HW8

November 10, 2019

[4]:
```
M = csvread('HW8_dat.csv',1);
M = transpose(M);
X = M(2,:);
Y = M(1,:);
Y_T = transpose(Y);
XX = linspace(0,2,200);

% Gaussian
K_G = KappaMatrix(X,'Gaussian');
alpha_G = RKHSRegression(K_G,Y_T,0.01);
YY_G = zeros(1,200);
for i = 1:200
    YY_G(i) = RegressionFunction(XX(i),X,alpha_G,'Gaussian');
end
% figure();
% scatter(X,Y);
% hold on;
% plot(XX,YY_G,'LineWidth',1,'Color','black');
% hold off;
RSS(X,Y,alpha_G,'Gaussian')

% Polynomial
K_P = KappaMatrix(X,'Polynomial');
alpha_P = RKHSRegression(K_P,Y_T,0.1);
YY_P = zeros(1,200);
for i = 1:200
    YY_P(i) = RegressionFunction(XX(i),X,alpha_P,'Polynomial');
end
% figure();
% scatter(X,Y);
% hold on;
% plot(XX,YY_P,'LineWidth',1,'Color','black');
% hold off;
RSS(X,Y,alpha_P,'Polynomial')

% Laplacian
K_L = KappaMatrix(X,'Laplacian');
```

```
alpha_L = RKHSRegression(K_L,Y_T,1);
YY_L = zeros(1,200);
for i = 1:200
    YY_L(i) = RegressionFunction(XX(i),X,alpha_L,'Laplacian');
end
% figure();
% scatter(X,Y);
% hold on;
% plot(XX,YY_L,'LineWidth',1,'Color','black');
% hold off;
RSS(X,Y,alpha_L,'Laplacian')

figure();
scatter(X,Y);
hold on;
plot(XX,YY_G,'Linewidth',1,'Color','black');
hold on;
plot(XX,YY_P,'Linewidth',1,'Color','red');
hold on;
plot(XX,YY_L,'Linewidth',1,'Color','green');
legend('Original Data','Gaussian','Polynomial','Laplacian')
hold off;

function G = GaussianKernel(x,y)
    G = exp(-(norm(x-y)^2/0.25));
end

function P = PolynomialKernel(x,y,c)
    P = (dot(x,y)+c)^2;
end

function L = LaplacianKernel(x,y)
    L = exp(-norm(x-y));
end

function K = KappaMatrix(X,type)
  p = size(X,1);
  n = size(X,2);
  K = zeros(n,n);
  for i=1:n
      xx = X(:,i);
      for j=i:n
         yy = X(:,j);
         if type == "Gaussian"
           K(i,j)=GaussianKernel(xx,yy);
         elseif type == "Polynomial"
           K(i,j)=PolynomialKernel(xx,yy,1);
```

```
         elseif type == "Laplacian"
           K(i,j)=LaplacianKernel(xx,yy);
         end
         K(j,i)=K(i,j);
      end
   end
end

function alpha = RKHSRegression(K,Y,lambda)
     n = size(K,1);
     alpha = (K+lambda*eye(n,n))\Y;
end

function y = RegressionFunction(x,X,alpha,type)
    n = size(X,2);
    y = 0;
    for i = 1:n
         xx = X(:,i);
         if type == "Gaussian"
           y = y + alpha(i)*GaussianKernel(xx,x);
         elseif type == "Polynomial"
           y = y + alpha(i)*PolynomialKernel(xx,x,1);
         elseif type == "Laplacian"
           y = y + alpha(i)*LaplacianKernel(xx,x);
         end
    end
end

function residual = RSS(X,Y,alpha,type)
    residual = 0;
    n = size(X,2);
    for i = 1:n
        diff = Y(i) - RegressionFunction(X(i),X,alpha,type);
        residual = residual + diff*diff;
    end
    residual = residual/n;
end
```

```
          File "<tokenize>", line 90
        end
        ^
    IndentationError: unindent does not match any outer indentation level
```

[6]:
```
L = load('MNIST_20x20.mat');
labels = L.labels;
imgs = L.imgs;
n = size(labels,1);
counter = zeros(4);
for i = 1:n
    if labels(i) == 1
        counter(1) = counter(1) + 1;
        IMG(counter(1),1) = i;
    elseif labels(i) == 2
        counter(2) = counter(2) + 1;
        IMG(counter(2),2) = i;
    elseif labels(i) == 3
        counter(3) = counter(3) + 1;
        IMG(counter(3),3) = i;
    elseif labels(i) == 4
        counter(4) = counter(4) + 1;
        IMG(counter(4),4) = i;
    end
end

% pair: k & l
for k = 1:3
    for l = k+1:4
        total = counter(k)+counter(l);
        X = zeros(401,total);
        for i = 1:counter(k)
            X(1:400,i) = reshape(imgs(:,:,IMG(i,k)),[400,1]);
            X(401,i) = 1;
        end
        for i = 1:counter(l)
            j = counter(k)+i;
            X(1:400,j) = reshape(imgs(:,:,IMG(i,l)),[400,1]);
            X(401,j) = -1;
        end

        ntraining = round(total*0.6);
        temp = zeros(401,1);
        for i = 1:ntraining
            r = randi([i,total]);
            temp = X(:,i);
            X(:,i) = X(:,r);
            X(:,r) = temp;
        end
        K = KappaMatrix(X(1:400,1:ntraining),'Polynomial');
        alpha = RKHSRegression(K,transpose(X(401,1:ntraining)),10);
```

```
        correct = 0;
        for i = ntraining+1:total
            xx = X(1:400,i);
            yy = RegressionFunction(xx,X(1:400,1:ntraining),alpha,'Polynomial');
            if yy*X(401,i)>0
                correct = correct + 1;
            end
        end
        accuracy(k,l) = correct/(total-ntraining);
    end
end




function P = PolynomialKernel(x,y,c)
    P = (dot(x,y)+c)^2;
end

function K = KappaMatrix(X,type)
  p = size(X,1);
  n = size(X,2);
  K = zeros(n,n);
  for i=1:n
      xx = X(:,i);
      for j=i:n
         yy = X(:,j);
         if type == "Gaussian"
           K(i,j)=GaussianKernel(xx,yy);
         elseif type == "Polynomial"
           K(i,j)=PolynomialKernel(xx,yy,1);
         elseif type == "Laplacian"
           K(i,j)=LaplacianKernel(xx,yy);
         end
         K(j,i)=K(i,j);
      end
   end
end

function alpha = RKHSRegression(K,Y,lambda)
     n = size(K,1);
     alpha = (K+lambda*eye(n,n))\Y;
end

function y = RegressionFunction(x,X,alpha,type)
    n = size(X,2);
```

```
    y = 0;
    for i = 1:n
         xx = X(:,i);
         if type == "Gaussian"
           y = y + alpha(i)*GaussianKernel(xx,x);
         elseif type == "Polynomial"
           y = y + alpha(i)*PolynomialKernel(xx,x,1);
         elseif type == "Laplacian"
           y = y + alpha(i)*LaplacianKernel(xx,x);
         end
    end
end
```

```
          File "<tokenize>", line 85
        end
        ^
    IndentationError: unindent does not match any outer indentation level
```

[ ]: