CLP Lab 5 Report

Albert Aparicio Isarn
albert.aparicio.isarn@alu-etsetb.upc.edu

Héctor Esteban
hect.esteban@gmail.com

1. Obtención del clasificador SVM

El resultado de la ejecución inicial del programa, con $P = 0{,}1$ y $h = 1$, ha dado los siguientes errores de clasificación:

Fase / **Clasificador**	**Training**	**Test**
SVM Lineal	0,065942	0,0728261
Kernel Gaussiano	0,391304	0,348913

Cuadro 1: Errores LC y QC obtenidos en entreno y en test para cada una de las tres dimensiones. $SNR = 10dB$

Contrariamente a lo que se podría esperar, en esta situación, el clasificador lineal da un resultado notablemente superior al clasificador con kernel Gaussiano. Como se ve en la figura 1, en el caso de $P = 0{,}1$ y $h = 1$, la probabilidad de error es alta.

Después de programar el doble bucle para obtener los valores de P y h óptimos, el gráfico bidimensional obtenido se muestra en la figura 1.

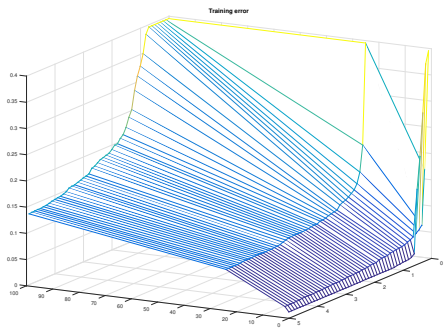

Figura 1: Representación del error de clasificación respecto de h y P

Los valores óptimos son: $P = 1{,}21$ y $h = 2{,}5$.
El código generado es el siguiente:

```
%% Clasificador kernel gaussiano
    P_opt = 0;
    h_opt = 0;
    err_val_opt = Inf;

    P = 0.01:0.1:5;
    h = [1, 2.5, 25, 100];

    err_train_val = zeros(length(P), length(h), 2);

    for j = 1:length(P)
        for i=1:length(h)
            Gauss_model = fitcsvm(X_train, Labs_train, 'BoxConstraint',P(
    j),...
                'KernelFunction','RBF','KernelScale',h(i));
            fprintf(1,'\n Clasificador Kernel Gaussiano\n')
            Gauss_out = predict(Gauss_model, X_train);
            Err_train=sum(Gauss_out~=Labs_train)/length(Labs_train);
            fprintf(1,'error train = %g   \n', Err_train)

            Gauss_out = predict(Gauss_model, X_val);
            Err_val=sum(Gauss_out~=Labs_val)/length(Labs_val);
            fprintf(1,'error val = %g   \n', Err_val)
            fprintf(1,'\n  \n  ')
            % Test confusion matrix
            CM_Gauss_test = confusionmat(Labs_val,Gauss_out)

            err_train_val(j, i, 1) = Err_train;
            err_train_val(j, i, 2) = Err_val;

            if Err_val < err_val_opt
                err_val_opt = Err_val;
                P_opt = P(j);
                h_opt = h(i);
            end
%             clear Err_train Err_test Gauss_out
        end
    end

    %% Plot 3D graphic
    figure
    mesh(h, P, err_train_val(:,:,1)), hold on
    title('Training error')
    hold off

    figure
    mesh(h, P, err_train_val(:,:,2)), hold on
    title('Validation error')
    hold off
```

s1_ex3_report.m

Una vez encontrado el clasificador óptimo, sobre la base de datos de test, el error obtenido es de $p_{e_{test}} = 0{,}0543478$. En comparación con la probabilidad encontrada en la primera ejecución ($p_e = 0{,}34891$), el clasificador óptimo mejora sustancialmente el rendimiento de la clasificación.

2. Análisis de la bondad del clasificador

Los indicadores de clasificación obtenidos con los parámetros óptimos son:

$$Ec = 0{,}0543 \tag{1}$$
$$A = 0{,}9457 \tag{2}$$
$$P = 0{,}9377 \tag{3}$$
$$S = 0{,}9348 \tag{4}$$
$$Es = 0{,}9608 \tag{5}$$
$$Fscore = 0{,}9298 \tag{6}$$

Para un caso en que se tengan 400 vectores SPAM y 4600 vectores MAIL, usar los cocientes P, S, E_s y F_{score} es más adecuado porque los errores al clasificar SPAM como MAIL serán escasos, ya que hay pocos ejemplos de SPAM, respecto de MAIL.

Esto hará que este tipo de errores tenga poca relevancia respecto la clasificación de los ejemplos de MAIL.

Los cocientes P, S, E_s y F_{score} valoran el comportamiento de la clasificación, fijándose en una clase cada vez, de manera que la proporción entre los ejemplos de una clase y otra no se tienen en cuenta.

3. Análisis de la validez de decisión

Las probabilidades a priori de las dos clases, después de clasificar son:

$$P_{mail} = 0{,}5848 \tag{7}$$
$$P_{spam} = 0{,}4152 \tag{8}$$

La predicción del vector de análisis ha sido $V_{analisis_{pred}} = 0(MAIL)$. El parámetro $Pred_{validity} = 1$ indica que la predicción es correcta.

El vector no contiene la palabra "Make"ni la palabra .ªddress".

La probabilidad a posteriori es de:

$$
\begin{aligned}
P(A|B) &\equiv Pr\{V_{analisis} = MAIL | clasificador = MAIL\} \\
P(A|B) &= \frac{P(B|A) \cdot P(A)}{P(B)} \\
P(B|A) &\equiv Pr\{clasificador = MAIL | V_{analisis} = MAIL\} \equiv S \\
P(A) &= \frac{N_{mail}}{N_{train}}, or \frac{N_{spam}}{N_{train}} \\
P(B) &= P_{mail}, or P_{spam}
\end{aligned}
\tag{9}
$$

Para el caso de $V_{analisis}$, los valores de las probabilidades son:

$$P(A) = 0{,}6014 \tag{10}$$
$$P(B) = 0{,}5848 \tag{11}$$
$$P(B|A) = 0{,}9348 \tag{12}$$
$$P(A|B) = 0{,}9615 \tag{13}$$

El código que genera estos datos es el siguiente:

```
%% A Posteriori Probability
%     P(B) = P_mail or P_spam
%     P(A) => A priori prob depending on DB
%     P(B|A) = S
%     P(A|B) = P(B|A)*P(A)/P(B)

    PBA = S

    if V_analisis_pred == 0 % Not SPAM
        PA = sum(Labs_train == 0)/length(Labs_train)
        PB = P_mail
    elseif V_analisis_pred == 1 % SPAM
        PA = sum(Labs_train == 1)/length(Labs_train)
        PB = P_spam
    end

    PAB = PBA*PA/PB
```

s3_ex3_report.m