Práctica 2

Pablo Arranz Ropero
Juan Alberto Camino Sáez
Grupo 2

Práctica 2: Regresión logística

En esta práctica, que se divide en dos apartados, se trata de aplicar regresión logística a dos conjuntos de datos. En el primer caso se trata de regresión logística asumiendo que la ecuación a obtener es una recta y en el segundo caso se trata de regresión logística regularizada con potencias de x_1 y x_2 hasta la sexta potencia (28 atributos en total).

Regresión logística asumiendo una recta

En el primer caso, tenemos los datos en el fichero *ex2data1.txt*, en el cual se encuentran los resultados de unos estudiantes en dos exámenes y si fueron admitidos o no.
Primeramente hemos visualizado los datos en pantalla con el siguiente código.

```
negativos = find(datos(:, 3) == 0)
positivos = find(datos(:, 3) == 1)
plot(datos(negativos, 1),
datos(negativos, 2), 'ko', 'MarkerFaceColor', 'y', 'MarkerSize', 7, datos(positivos, 1),
datos(positivos, 2), 'ko', 'marker', '+');
```

Para este caso, hemos creado un fichero llamado *sigmoide.m,* en el cual calculamos el valor de la función sigmoide para un número, vector o matriz.

```
function [sigmoide] = sigmoide(z)
    for i = 1:rows(z)
        for j = 1:columns(z)
            sigmoide(i,j) = 1/(1+ (e.^((-1)*z(i,j))));
        endfor
    endfor
endfunction
```

Esta función sigmoide la utilizaremos en el cálculo de la función de coste y en el cálculo del gradiente, ya que el valor que nos devuelva la función sigmoide será el valor de la hipótesis en la regresión logística.

Crearemos un fichero llamado *coste.m* que se encargará de calcular la función de coste y el valor del gradiente para unas determinadas thetas. Esta función es la que pasaremos a la función *fminunc* para que realice el descenso de gradiente.

```
function [J, grad] = coste(theta, X, y)

  valhipotesis = sigmoide((-1)*theta'*X');
  m = rows(X);

  valory0 = ((-1) * y' * log(valhipotesis)');
  valory1 = ((1-y)' * log(1-valhipotesis)');

  J = (1/m)*(valory0-valory1);
```

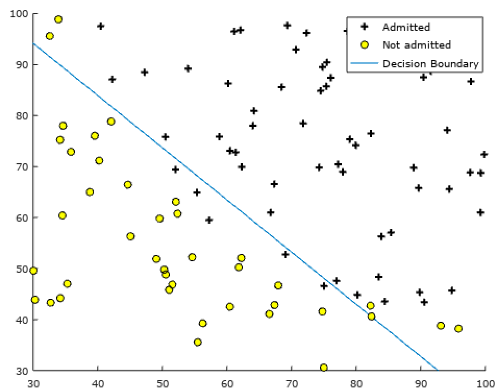
```
        grad = (-1/m) * (valhipotesis' - y)' * X;

      endfunction
```

Con 0s como thetas iniciales, la función de coste tendrá un valor de 0.69 y los gradientes tendrán un valor de 0.1, 12.01 y 11.26.

Llamaremos a la función fminunc pasándole la función que hemos creado y unas thetas iniciales que serán *zeros(3, 1).*

```
opciones = optimset('GradObj' , 'on', 'MaxIter', 1500);
[theta, cost] = fminunc(@(t)(coste(t, X, y)), zeros(3, 1),
opciones);
```

Esto nos dará un coste de 0.203 y unas thetas de 25.16, -0.206 y -0.201. Aplicando estos thetas a la función plotDecisionBoundary obtenemos la siguiente gráfica.



Para evaluar el porcentaje de datos que se han clasificado correctamente crearemos un fichero *percentage.m* que se encargará de calcular la proporción de la siguiente manera:

```
function [percentage] = percentage(theta, X, Y)

    resultados = sigmoide((-1)*theta'*X')';

    resultadoscorrectos = sum(Y - resultados > -0.5 & Y -
resultados <= 0.5);

    percentage = resultadoscorrectos / rows(Y);

endfunction
```

El resultado obtenido nos dice que se han clasificado bien el 89% de los casos.

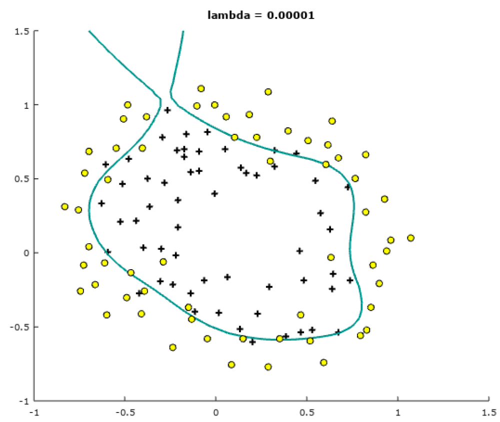
Regresión logística regularizada

En cuanto a la regresión logística regularizada enviaremos nuestra matriz X a la función mapFeature que se encargará de generar términos polinómicos de x_1 y x_2.

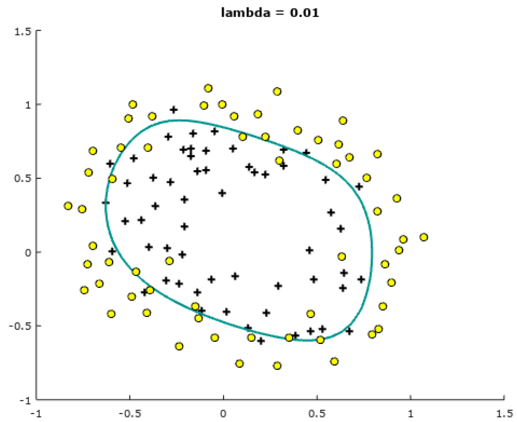
Utilizaremos esta matriz X para calcular la versión regularizada de la función de coste, a la que pasaremos también por parámetro la variable lambda con diferentes valores para ver ejemplos de underfitting y overfitting.

```
function [J, grad] = costereg(theta, X, y, lambda)

  valhipotesis = sigmoide((-1)*theta'*X');
  m = rows(X);

  valory0 = ((-1) * y' * log(valhipotesis)');
  valory1 = ((1-y)' * log(1-valhipotesis)');

  J = (1/m)*(valory0-valory1) + (lambda/(2*m)) * sum((theta(2:rows(theta),:)).^2);

      grad = ((-1/m) * (valhipotesis' - y)' * X + (lambda/m) * theta(2:rows(theta),:))(1, :);

      endfunction
```

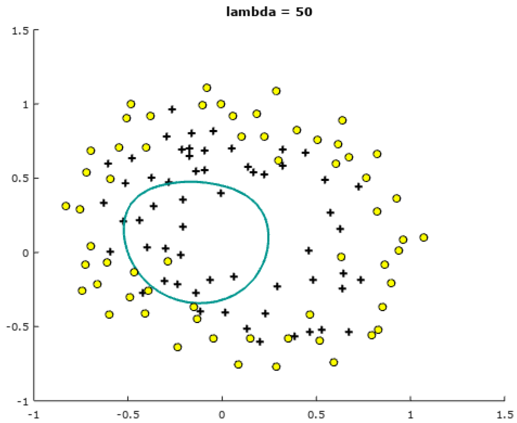
Pasaremos esta función a la función fminunc para obtener los valores de theta óptimos y con los valores theta obtenidos obtendremos las siguientes gráficas en función del lambda aplicado:



Ejemplo de overfitting con un valor de lambda muy pequeño.



Ejemplo de un valor de lambda que proporciona un modelo razonable.



Ejemplo de underfitting con un valor de lambda muy grande.