onto a low dimensional subspace of this feature set, that is a normalized polynomial feature.



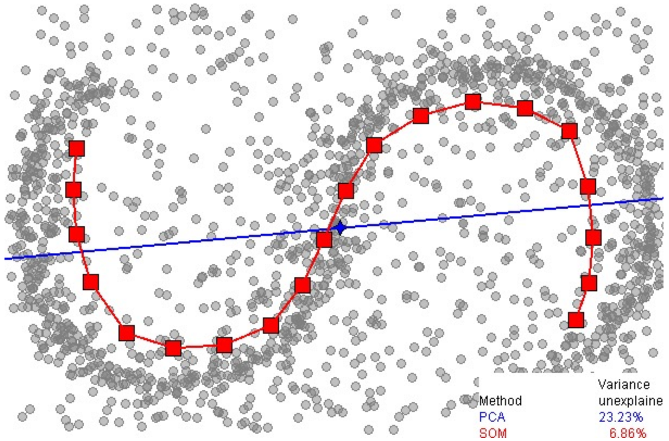


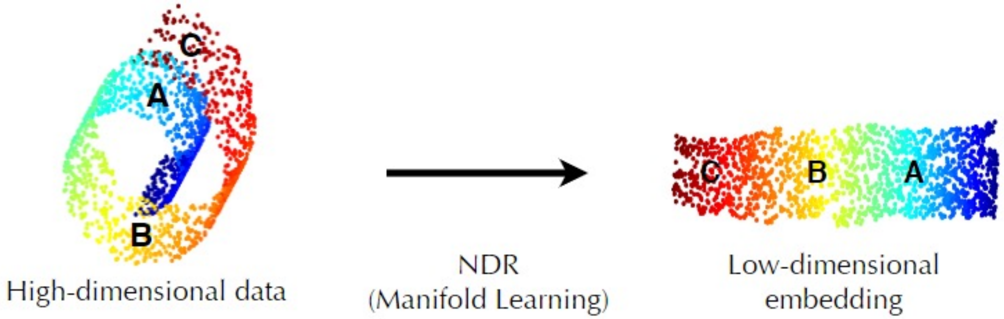
Applying the **radial basis function** (rbf) kernel PCA to the concentric circle data set results in a dramatic clustering.





Manifold learning





In manifold learning, we try to fit a more complicated manifold to the underlying data. Manifold learning is of course much more complicated than simple linear dimensional reduction, but there do exist good algorithms.

Scikit-Learn:

https://scikit-learn.org/stable/modules/manifold.html

https://scikit-learn.org/stable/modules/decomposition.html#pca