Aprendizaje Automático - Trabajo Práctico 6

Gonzalo Castiglione - 49138

June 30, 2012

Objetivo: Aplicar diversos métodos de estadística no paramétrica para aprender a hacer inferencia a partir de datos experimentales

1 Aprendizaje basado en instancias

1. Resultados

   A continuación se muestra una tabla comprativa entre las mediciones seleccionadas y el resultado de uso del algoritmo $knnclassify$ (k vecinos mas cercnanos).

	k	Error porcentual cometido	Porcentaje de correctos
Ancho y largo de los pétalos	10	0.1400	***0.8600***
	20	0.1933	0.8067
	30	0.1600	0.8400
Ancho y largo de los cépalos	10	0.040	**0.9600**
	20	0.0460	0.954
	30	0.0460	0.954
Las 4 variables	10	0.0133	**0.9867**
	20	0.0200	0.98
	30	0.0267	0.9733
Seleccion aleatoria (d) * (100 entrenamiento + 50 testeo)	10	0.0200	**0.9800**
	20	0.0200	**0.9800**
	30	0.1000	**0.9000**

   k = Cantidad de vecinos a considerar.

   Los valores en **negrita** representan los mejores valores en cuanto a cantidad de mediciones clasificadas correctamente para esa clase.

Se puede observar que para este probelma, la mejor clasificación de los individuos se produce cuando $k \approx 10$. A excepción de la selección aleatoria, que parece funcionar mejor, permitiendo una clasificación con conjuntos tomando cualquier tamaño de vecinos (entre los valores de testeo al menos) y obteniendo aún mejores resultados.

Otra observación que puede realizarse es que se obtienen resultados mucho mas precisos cuando se clasifican las flores por el tamaño de sus cépalos que tomando el tamaño de los pétalos.

En comparación con los métodos aplicados en los trabajos prácticos anteriores, el único que logró clasificar con un nivel de acierto igual o superior fue el método de *Árbol de desición* (ID3), el cual también obtuvo un acierto del 98%.

* Debido a que se está tomando dos tercios del conjunto original (una buena parte del conjunto), no hace falta verificar que se hallan tomado un poco de cada subconjunto de mediciones (uno por cada tipo de especie) ya que es muy raro que no se tomen suficientes elementos de alguno de los tres conjuntos.

1. Gráficos obtenidos

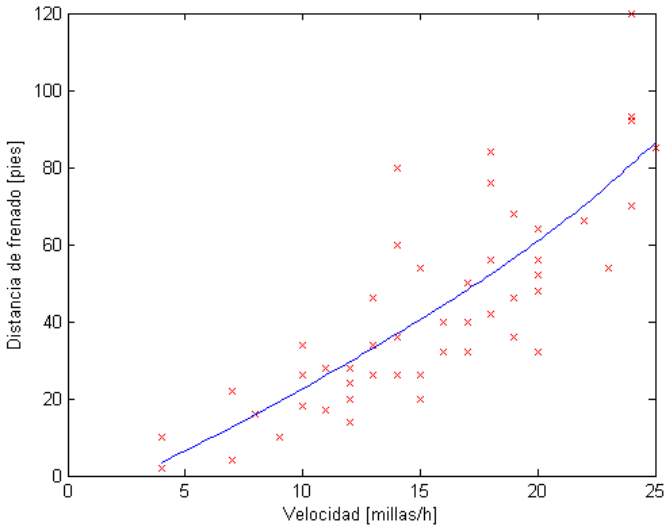

Figure 1: Regresión lineal local pesada con núcleo gaussiano

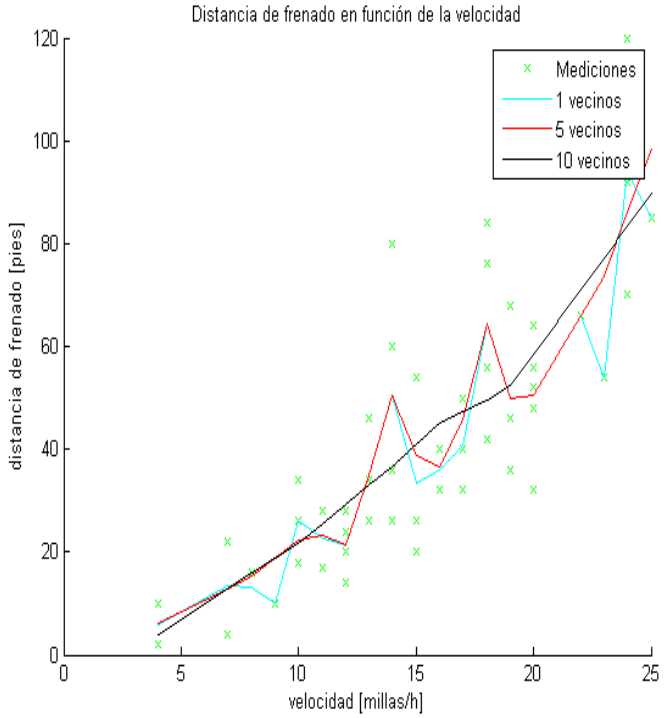

Figure 2: Regresión lineal local pesada con núcleo tricubo

Se puede ver en la *Figura* 2 que, conforme se aumenta la cantidad de vecinos a considerar, se suaviza en cuanto a los picos que genera y se asemeja mas a la curva obtenida en la Figura 1 para la regresión lineal local pesada con núcleo gaussiano.

1. Resultados

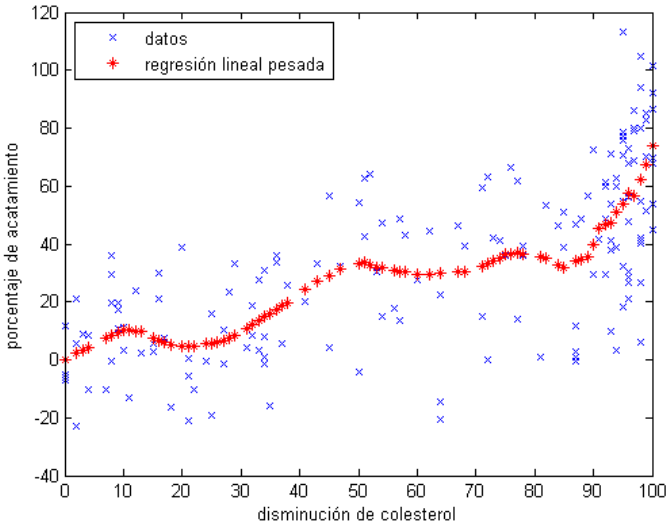

Figure 3: Nivel de colesterol según porcentaje de a acatamiento y regresión lineal

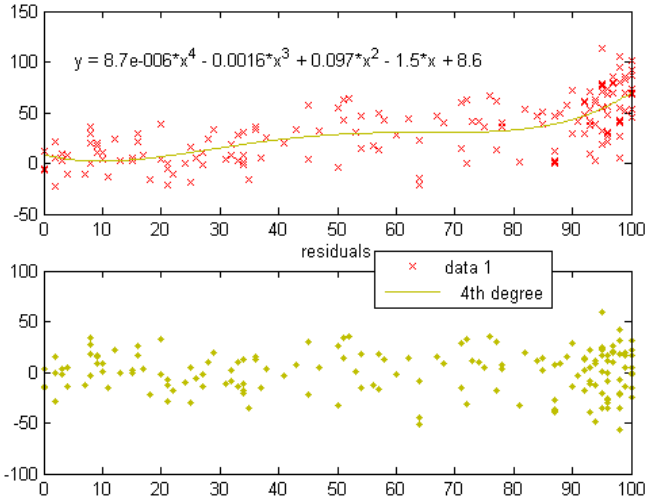

Figure 4: Polynomio de grado 4 que mejor ajusta a los datos

Luego de analizados los datos graficados y su línea de regresión, se puede concluir que la pastilla efectivamente funciona a combatir el colesterol ya que se puede ver que, en la mayoría de los casos, la gente que mejor acató la toma del medicamento sufrió una baja importante en su nivel de colesterol.

2 Código

1. .

```
% a, b, c
% Se puede variariar la asignacion de las variables
% sample, training y k para distintas pruebas.
load fisheriris;
group = species;
sample = meas(:, 1:2);
training = meas(:, 1:2);
checkgroup = species;
distance = 'euclidean';
n = size(sample, 1);
k = 10;
c = knnclassify(sample,training, group, k, distance);
mistake = size(find(strcmp(c,checkgroup)==0),1);
correct = (n - mistake) / n
error = mistake / n
% d -- division en conjunto de entrenamiento y testeo
total = 100;
lenght = size(meas,1);
indexes = randperm(lenght);
g1 = indexes(:, 1:total);
g2 = indexes(:, (total + 1):lenght);
group = species(g1,:);
sample = meas(g2,:);
training = meas(g1,:);
checkgroup = species(g2,:);
```

2. .

```
% a
autosData=load('autos.txt');
x=autosData(:,2);
y=autosData(:,3);
r=ksrlin(x,y);
plot(x, y, 'rx', r.x, r.f, 'b');
xlabel('Velocidad [millas/h]');
ylabel('Distancia de frenado [pies]');
% b
lengthdata = size(y,1);
colors = {'m', 'g', 'c', 'r', 'k'};
colorsize = size(colors,2) vecinos = [1, 5, 10, 30];
j = 1;
hold on;
for i = vecinos
    span = ceil((i *length(y))/lengthdata);
    z = smooth(x, y, span, 'lowess');
    c = mod(j,colorsize)+1;
if i == 1
plot(x,z, strcat(colors{c}, 'x'));
else
plot(x,z, colors{c});
end
    j = j+1;
end
j = 1;
l{j} = 'Mediciones';
```

```
j = j+1;
for i = vecinos
    l{j} = strcat(num2str(i),' vecinos');
    j = j+1;
end
legend(l);
title({'Distancia de frenado en función de la velocidad'}); xlabel('velocidad [millas/h]');
ylabel('distancia de frenado [pies]');
```

3. .

```
colesterol=load('colesterol.txt');
x=colesterol(:,2);
y=colesterol(:,3);
% a
plot(x,y,'bx');
xlabel('disminución de colesterol');
ylabel('porcentaje de acatamiento');
hold on;
% polinomio que mejor ajusta fue creado con la tool del matlab
% c
span = ceil(0.2*length(y));
z = smooth(x,y,span,'lowess');
plot(x,z,'r*');
legend('datos', 'regresión lineal pesada', 'Location', 'NorthWest');
```