Dis

1

For margin errors,

$$\xi_i^* = 1 - y_i (w^{*T} x_i + b^*)$$

$\therefore$

$$y_i (w^{T} x_i + b) = 1 - \xi_i \quad ①$$

$$w^T x_i = ((1-\xi_i)/y_i - b)$$

The eq. of hyperplane is

$$y_i (w^T x + b) = 1 \quad ②$$

$$\text{ie } w^T x + b - 1/y_i = 0$$

~~Both of these~~ hyperplanes We know distance of a point (z) from a hyperplane is $(w^T x + b)$ is given by

$$\|r\| = \frac{|w^T z + b^*|}{\|w\|}$$

Here z is x_i

$$\therefore w^T x_i = \frac{1-\xi_i}{y_i} - b$$

& $b^* = b - 1/y_i$

$$\therefore |r| = \frac{|(\cancel{1} - \xi_i)/y_i - \cancel{b} + \cancel{b} - \cancel{1}/y_i|}{\|w\|}$$

$$|r| = \frac{|\xi_i / y_i|}{\|w\|}$$

$\therefore |r| \propto \xi_i$

& The constant of proportionality

$$= \frac{1}{|y_i| \, \|w\|}$$

Q2

$\sigma = 4$

$C = 512$

CV_error = 0.2480

Test_error = 0.182

Support vector = 342.

Q3 Using Rayleigh Quotient

~~$\|X - PX\|_F^2 = Tr(X^T X) - Tr(X^T P X)$~~

~~where $P = AA^T$~~

$$\min_{\mu, A, \{\theta_i\}} \sum \|x_i - \mu - A\theta_i\|^2 = \min_{P \in P_K} \|X - PX\|_F \quad (1)$$

Now using rayleigh quotient

$$\|X - PX\|_F^2 = Tr((X - PX)^T (X - PX))$$

$$= Tr(X^T X) - Tr(X^T P X)$$

$$Tr(X^T P X) = \underbrace{Tr(A^T X X^T A)}$$

we need to maximize $Tr(A^T X X^T A)$ in order to minimize $\|X - PX\|_F$.

$\max_{A \in A_K} Tr(A^T C A)$ is

$A = [u_1 \ldots u_K]$

where $C = U \Lambda U^T$

Scanned by CamScanner

$\therefore \max_{A \in \mathcal{A}_k} tr(A^T X X^T A)$ is when

$A = [u_1 \; ... \; u_k]$

& $X X^T = \underbrace{U \Lambda U^T}_{d\times n \;\; n\times d}$

$\therefore \; tr(A^T X X^T A) = tr([u_1 \cdots u_k]^T U \Lambda U^T [u_1 \cdots u_k])$

$= tr[[u_1 \cdots u_k][u_1 \cdots u_k]^T U \Lambda U^T]$

$\cancel{[u_1 \; .... \; u_k]^T}$

$\cancel{[u_1 \; ... \; u_k][u_1 \; ... \; u_k]^T} =$

$$\cancel{\begin{pmatrix} 1 & & & \\ & 1 & & 0 \\ & & 1 & \\ 0 & & & 1 \\ 0 & 0 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & & 0 & 0 \end{pmatrix}}$$

$$\underbrace{[\underset{d\times 1}{u_1} \; ... \; u_k]^T}_{k\times d} \underbrace{[u_1 \; ... \; u_d]}_{d\times d} = \begin{bmatrix} 1 & & & \\ & 1 & 0 & \\ 0 & & 1 & \\ & & & 1 \; 0 \; 0 \end{bmatrix}_{k\times d}$$

k^{th} position.

||ly.

$$\underbrace{[u_1 \; .... \; u_d]^T}_{d\times d} \underbrace{[u_1 \; ... \; u_k]}_{d\times k} = \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \\ 0 & & 0 \\ \vdots & & \vdots \\ 0 & 0 & 0 \end{bmatrix}_{d\times k}$$

$$= tr\left(\begin{bmatrix} 1 & & & 0 \cdots 0 \\ & \ddots & & 0 \cdots 0 \\ & & 1 & 0 \cdots 0 \end{bmatrix}_{k \times d} \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_d \end{bmatrix}_{d \times d} \begin{bmatrix} 1 & & 0 \\ 0 & \ddots & 1 \\ 0 & & 0 \end{bmatrix}_{d \times k} \right)$$

$$= n \sum_{i=1}^{k} \lambda_i$$

||ly

$$tr(XX^T) = tr(U \Lambda U^T)$$

$$= tr(\underbrace{U^T U}_{nI} \Lambda)$$

$$= tr_d(n \Lambda)$$

$$= n \sum_{i=1}^{d} \lambda_i$$

$$\therefore \min_{\mu, A, \theta_i} \| x_i - \mu - A\theta_i \|^2 = n \sum_{i=1}^{d} \lambda_i - n \sum_{i=1}^{k} \lambda_i$$

$$= n \sum_{k+1}^{d} \lambda_i$$

b) ~~For the subspace A to be unique~~ ~~the solution to~~ Φ

~~We know, the~~ for subspace A to be unique the solution of above equation should be unique ie the eigenvalue of λ_k should be different from λ_{k+1}.

Other way to look at this is that

the solution to

$\min \sum \|x_i - \mu - A\theta_i\|^2 = Tr(XX^T) - Tr(A^T XX^T A)$

Here $Tr(XX^T)$ is deterministic

& max $tr(A^T XX^T A)$ is when A is $[u_1 \cdots u_k]$

When $\lambda_k = \lambda_{k+1}$ then there can be two possible choices for u_k & A will not be unique.

$\therefore$ Both necessary & sufficient cond is $\lambda_k \neq \lambda_{k+1}$ for A to be unique.

Q4 a)

	Principal components	% reduction
95% :	43	0.9787
99% :	167	0.9187

b) The 2^{nd} principal component is capturing the case in which the lighting is lower left side of cheek. The 10th principal component is capturing nose, 15th is capturing people with big lips. So, the components are capturing different lighting conditions & features.

```matlab
clear all;
close all;
clc;
load diabetes_scaled.mat;

% Creating test and train data sets
X_tr = X(1:500,:);
X_tt  = X(501:end,:);
Y_tr = y(1:500);
Y_tt  = y(501:end);

grid_sigma = 2.^(0:5);
grid_C = 2.^(6:11);
tol = 0.01;
[n, p] = size(X_tr);
% Cauchy kernel
cauchy_kernel = @(u,v,sigma) (1 + dist2(u, v)/(sigma^2)) .^-1;

num_sv = zeros(length(grid_sigma), length(grid_C));
CV_error_grid = zeros(length(grid_sigma), length(grid_C));

for i = 1:length(grid_sigma)
    grid_sigma_iter = grid_sigma(:,i);
    for j = 1:length(grid_C)
        grid_C_iter = grid_C(:,j);
            CV_error_iter = 0;
            for k = 1:5
                ind = (k-1)*100 + 1: k*100;

                X_tr_iter = X_tr;
                X_tr_iter( ind,:) = [];
                X_CV_iter = X_tr(ind,:);

                y_tr_iter = Y_tr;
                y_tr_iter(ind,:) = [];
                y_CV_iter = Y_tr(ind,:);

                kmat_tr = cauchy_kernel(X_tr_iter, X_tr_iter,
grid_sigma_iter);
                [alpha, bias] = smo(kmat_tr, y_tr_iter', grid_C_iter, tol);
                kmat_CV = cauchy_kernel(X_CV_iter, X_tr_iter,
grid_sigma_iter);
                y_pred = sign(kmat_CV * (y_tr_iter .* alpha') + bias);
                CV_error_iter = 1 - sum((y_pred == y_CV_iter))/100 +
CV_error_iter;
            end
            CV_error_grid(i, j) = CV_error_iter / 5;
    end
end

[min_error, ind] = min(CV_error_grid(:));
[m,n] = ind2sub(size(CV_error_grid),ind);

sigma_opt = grid_sigma(m); %4
C_opt = grid_C(n); %512

% Selected parameters: (sigma, C) :: (4, 512)
```

```
% CV error: min_error :: 0.2480

% Final model
kmat_tr = cauchy_kernel(X_tr, X_tr, sigma_opt);
[alpha, bias] = smo(kmat_tr, Y_tr', C_opt, tol);
kmat_tt = cauchy_kernel(X_tt, X_tr, sigma_opt);
y_pred = sign(kmat_tt * (Y_tr .* alpha') + bias);
error_tt = 1 - sum((y_pred == Y_tt))/size(Y_tt, 1);
% test error :: 0.182
500 - sum(alpha == 0);
%number of support vectors :: 342
```

Question 4

```
clear;
clc;

load yalefaces; % loads the 3-d array yalefaces
% for i=1:size(yalefaces,3)
%     x = double(yalefaces(:,:,i));
%     imagesc(x);
%     colormap(gray)
%     drawnow
%     %pause(.1)
% %     [U, S, V] = svd(x);
% end

yalefaces_mat = double(reshape(yalefaces, [], 2414)');
x_mean = mean(yalefaces_mat);
x_mean_mat = ones(size(yalefaces_mat))* diag(x_mean);
cov_base = yalefaces_mat - x_mean_mat;
cov_mat = (cov_base' * cov_base) ./ 2414;
[U, D] = eig(cov_mat);
eig_values = sum(D);
[sort_eig_values, index] = sort(eig_values,'descend');
semilogy(sort_eig_values)
for i = 1:length(sort_eig_values)
    var_cap = sum(sort_eig_values(:, 1:i)) / sum(sort_eig_values);
    if var_cap >= .95
        break
    end
end
% .95 variation captured: 43
% % dim reduction: .9787
for i = 1:length(sort_eig_values)
    var_cap = sum(sort_eig_values(:, 1:i)) / sum(sort_eig_values);
    if var_cap >= .99
        break
    end
end
% .99 variation captured: 167
% % dim reduction: .9172
subplot(5,4, 1)
```

```
a = reshape(x_mean, 48, 42);
imagesc(a);
colormap(gray)
drawnow

for i = 1:19
    subplot(5, 4, i+1);
    x = reshape(U(:, index(i)), 48, 42);
    imagesc(x);
    colormap(gray)
    drawnow
end
```