Input space

linear PCA ← input space, using D

kernel PCA

kernelization

PCA → kernel PCA
LDA → kernel LDA
Regression → kernel regression

allowed to use $K$

$$K(x; x') = \phi(x)\phi(x')^T$$
Any linear direction is a linear combination of the points in their space.

Find direction $v$:

$$v = \frac{1}{N} \sum_{i=1}^{N} \phi(x_i)$$

New mean will be $\mu_Y = 0$

After this, new mean will be $\mu_Y = 0$

$$\phi(x_i) = \phi(x_i) - \mu_Y$$

Feature space.

Given point $\phi(x_i)$ in feature space.

Compute $k$-nearest neighbors:

$$K(x_i, y_j) = \phi(x_i) \phi(y_j)^T$$

$$\phi(x) = \phi(x)$$

Regression is kernel regression.
\[ \tilde{u}_1 = \sum_{i=1}^{n} c_i \phi(x_i) \]

\[ \tilde{c} = (c_1, c_2, c_3, \ldots, c_n) \]

Coefficient Vector (Unknown)

\[ \sum_{\phi} \tilde{u}_i = \lambda_i \tilde{u}_i \]

\[ \frac{1}{n} \left[ \sum_{i=1}^{n} \phi(x_i) \phi(x_i)^T \right] \left[ \sum_{j=1}^{n} c_j \phi(x_j) \right] = \lambda_i \sum_{i=1}^{n} c_i \phi(x_i) \]

\[ \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} c_j \phi(x_i) \left[ \phi(x_i)^T \phi(x_j) \right] = \lambda_i \sum_{i=1}^{n} c_i \phi(x_i) \]

\[ \frac{1}{n} \sum_{i=1}^{n} \left( \phi(x_i) \sum_{j=1}^{n} c_j k(x_i, x_j) \right) = \lambda_i \sum_{i=1}^{n} c_i \phi(x_i) \]

\[ \sum_{i} \phi(x_i), \sum_{j=1}^{n} c_j k(x_i, x_j) = n \lambda_i \sum_{i} c_i \phi(x_i) \]

LHS = RHS

Pick a point \( \phi(x_k) \)

\[ \phi(x_k)^T [\text{LHS}] = \phi(x_k)^T \left[ \text{RHS} \right] \]
\[
\sum_{i=1}^{n} \phi(x_k)^T \phi(x_i) \sum_{j=1}^{n} c_j k(x_j, x_i) = \lambda \sum_{i=1}^{n} c_i \phi(x_k)^T \phi(x_i)
\]

\[
\left\{ \sum_{i=1}^{n} k(x_k, x_i) \sum_{j=1}^{n} c_j k(x_j, x_i) \right\} = \lambda \sum_{i=1}^{n} c_i k(x_k, x_i)
\]

\(n\) equations, one per choice of \(\phi(x_k)\)

\[
k = 1, 2, \ldots, n
\]

1 matrix equation

\[
K^2 \vec{c} = \lambda \vec{k} \vec{c}
\]

\(K\leftarrow\text{kernel matrix}\)

\(\vec{c}\leftarrow\text{Coefficient vector}\)

\[
\vec{u}_1 = \sum_{i=1}^{n} c_i \phi(x_i)
\]

1st PC in feature space

\[d\]

\[D \rightarrow \Sigma \]

\[d \ldots\]

Simplified

\[
k^{-1} \times k^{-1} = \lambda \vec{k} \vec{c}
\]

\(\text{Linear kernel}\)
\[ k^2 \mathbf{z} = \lambda \mathbf{z} \]

Eigenvalue - eigenvector eq.
\( \mathbf{z} \) in the dominant eigenvector
\( \lambda_1 = \text{max} \lambda_i \) in the largest eigenvalue

\[ k^2 \mathbf{z} = \gamma \mathbf{z} \]

\[ k \mathbf{z} = \gamma \mathbf{z} \]

To solve

\[ k^2 \mathbf{z} = \lambda \mathbf{z} \]

we need to solve

\[ k \mathbf{z} = \lambda \mathbf{z} \]

all valid \( \mathbf{z} \) are common to both 0 and 1

\[ \text{corresponding to non-zero eigenvalue} \]

Kernel PCA

\[ D \leftarrow \text{input} \]
1) Create $K$

2) Center $K = (I - \delta) K (I - \delta)$

3) Solve for $\tilde{c}_i$

   \[ K \tilde{c}_i = \lambda_i \tilde{c}_i \]

4) Simply project all points $\phi(x_k)$ onto $\tilde{u}_1$

   \[ \text{projected value} = \phi(x_k)^T \tilde{u}_1 \]

   \[ = \phi(x_k)^T \left( \sum_{i=1}^{n} c_{ki} \phi(x_i) \right) \]

   \[ = \sum_{i=1}^{n} c_{ki} \phi(x_k)^T \phi(x_i) \]

   \[ a_k = \sum_{i=1}^{n} (c_{ki}) K(x_k, x_i) \]

   Projection of $\phi(x_k)$ along $\tilde{u}_1$
\[ \mathbf{u}_1 = \sum_{i=1}^{n} c_i \phi(x_i) \]

How to ensure that \( \mathbf{u}_1 \) has unit length:

\[ ||\mathbf{u}_1||^2 = \mathbf{u}_1^T \mathbf{u}_1 = 1 \]

\[
\begin{bmatrix}
\sum_{i=1}^{n} c_i \phi(x_i)
\end{bmatrix}
\begin{bmatrix}
\sum_{j=1}^{n} c_{ij} \phi(x_j)
\end{bmatrix}
\]

Yescale \( \mathbf{z}_1 \):

\[
\mathbf{z}_1 = \frac{1}{\sqrt{\lambda_1}} \mathbf{c}_1
\]

\[ D \rightarrow \mathbf{K} \]

Coefficients:

\[ \mathbf{c}_1 \]
new projected data onto $\tilde{u}_1$

$\text{variance} = \frac{\eta_1}{n} \cdot \lambda_1$

$\lambda_1 = \frac{\eta_1}{n}$

variance along $u_1$

\[ k^i \tilde{c}_i = n \lambda^i \tilde{c}_i \]

\[ k^i \tilde{c}_i = n_i \tilde{c}_i \]

$d$-dim

$k$-dim

linear directions

Non-linear directions

Data set

$k$ linear directions compared to $\lambda_1 \lambda_2 \ldots \lambda_d$

kernel PCA

what is the difference

replace PCA

dimension less
**Kernel LDA**

LDA: maximize separation between groups

\[ D \]

Positive class \( D_1 (+) \)

Negative class \( D_2 (-) \)

Projected out of projection

Projected means \( \bar{m}_1, \bar{m}_2 \)

Projected scatter

\[
Bw = \lambda Sw
\]

\( B \) is the between-class scatter matrix

\( S \) is the within-class scatter matrix

\[
B = (\bar{m}_1 - \bar{m}_2)(\bar{m}_1 - \bar{m}_2)^T
\]

\[
S_1 = \sum_{x_i \in D_1} (\bar{x}_i - \bar{m}_1)(\bar{x}_i - \bar{m}_1)^T
\]

\[
S = S_1 + S_2
\]

Outer product
Assume that we have points in feature space
\[ \phi(x_i) \quad i = 1, \ldots, n \]
in feature space
\[ \tilde{\mathbf{w}} = \sum_{i=1}^{n} a_i \phi(x_i) \]
\[ \hat{\mathbf{a}} = (a_1, a_2, \ldots, a_n) \]
\[ \{ \text{Coefficients} \} \]
project all points \( \phi(x_i) \) onto \( \tilde{\mathbf{w}} \)
compute projected means \( m_1, m_2 \)
scatter \( s_1, s_2 \)
\[ \max J = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2} \]
\( m_1 \) = mean of projected points from class 1
\( \tilde{\mathbf{w}} \) is unit norm
\[ \phi(x_i) \text{ onto } \tilde{\mathbf{w}} \]
projected point:
\[ \tilde{\mathbf{w}}^T \phi(x_i) \]
\[ \left[ \sum a_j \phi(x_j) \right]^T \phi(x_i) \]
Kernel LDA

Kernel Trick

\[ \mathbf{M} \ast \mathbf{a} = \lambda_1 \mathbf{N} \ast \mathbf{a} \]

\[ \mathbf{N} \ast \mathbf{M} \ast \mathbf{a} = \lambda_1 \mathbf{N} \ast \mathbf{a} \]

\[ \mathbf{w} = \sum_{i=1}^{n} a_i \phi(x_i) \]

LDA direction

We cannot compute \( \mathbf{w} \) since we do not have \( \phi(x_i) \)’s
but we can project all points onto \( \mathbf{w} \) to create a new dataset.

### Regression

**Variables:**
- \( A_1, A_2, \ldots, A_d \) (independent attribute/variable)
- \( \mathbf{D} \) (dataset)
- \( Y \) (response variable)
- \( n \) (number of data points)

**Regression Task:**
- Predict \( Y \) as some function of \( A_1, A_2, \ldots, A_d \)

\[
Y = f(A_1, A_2, \ldots, A_d)
\]

**Linear Assumption:**
- **Linear Regression**
  \[
  Y = \mathbf{w}^T \mathbf{A} + \epsilon
  \]

- **Error Term:** \( \epsilon \)
Linear assumption

GAM: generalized additive models

\[
\hat{y} = \omega_1 x_1 + \omega_2 x_2 + \ldots + \omega_d x_d + \omega_{d+1} x_1^2 + \omega_{d+2} x_2^2 + \ldots + \varepsilon
\]

Linear regression

\[
\hat{y} = \omega_1 x_1 + \omega_2 x_2 + \ldots + \omega_d x_d
\]

Predicted response

\[\hat{y}_i \leftarrow \text{true response} \]
\[\hat{y}_i \leftarrow \omega_1 x_1 + \omega_2 x_2 + \ldots + \omega_d x_d + \omega_{d+1} x_1^2 + \omega_{d+2} x_2^2 + \ldots + \text{bias}
\]

Residual error:

\[y_i - \hat{y}_i\]

Residual sum of squared errors (SSE):

\[\text{SSE} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2\]

SSE objective:

\[\min SSE\]
Regression test: find the line that minimizes \( \text{SSE} \) over \( y = \omega_1 x + \omega_0 \)

\[ \hat{y}_i = \omega_1 x_i + \omega_0 \]

\[ \hat{y} = mx + b = \omega_1 x + \omega_0 \]

\[ \omega = (\omega_0, \omega_1) \quad \text{unknown} \]

Objective:
\[ \min \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \]

\[ \min_{\omega_0, \omega_1} J = \sum_{i=1}^{n} \left( y_i - \omega_1 x_i - \omega_0 \right)^2 \]