Homework 3

Yunian Pan

October 27, 2018

1 Problem 1: Kernel

- $k(u,v) = \alpha k_1(u,v) + \beta k_2(u,v),\ for\ \alpha, \beta \geq 0$

  Through building vector blocks:

$$\begin{aligned} k(u,v) &= \alpha\phi_1(u)\cdot\phi_1(v) + \beta\phi_2(u)\cdot\phi_2(v) \\ &= (\sqrt{\alpha}\phi_1(u), \sqrt{\beta}\phi_2(u))\cdot(\sqrt{\alpha}\phi_1(v), \sqrt{\beta}\phi_2(v)) \\ &= \phi(u)\cdot\phi(v) \end{aligned}$$

  Which holds symmetric and positive semi-definite property.

- $k(u,v) = k_1(u,v)k_2(u,v)$

$$\begin{aligned} k(u,v) &= \phi_1(u)\cdot\phi_1(v) \times \phi_2(u)\cdot\phi_2(v) \\ &= (\sum_i f_i(u)f_j(v))(\sum_j g_j(u)g_j(v)) \\ &= \sum_{ij} f_i(u)g_j(u)f_i(v)g_j(v) \\ &= \sum_m \Phi_m(u)\Phi_m(v) \qquad (\Phi_m(x) = f_i(x)g_j(x),\ m = ij) \\ &= \phi(u)\cdot\phi(v) \end{aligned}$$

  Which is symmetric for it keeps the same form when u and v get exchanged.

  Consider the positive semi-definiteness, since $k_1(u,v)$ and $k_2(u,v)$ are both positive semi-definite, the Gram matrix K whose entries are the product of $K_{1,ij}$ and $K_{2,ij}$ is also positive semi-definite. (K_1 and K_2 are Gram matrix of k_1 and k_2 inner product for all $u, v \in \chi$.)

- $k(u,v) = k_1(f(u), f(v)),\ where\ f : \chi \to \chi$

$$\begin{aligned} k(u,v) &= \phi_1(f(u)) \cdot \phi_1(f(v)) \qquad\qquad define\ \phi(x) = \phi_1(f(x)) \\ &= \phi(u) \cdot \phi(v) \end{aligned}$$

Which inherits symmetric and positive semi-definite property from k_1.

- $k(u,v) = g(u)g(v),\ where\ g : \chi \to \mathbb{R}$

Obviously here $\phi(u) = g(u)$, $k(u,v)$ is a valid symmetric kernel for $k(u,v) =< \phi(u), \phi(v) >$.

All we need to do is check the positive semi-definiteness of Gram matrix $K_{ij} = g(x_i)g(x_j)$ for $\{x_i, x_j \in \chi\}$, $K = GG^{\mathsf{T}}$. we can find G and any other $n-1$ vectors which are in the orthogonal complement of G, with eigenvalues $\|G\|^2\ and\ 0(n-1\ multiplicity)$. So K is positive semi-definite.

- $k(u,v) = f(k_1(u,v)),\ where\ f\ is\ a\ polynomial\ with\ positive\ coefficients$

$$k(u,v) = \sum_i c_i(\phi_1(u)\phi_1(v))^i \quad c_i \geq 0$$

$apply\ the\ second\ result\ repeatedly:$

$$k(u,v) = \sum_i c_i k_i(u,v) \quad c_i \geq 0$$

$apply\ the\ first\ result\ repeatedly:$
$k(u,v)\ is\ a\ valid\ kernel.$

- $k(u,v) = \exp(k_1(u,v))$

$$\begin{aligned} k(u,v) &= \lim_{n\to\infty} \sum_{i=0}^{n} \frac{k_1(u,v)^i}{i!} \\ &= f(k_1(u,v))\ where\ f\ is\ a\ polynomial\ with\ positive\ coefficients \end{aligned}$$

Therefore $k(u,v)$ is a valid kernel.

- $k(u,v) = \exp(\frac{-\|u-v\|^2}{\sigma^2})$

$$\begin{aligned} k(u,v) &= \exp(-\frac{\|u\|^2}{\sigma^2}) \times \exp(\frac{2 \times u \cdot v}{\sigma^2}) \times \exp(-\frac{\|v\|^2}{\sigma^2}) \\ &= g(u)g(v)\exp(\phi(u) \cdot \phi(v)) \end{aligned}$$

Where $g(x) = \exp(-\frac{\|x\|^2}{\sigma^2})$ and $\phi(x) = \frac{\sqrt{2}x}{\sigma}$, thus apply the conclusion that $k(u,v) = k_1(u,v)k_2(u,v)$, $k(u,v) = g(u)g(v),\ where\ g : \chi \to \mathbb{R}$ and $k(u,v) = \exp(k_1(u,v))$ are all valid kernels, $k(u,v) = \exp(\frac{-\|u-v\|^2}{\sigma^2})$ is also a valid kernel.

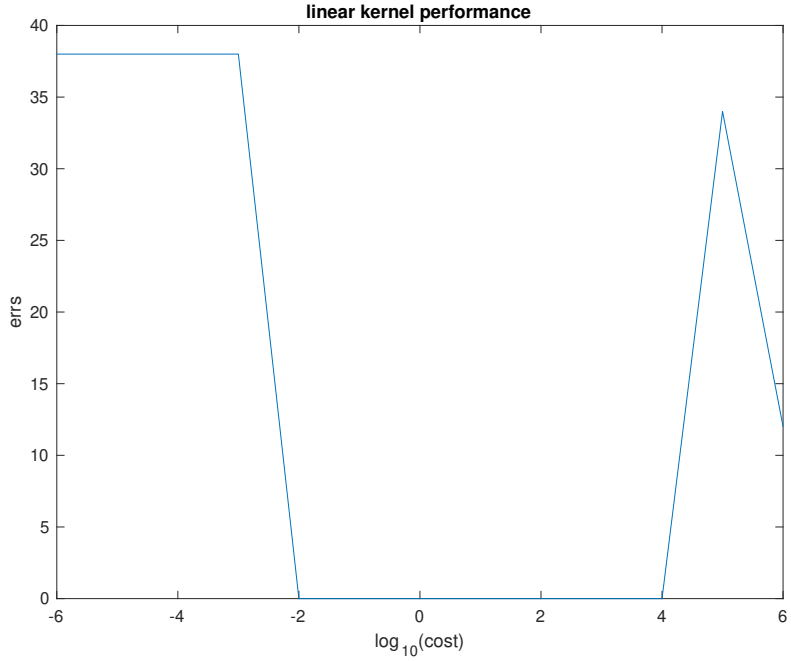


Figure 2.1: linear performance signum

2 Problem 2: SVMs

For linear kernel, we only need to adjust the cost to see the performance of different SVMs, there's no error when cost ranges from 1 to 100 , so I extended the cost value C to $10^{-6} \rightarrow 10^{6}$. As for polynomial kernel and rbf kernel, I used the same cost range with polynomial order p ranging from 1 to 13 and variance σ ranging from 10^{-6} to 10^{6}.

Using the signum function, the performances are shown as in 2.1, 2.2, 2.3.

Using the soft-margin, the performances are shown as in 2.4, 2.5, 2.6.

2.1 shows that when output $f(x)$ is as

$$f(x) = sign(\sum_{t} \alpha_t y_t < x_t, x >)$$

The SVMs perform well when $\log_{10} C \in (-2, 4)$ as the error becomes 0 within that interval.

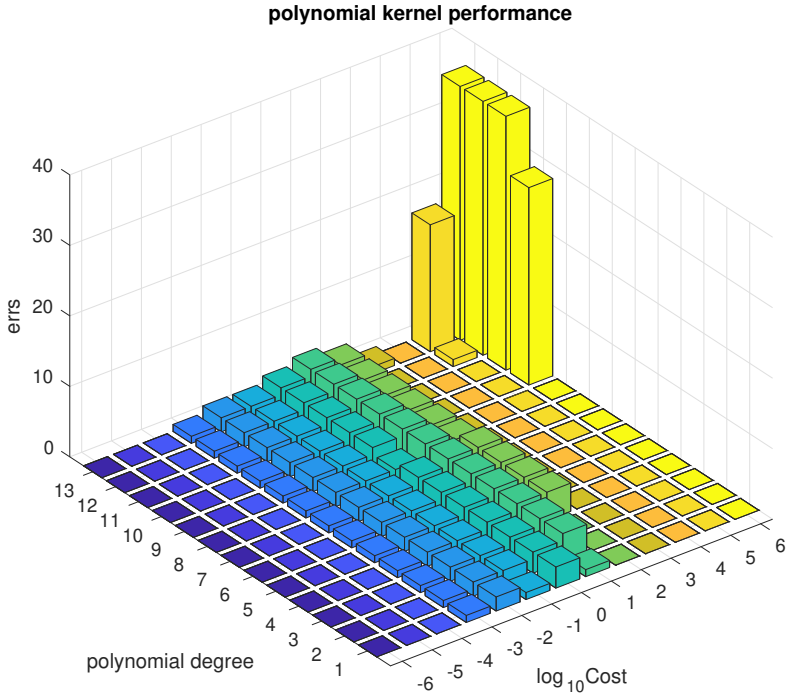


Figure 2.2: polynomial performance signum

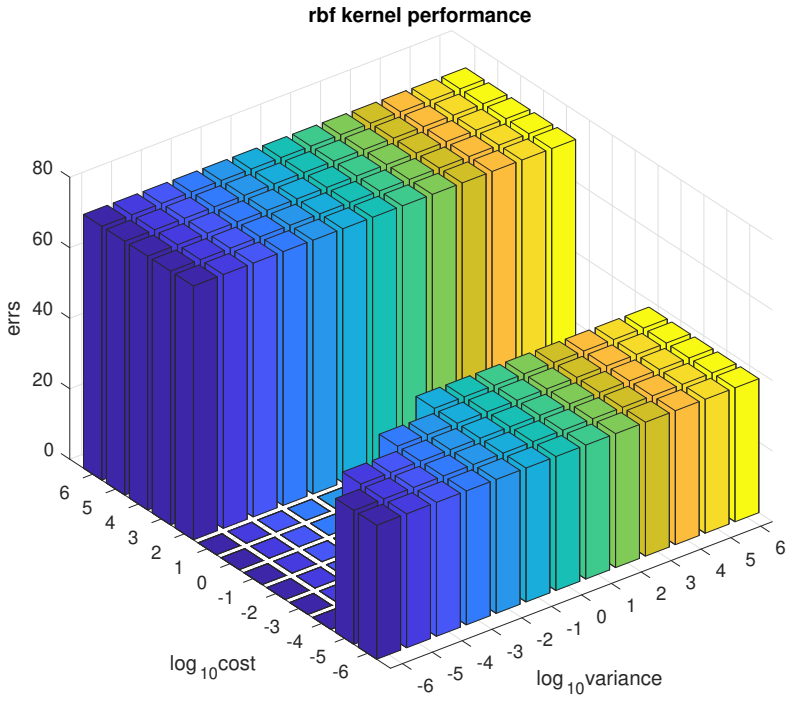


Figure 2.3: rbf performance signum

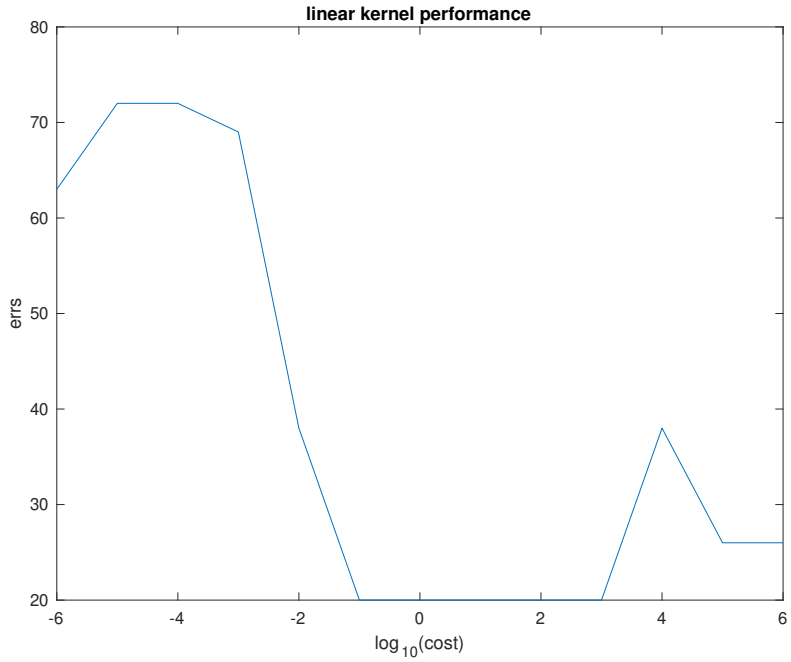


Figure 2.4: linear performance softmargin

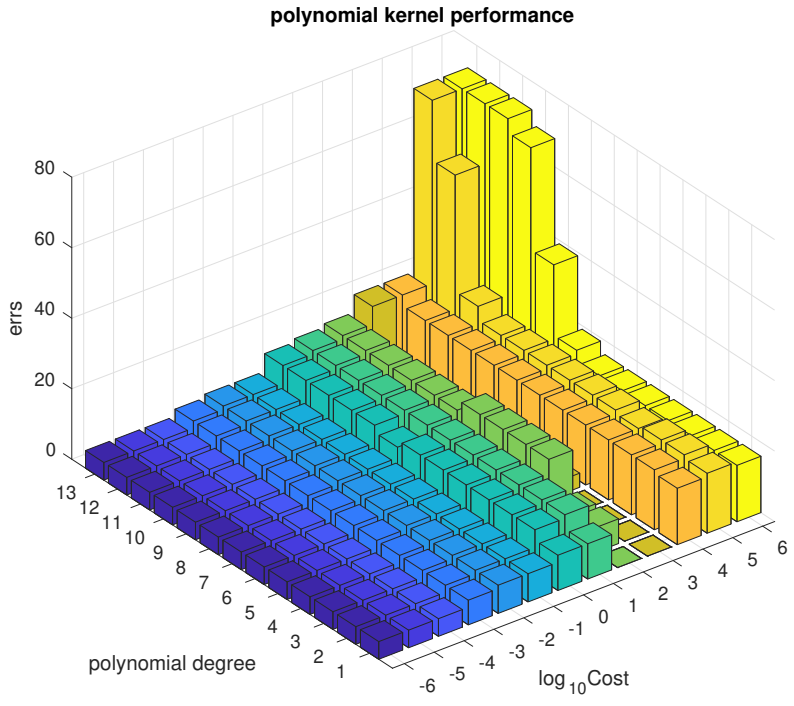


Figure 2.5: polynomial performance softmargin

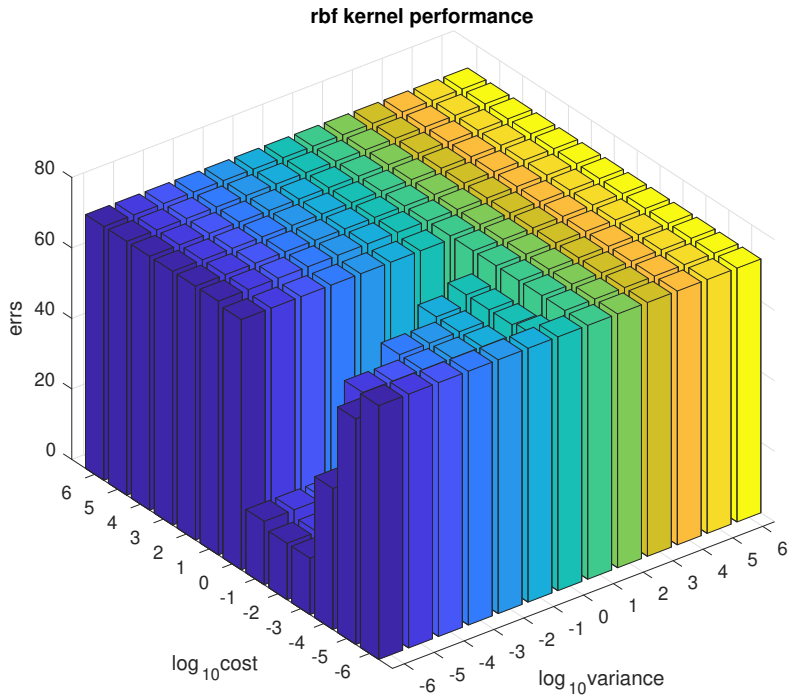


Figure 2.6: rbf performance softmargin

Nevertheless there's some slight difference in 2.4 as the $f(x)$ becomes

$$f(x) = \begin{cases} +1 & \sum_t \alpha_t y_t < x_t, x > -1 > 0 \\ -1 & \sum_t \alpha_t y_t < x_t, x > +1 < 0 \\ \sum_t \alpha_t y_t < x_t, x > & else \end{cases}$$

In 2.5

$$f(x) = \begin{cases} +1 & \sum_t \alpha_t y_t (< x_t, x > +1)^n + b_0 - 1 > 0 \\ -1 & \sum_t \alpha_t y_t (< x_t, x > +1)^n + b_0 + 1 < 0 \\ \sum_t \alpha_t y_t (< x_t, x > +1)^n + b_0 & else \end{cases}$$

In 2.6

$$f(x) = \begin{cases} +1 & \sum_t \alpha_t y_t \exp(\frac{-\|x_t - x\|^2}{2\sigma^2}) + b_0 - 1 > 0 \\ -1 & \sum_t \alpha_t y_t \exp(\frac{-\|x_t - x\|^2}{2\sigma^2}) + b_0 + 1 < 0 \\ \sum_t \alpha_t y_t \exp(\frac{-\|x_t - x\|^2}{2\sigma^2}) + b_0 & else \end{cases}$$

Since the classifier is more strict, there are generally greater recognition errors in 2.5 and 2.6. It can be denoted in both 2.5 and 2.2 that the setting where $p \leq 5$ *and* $10^2 \leq C \leq 10^4$ gives the best polynomial performance. When it comes to rbf kernel, the C should be ranging from 10^{-4} to 10^1 while σ seems to have less effect on the performance according to 2.3.

Conclusion:

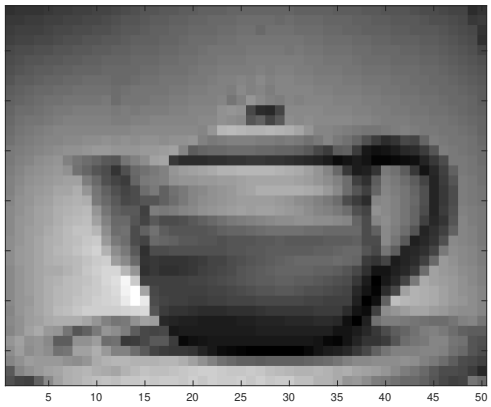
1. linear $C \in (10^{-2}, 10^4)$.
2. polynomial $p \leq 5$ *and* $10^2 \leq C \leq 10^4$.
3. rbf $C \in (10^{-4}, 10^1)$ *and* $\sigma < 10^{-3}$.

3 Problem 3: PCA

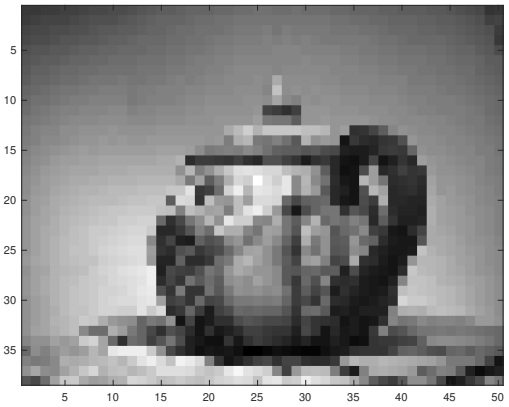
Reconstruct the data using PCA with least squares error using only the mean and a linear combination of the top 3 eigenvectors. Show 10 different images before and after reconstruction.

Discussion: PCA reserves the main information of the data, which contains the mean value and featured terms, the mean value can be intuitively viewed as a spinning teapot which has no spout and handle, and the featured terms can be viewed as the information (pixels and positions) of its handle and spout. By constructing covariance matrix we obtain the self-derivate and the co-derivate on every dimension of the datasets, and for every data point we pick the most effective 3 directions by Eigen-decomposition of the covariance matrix to reconstruct it.

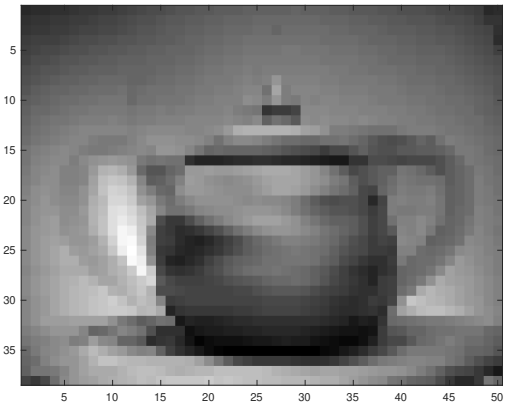
(a) 1



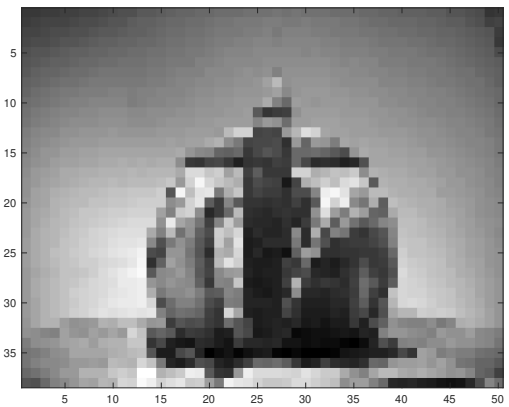
(b) 1 reconstructed



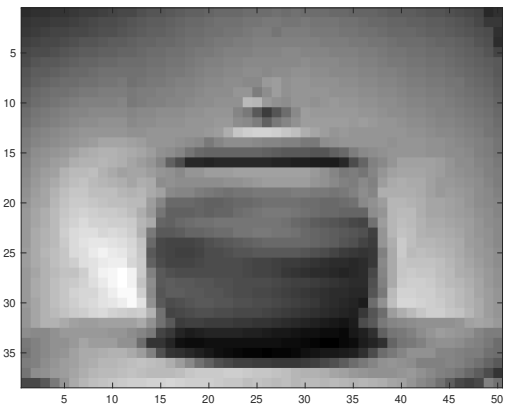
(c) 2



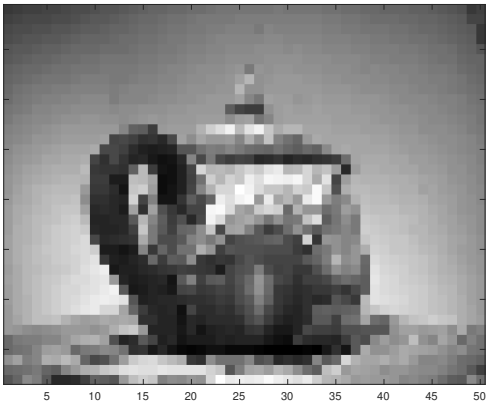
(d) 2 reconstructed



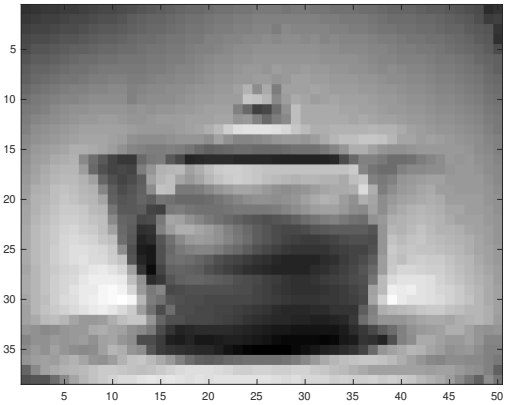
(e) 3



(f) 3 reconstructed

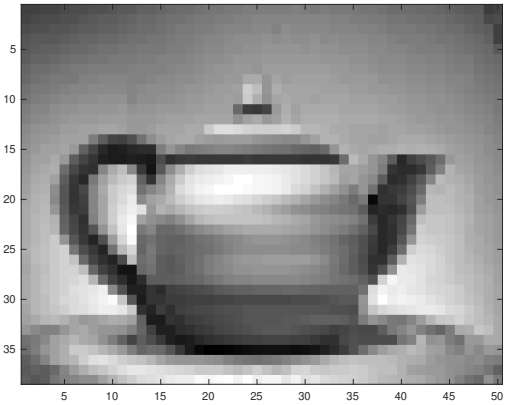


(g) 4



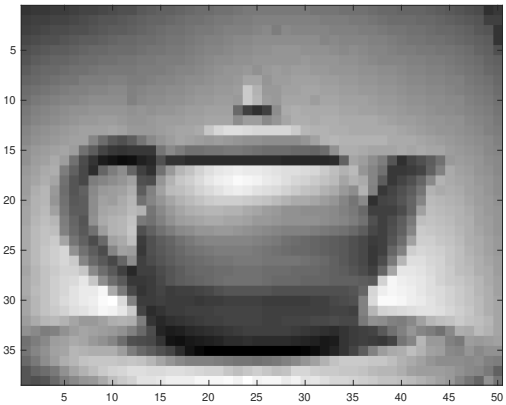
(h) 4 reconstructed

(i) 5

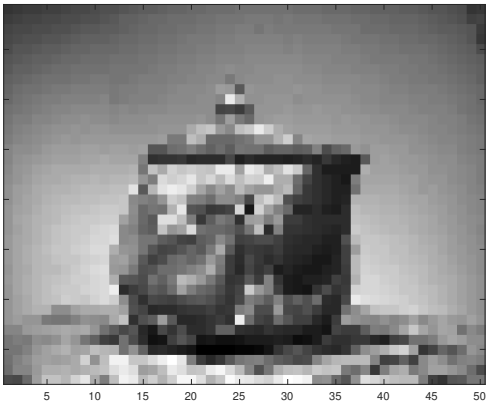


(j) 5 reconstructed

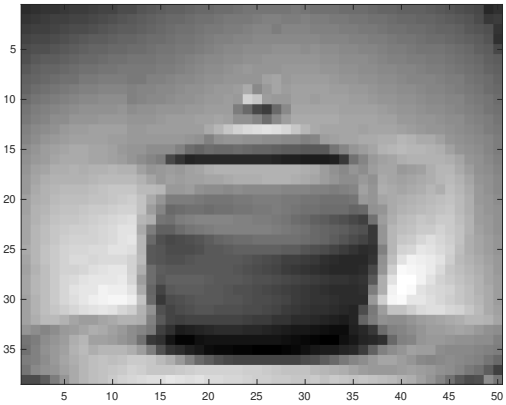
(k) 6



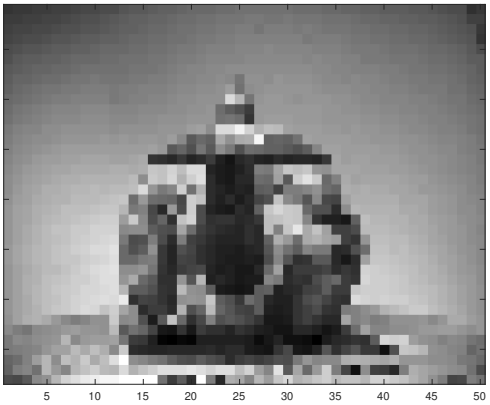
(l) 6 reconstructed



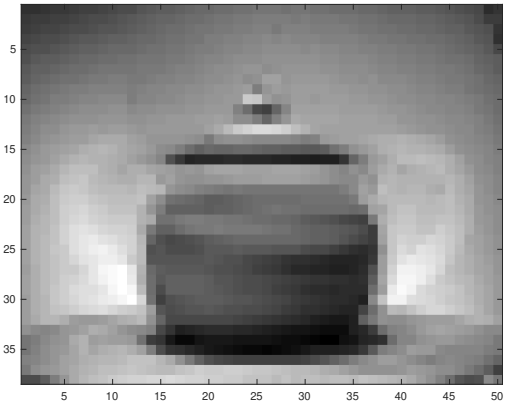
(m) 7



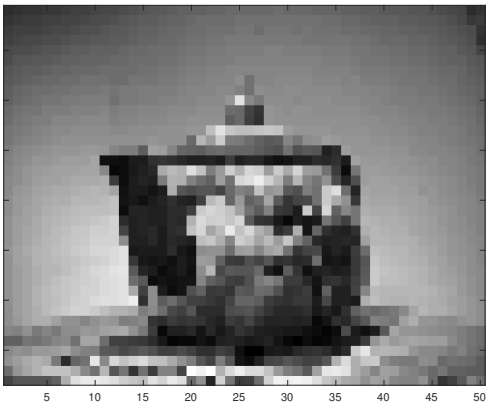
(n) 7 reconstructed



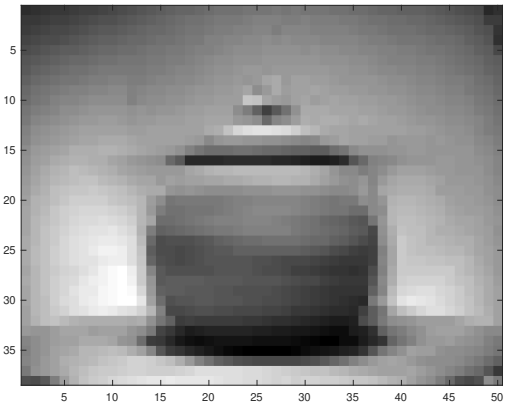
(o) 8



(p) 8 reconstructed

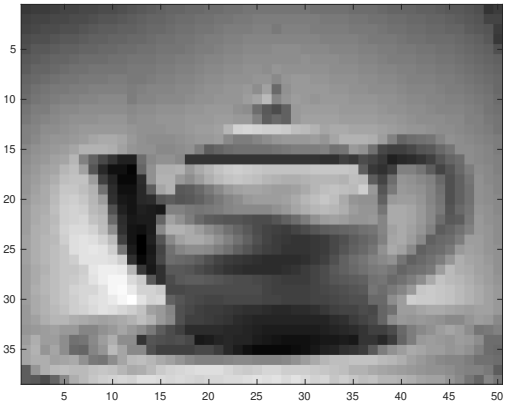


(q) 9



(r) 9 reconstructed

(s) 10



(t) 10 reconstructed