Ejemplo Kernel Regresión mínimos cuadrados (Kernel Ridge Regression)

Presentación en vídeo en: http://personales.upv.es/asala/YT/V/kerlsm.html

Este código ejecutó correctamente en Matlab R2018b

Tabla de Contenidos

1.- Generación de los datos

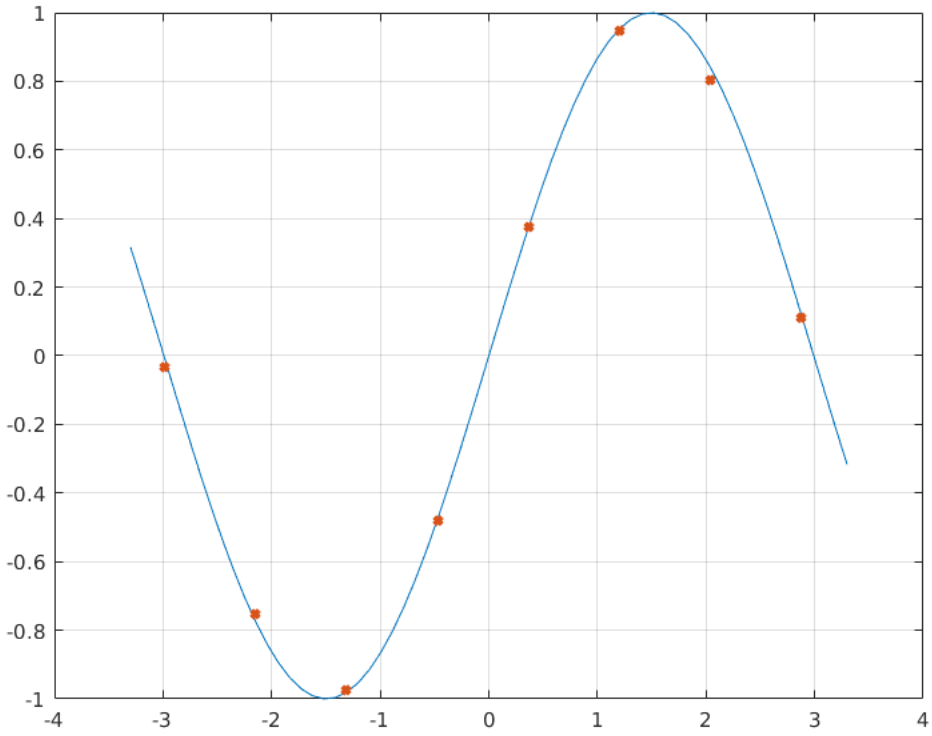
```
Ntodos=64;
x_todos=(0:(Ntodos-1))*2.1*pi/(Ntodos-1)-1.05*pi;
y_todos=sin(1.05*x_todos);
```

Entrenamos con un subconjunto de los datos, contaminados con un poco de ruido de medida

```
X=x_todos(4:8:end);
Y=y_todos(4:8:end);
N=length(X)
```

```
N = 8
```

```
Y=Y+randn(1,N)*0.02;
plot(x_todos,y_todos)
hold on
plot(X,Y,'x','LineWidth',3), grid on, hold off
```



2.- Regresión

Hagamos distintas pruebas de Kernels:

```
prueba=4;
c=.03; %parámetro de regularización
switch prueba
    case 0
    % a mano el kernel K con phi explícito:
        phi=@(x) [x;x.^2;ones(size(x));sin(x*0.8)];
        XPhi=phi(X);
        size(XPhi)
        K=XPhi'*XPhi;
        %pero con phi de tamaño 3, mejor pseudoinversa de siempre!!!!
        thetaclasicoasinregularizar=Y*pinv(XPhi)
        thetaclasico_conregularizacion=Y*XPhi'/(XPhi*XPhi'+c*eye(4))
    case 1
        %prueba 1: lineal
        kappa=@(x,y) (1+1*x'*y);
    case 2
        %prueba 2: polinomial, probar grados 3, 10
        kappa=@(x,y) (1+1*x'*y)^10;%Polynomial Kernel
    case 3
        %prueba 3: polinomial con más regularización a grados altos
        kappa=@(x,y) (1+0.15*x'*y)^7;%Polynomial Kernel
    case 4
```

```
        %prueba 4: Kernel RBF
        sigma=2.0; %probar sigma 0.2, 2 y 20
        kappa=@(x,y) exp(-norm(x-y)^2/sigma^2);%Gaussian Kernel
end
if(prueba>0)
    K=generaKernel(X,kappa)
end
```

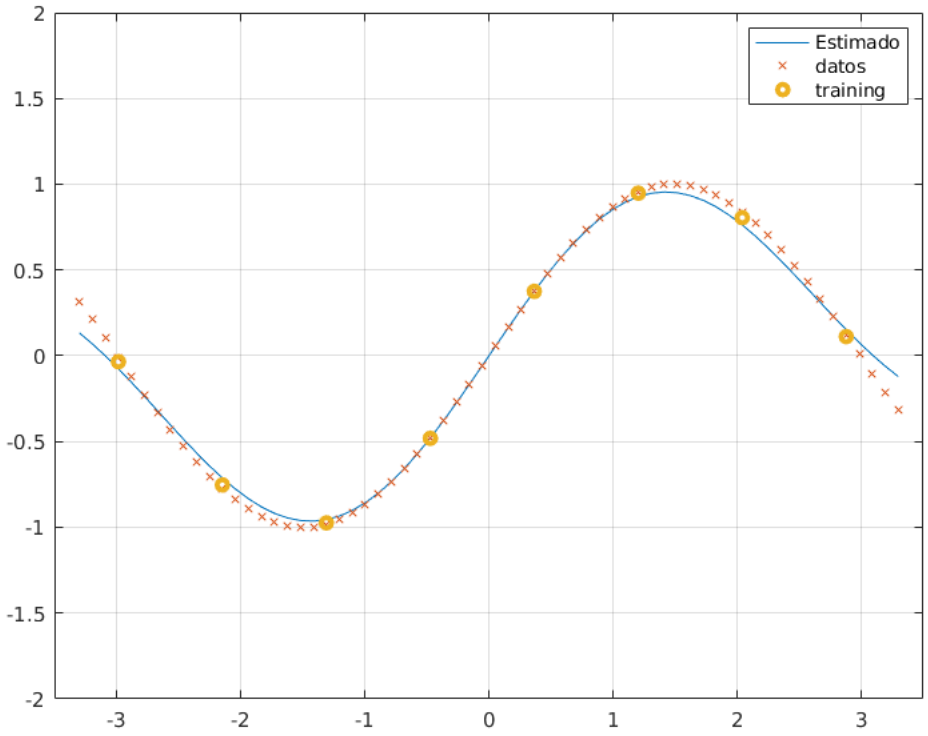
```
K = 8×8
    1.0000    0.8391    0.4957    0.2062    0.0604    0.0124    0.0018    0.0002
    0.8391    1.0000    0.8391    0.4957    0.2062    0.0604    0.0124    0.0018
    0.4957    0.8391    1.0000    0.8391    0.4957    0.2062    0.0604    0.0124
    0.2062    0.4957    0.8391    1.0000    0.8391    0.4957    0.2062    0.0604
    0.0604    0.2062    0.4957    0.8391    1.0000    0.8391    0.4957    0.2062
    0.0124    0.0604    0.2062    0.4957    0.8391    1.0000    0.8391    0.4957
    0.0018    0.0124    0.0604    0.2062    0.4957    0.8391    1.0000    0.8391
    0.0002    0.0018    0.0124    0.0604    0.2062    0.4957    0.8391    1.0000
```

```
w=Y/(K+c*eye(N)) %solución óptima
```

```
w = 1×8
    1.3749   -1.3498   -0.6273   -0.0724   -0.0504    0.6826    1.4108   -1.3452
```

3.- Comprobación del ajuste resultante

```
 Ntodo=length(x_todos);
 lista=zeros(1,Ntodo);
 if(prueba==0)
     theta=w*XPhi'
 end
for i=1:Ntodo
     if(prueba>0)
        lista(i)=modeloestimado(x_todos(i),X,kappa,w);
     else %explícito:
         lista(i)=theta*phi(x_todos(i));
     end
end
plot(x_todos,lista)
hold on
plot(x_todos,y_todos,'x')
plot(X,Y,'o','LineWidth',3)
hold off, grid on, axis([-3.5 3.5 -2 2])
legend('Estimado','datos','training')
```




```
function K=generaKernel(X,kappa)
N=size(X,2);
K=zeros(N,N);
for i=1:N
    K(i,i)=kappa(X(i),X(i));
    for j=(i+1):N
        K(i,j)=kappa(X(i),X(j));
        K(j,i)=K(i,j);
    end
end
end

function yestimada=modeloestimado(x,X,kappa,w)
yestimada=0;
N=size(X,2);
 for i=1:N
     yestimada=yestimada+w(i)*kappa(X(i),x);
 end

end
```