Práctica 1 Búsqueda Local APC
Metaheurísticas
Algoritmos KNN, Relief y Búsqueda Local

Ignacio Aguilera Martos
DNI: 77448262V e-mail: nacheteam@correo.ugr.es
Grupo de prácticas 1 Lunes 17:30-19:30

Curso 2017-2018

Índice

1. Introducción del problema

Para el problema de clasificación partimos de un conjunto de datos dado por una serie de tuplas que contienen los valores de atributos para cada instancia. Esto es una n-tupla de valores reales en nuestro caso.
El objetivo del problema es obtener un vector de pesos que asocia un valor en el intervalo $[0, 1]$ indicativo de la relevancia de ese atributo. Esta relevancia va referida a lo importante que es en nuestro algoritmo clasificador ese atributo a la hora de computar la distancia entre elementos.
Resumiendo lo que tenemos es un algoritmo clasificador que utiliza el vector de pesos calculado para predecir la clase a la que pertenece una instancia dada. Este algoritmo clasificador es el KNN con k=1. Lo que hace es calcular según la distancia euclídea (o cualquier otra) la tupla más cercana a la que queremos clasificar ponderando cada atributo con el correspondiente peso del vector, es decir, la distancia entre dos elementos sería:

$$d(e, f) = \sqrt{\sum_{i=0}^{n} w_i * (e_i - f_i)}$$

Donde e y f son instancias del conjunto de datos, w el vector de pesos y n la longitud de e y f que es la misma.
La calificación que se le asigna al vector w depende de dos cosas: la tasa de aciertos y la simplicidad.
La tasa de aciertos se mide contando el número de aciertos al emplear el clasificador descrito y la simplicidad se mide como el número de elementos del vector de pesos que son menores que 0.2, ya que estos pesos no son empleados por el clasificador, o lo que es lo mismo, son sustituidos por cero. Por lo tanto las calificaciones siguen las fórmulas:

$$Tasa\_acierto = 100 \cdot \frac{n^o aciertos}{n^o datos} \ , \ Tasa\_simplicidad = 100 \cdot \frac{n^o valoresdew < 0{,}2}{n^o deatributos}$$

$$Tasa\_agregada = \frac{1}{2} \cdot Tasa\_acierto + \frac{1}{2} \cdot Tasa\_simplicidad$$

Cabe destacar que todas las tasas están expresadas en porcentajes, por lo tanto cuanto más cercano sea el valor a 100 mejor es la calificación.
De esta forma a través del algoritmo que obtiene el vector de pesos para el conjunto de datos dado y el clasificador obtenemos un programa que clasifica de forma automática las nuevas instancias de datos que se introduzcan.

2. Introducción de la práctica

En esta práctica he analizado el comportamiento de los algoritmos KNN con k=1, el algoritmo greedy Relief y una implementación del algoritmo de búsqueda local para el problema de obtención de un vector de pesos para clasificar un conjunto de datos. Así mismo he realizado la implementación del algoritmo KNN con k variable para poder estudiar si varían los resultados al aumentar el valor de K o por contra obtenemos demasiado ajuste.
Para empezar al leer los ficheros de datos dados para el problema me he dado cuenta de que tenemos tuplas repetidas, cosa que he tenido en cuenta para no usarlas en la clasificación, ya que siempre obtendríamos distancia 0 para dicha tupla. Para ello en vez de comprobar el índice dentro del vector he comprobado si las tuplas son iguales para no usarlas.
Así mismo he implementado diferentes distancias a parte de la euclídea para comprobar si los resultados son mejores o peores en función de la distancia para cada conjunto de datos.
Para terminar, antes de analizar los datos, se debe considerar que los datos han sido redondeados a 4 decimales para no obtener tablas excesivamente largas. Si se desea obtener los datos completos se puede ejecutar el programa como se describe en la sección 7.

3. Descripción común a todos los algoritmos

Los algoritmos empleados han sido el KNN, el algoritmo greedy Relief y la metaheurística de búsqueda local.
Estos algoritmos comparten ciertos métodos y operadores que pasaré a explicar en esta sección.
Para empezar se debe destacar que la representación escogida para las soluciones es un vector de números reales, es decir, si n es el número de características:

$$w \in \mathbb{R}^n \ t.q. \ \forall i \ con \ 0 \leq i < n \ se \ tiene \ w_i \in [0,1]$$

O lo que es lo mismo, un vector de tamaño n con todas las posiciones rellenas con números del intervalo [0,1].
A estos números me referiré como pesos asociados a las características, ya que lo que nos indican es el grado de importancia de dicha característica a la hora de clasificar los datos, siendo 1 el máximo de relevancia y 0 el mínimo.
Así mismo cabe destacar que nuestra intención en este problema es obtener una buena calificación de dicho vector de pesos. Esto lo medimos mediante las tasas de acierto y simplicidad que se definen como:

$$Tasa\_acierto = 100 \cdot \frac{n^o aciertos}{n^o datos} \ , \ Tasa\_simplicidad = 100 \cdot \frac{n^o valoresdew < 0{,}2}{n^o deatributos}$$

$$Tasa\_agregada = \frac{1}{2} \cdot Tasa\_acierto + \frac{1}{2} \cdot Tasa\_simplicidad$$

La tasa de aciertos lo que nos mide es en un porcentaje cuántas instancias hemos clasificado correctamente mediante el algoritmo KNN usando el vector de pesos w.
La tasa de simplicidad nos mide cuántos de los valores que tiene el vector de pesos son menores que 0.2. Esto se hace ya que como imposición del problema tenemos que si alguna de los pesos es menor que 0.2 no debemos usarlo, o lo que es lo mismo, debemos sustituirlo por un 0 en la función de la distancia que luego describiré. Midiendo esto obtenemos un dato de cuanto sobreajuste ha tenido nuestro algoritmo a la hora de obtener el vector de pesos. Cuantas menos características necesitemos para discernir la clase a la que pertenece una instancia de los datos, más simple será clasificar dicha instancia. Se expresa en porcentaje indicando 0 como ninguna simplicidad y 100 como la máxima simplicidad.

De esta forma combinando ambas tasas obtenemos la tasa agregada que nos hace la media entre ambas tasas, de forma que le asignamos la misma importancia a acertar en la clasificación de las instancias y a la simplicidad en la solución. Cabe destacar que es imposible obtener una tasa de un 100 % a no ser que los datos se compongan únicamente de un punto ya que ello implicaría que la simplicidad ha de ser un 100 % (todos las posiciones del vector menores que 0.2) y por tanto la distancia sería 0 en todos los casos. De esta forma aspiraremos a una calificación lo mas alta posible pero teniendo en cuenta las restricciones de la función objetivo construida.
Las funciones y operadores de uso común los he agrupado en un fichero llamado auxiliar.py. Este fichero contiene las funciones de lectura de datos, distancias, una función que devuelve el elemento más común de una lista, la norma euclídea y una función para dividir los datos en el número de particiones que queramos manteniendo el porcentaje de elementos de cada clase que había en el conjunto de datos original.

3.1. Función de lectura de datos

La función de lectura de datos recibe la ruta del fichero arff y lee el contenido del mismo dando como resultado una lista de listas en la que cada una de ellas es una tupla o instancia de los datos.
El pseudocódigo de la función es:

Algorithm 1 lecturaDatos(nombre_fich)

```
data ← [ ]
for linea de nombre_fich do
  if se ha leído @data then
    data ← [data,linea]
  end if
end for
for tupla en data do
  for atributo en tupla do
    Normalizar.
  end for
end for
return data
```

Para esta implementación en concreto nos apoyamos en que Python tiene polimorfismo para todos los tipos de datos sin necesidad de declarar las variables, de forma que no nos importa que los datos sean numéricos o de tipo string.

3.2. Funciones de distancia

Las funciones de distancia siguen todas el mismo esquema de código, cambiando únicamente la fórmula empleada en cada caso para computar la distancia. Voy a describir las 3 distancias que he implementado teniendo esto en cuenta.
Se debe tener en cuenta que e1 y e2 son ambos dos tuplas del conjunto de datos de las que vamos a obtener la distancia y w es el vector de pesos que toma parte en el cómputo.
El resto de funciones de distancia siguen el mismo esquema pero cambiando la fórmula para obtener el valor de la misma.

Algorithm 2 distanciaEuclidea(e1,e2,w)

if longitud(e1)!=longitud(e2) **then**
  No se puede hallar la distancia.
else
  distancia = $\sqrt{\sum_{i=1}^{longitud(e1)} w_i \cdot (e1_i - e2_i)^2}$
end if

3.3. Función MasComun

Esta función lo que hace es devolvernos el elemento que más se repite dentro de un vector, esto es utilizado en el algoritmo KNN para obtener la clase más común entre los k elementos con distancia mínima al dado.

Algorithm 3 masComun(lista)

vector_repeticiones ← [ ]
for elemento en lista **do**
  Contar el número de veces que aparece e introducirlo en el vector de repeticiones.
end for
Obtener el elemento con mayor número de apariciones.

3.4. Función de norma euclídea

Esta función toma una lista de elementos y devuelve la norma euclídea asociada al vector que representa la lista.

Algorithm 4 normaEuclidea(e)

norma ← 0
for ei en e **do**
  norma ← norma+ei^2
end for
norma ← $\sqrt{norma}$
return norma

3.5. Función de división de datos

Esta función divide el conjunto de datos inicial en n particiones todas ellas respetando el porcentaje de ocurrencias de cada clase que tenía el conjunto de datos inicial.

4. KNN

5. Relief

6. Búsqueda Local

7. Ejecución del programa y explicación

Algorithm 5 divideDatosFCV(data,n)

```
particiones ← [ ]
tam_particion ← longitud(data) / n
clases ← Posibles clases del conjunto data.
proporciones_clases ← [ ]
for clase en clases do
  proporciones_clases ← [proporciones_clases, nueva_proporcion]
end for
for i en 0..n do
  particion ← [ ]
  for j en 0..numero_clases do
    numero_elementos_clase = proporciones_clases[j]*tam_particion
    for k en 0..longitud(data) do
      Introducir en particion el número de elementos de clase calculado para cada clase.
    end for
  end for
end for
Si han sobrado datos sin colocar los ponemos en la última partición.
return particiones
```

	Ozone				Parkinsons				Spectf-Heart			
	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)
Partición 1	64.0625	0.0000	32.0313	0.4475	68.4211	0.0000	34.2105	0.0565	76.4706	0.0000	38.2353	0.2591
Partición 2	68.7500	0.0000	34.3750	0.4626	68.4211	0.0000	34.2105	0.0549	80.8824	0.0000	40.4412	0.3027
Partición 3	70.3125	0.0000	35.1563	0.4515	81.5789	0.0000	40.7895	0.0549	73.5294	0.0000	36.7647	0.3203
Partición 4	71.8750	0.0000	35.9375	0.4857	57.8947	0.0000	28.9474	0.0551	61.7647	0.0000	30.8824	0.3036
Partición 5	70.3125	0.0000	35.1563	0.4569	67.4419	0.0000	33.7209	0.0608	75.3247	0.0000	37.6623	0.2703
Media	69.0625	0.0000	34.5313	0.4608	68.7515	0.0000	34.3758	0.0565	73.5943	0.0000	36.7972	0.2912

Cuadro 1: Resultados 1NN

	Ozone				Parkinsons				Spectf-Heart			
	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)
Partición 1	60.9375	0.0000	30.4688	0.4374	71.0526	0.0000	35.5263	0.0576	73.5294	0.0000	36.7647	0.2622
Partición 2	78.1250	0.0000	39.0625	0.4369	71.0526	0.0000	35.5263	0.0551	77.9412	0.0000	38.9706	0.3021
Partición 3	68.7500	0.0000	34.3750	0.4377	84.2105	0.0000	42.1053	0.0551	63.2353	0.0000	31.6176	0.3265
Partición 4	52.5000	0.0000	31.2500	0.4363	63.1579	0.0000	31.5789	0.0547	67.6451	0.0000	33.8235	0.3091
Partición 5	71.8750	0.0000	35.9375	0.4384	69.7674	0.0000	34.8837	0.0606	70.1299	0.0000	35.0649	0.2806
Media	68.4375	0.0000	34.2188	0.4374	71.8482	0.0000	35.9241	0.0566	70.4966	0.0000	35.2483	0.2961

Cuadro 2: Resultados 3NN

	Ozone				Parkinsons				Spectf-Heart			
	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)
Partición 1	68.7500	0.0000	34.3750	0.4497	78.9474	0.0000	39.4737	0.0572	66.1765	0.0000	33.0882	0.2653
Partición 2	75.0000	0.0000	37.5000	0.4471	76.3158	0.0000	38.1579	0.0562	75.0000	0.0000	37.5000	0.3069
Partición 3	67.1875	0.0000	33.5938	0.4397	86.8421	0.0000	43.4211	0.0553	63.2353	0.0000	31.6176	0.3257
Partición 4	65.6250	0.0000	32.8125	0.4419	57.8947	0.0000	28.9474	0.0555	60.2941	0.0000	30.1471	0.3130
Partición 5	68.7500	0.0000	34.3750	0.4435	69.7674	0.0000	34.8837	0.0605	72.7273	0.00000	36.3636	0.2775
Media	69.0625	0.0000	34.5313	0.4444	73.9535	0.0000	36.9767	0.0570	67.4866	0.0000	33.7433	0.2977

Cuadro 3: Resultados 5NN

	Ozone				Parkinsons				Spectf-Heart			
	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)
Partición 1	64.0625	94.5205	79.2915	0.9635	68.4211	86.9565	77.6888	0.1285	72.0588	55.5556	63.8072	0.4622
Partición 2	67.1875	94.5205	80.8540	1.0163	68.4211	86.9565	77.6888	0.1286	77.9412	55.5556	66.7484	0.6366
Partición 3	71.8750	94.5205	83.1978	1.0686	81.5789	86.9565	84.2677	0.1283	75.0000	80.0000	77.5000	0.7115
Partición 4	68.7500	94.5205	81.6353	1.0099	57.8947	86.9565	72.4256	0.1298	67.6471	55.5556	61.6013	0.6471
Partición 5	71.8750	76.7123	74.2937	1.0569	67.4419	95.6522	81.5470	0.1207	68.8312	71.1111	69.9711	0.3975
Media	68.7500	90.9589	79.8545	1.0230	68.7515	88.6957	78.7236	0.1272	72.2956	63.5556	67.9256	0.5710

Cuadro 4: Resultados Relief con K=1

	Ozone				Parkinsons				Spectf-Heart			
	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)
Partición 1	57.8125	94.5205	76.1665	0.9339	71.0526	86.9565	79.0046	0.1293	66.1765	55.5556	60.8660	0.4724
Partición 2	79.6875	94.5205	87.1040	0.9465	71.0526	86.9565	79.0046	0.1318	73.5294	55.5556	64.5425	0.6436
Partición 3	71.8750	94.5205	83.1978	0.9264	84.2105	86.9565	85.5835	0.1293	67.6471	80.0000	73.8235	0.7233
Partición 4	71.8750	94.5205	83.1978	0.9448	63.1579	86.9565	75.0572	0.1369	67.6471	55.5556	61.6013	0.6602
Partición 5	68.7500	76.7123	72.7312	0.9411	69.7674	95.6522	82.7098	0.1239	70.1299	71.1111	70.6205	0.4006
Media	70.0000	90.9589	80.4795	0.9385	71.8482	88.6957	80.2719	0.1302	66.5852	39.5556	53.0704	98.2743

Cuadro 5: Resultados Relief con K=3

	Ozone				Parkinsons				Spectf-Heart			
	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)
Partición 1	70.3125	94.5205	82.4165	0.9506	78.9474	86.9565	82.9519	0.1300	72.0588	55.5556	63.8072	0.4664
Partición 2	73.4375	94.5205	83.9790	0.9557	76.3158	86.9565	81.6362	0.1305	75.0000	55.5556	65.2778	0.6453
Partición 3	67.1875	94.5205	80.8540	0.9430	86.8421	86.9565	86.8993	0.1305	66.1765	80.0000	73.0882	0.7261
Partición 4	65.6250	94.5205	80.0728	0.9523	57.8947	86.9565	72.4256	0.1314	61.7647	55.5556	58.6601	0.6570
Partición 5	67.1875	76.7123	71.9499	0.9522	69.7674	95.6522	82.7098	0.1222	67.5325	71.1111	69.3218	0.4047
Media	68.7500	90.9589	79.8545	0.9508	73.9535	88.6957	81.3246	0.1289	68.5065	63.5556	66.0310	0.5799

Cuadro 6: Resultados Relief con K=5

	Ozone				Parkinsons				Spectf-Heart			
	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)
Partición 1	70.3125	34.2566	52.2795	191.1525	68.4211	30.4348	49.4279	10.3490	75.0000	28.8889	51.9444	62.9635
Partición 2	68.7500	41.0959	54.9229	306.5837	68.4211	21.7391	45.0800	5.3823	77.9412	51.1111	64.5261	114.7232
Partición 3	73.4375	46.5753	60.0064	238.3245	78.9474	26.0870	52.5172	8.5750	69.1176	55.5556	62.3366	221.4198
Partición 4	64.0625	36.9863	50.5244	284.2205	60.5263	30.4348	45.4805	7.8226	69.1176	31.1111	50.1144	53.3077
Partición 5	73.4375	45.2055	59.3215	335.4325	65.1163	17.3913	41.2538	4.0212	66.2338	40.0000	53.1169	77.1713
Media	70.0000	40.8219	55.4110	271.1427	68.2864	25.2174	46.7519	7.2300	71.4820	41.3333	56.4077	105.9171

Cuadro 7: Resultados Búsqueda Local con K=1

	Ozone				Parkinsons				Spectf-Heart			
	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)
Partición 1	60.9375	56.1644	58.5509	294.3525	73.6842	34.7826	54.2334	14.6002	60.2941	31.1111	45.7026	71.3179
Partición 2	76.5625	42.4658	59.5141	223.4619	68.4211	34.7826	51.6018	24.7653	76.4706	48.8889	62.6797	162.4381
Partición 3	76.5625	38.3562	57.4593	192.8862	81.5789	30.4348	56.0069	10.1302	64.7059	24.4444	44.5752	106.4757
Partición 4	62.5000	47.9452	55.2226	297.2501	65.7895	30.4348	48.1121	7.0666	69.1176	46.6667	57.8922	75.3324
Partición 5	73.4375	49.3151	61.3763	345.1076	74.4186	39.1204	56.7745	13.2932	62.3377	46.6667	54.5022	75.8076
Media	70.0000	46.8493	58.4247	270.6116	72.7785	33.9130	53.3458	13.9711	66.5852	39.5556	53.0704	98.2743

Cuadro 8: Resultados Búsqueda Local con K=3

	Ozone				Parkinsons				Spectf-Heart			
	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)	%_clas	%_red	Agr.	T (seg)
Partición 1	64.0625	43.8356	53.9491	241.0353	81.5789	34.7826	58.1808	11.7544	79.4118	24.4444	51.9281	42.3700
Partición 2	76.5625	46.5753	61.5689	251.2154	73.6842	26.0870	49.8856	9.0216	67.6471	62.2222	64.9346	135.2455
Partición 3	62.5000	53.4247	57.9623	257.2673	86.8421	43.4783	65.1602	13.4645	66.1765	37.7778	51.9771	135.2988
Partición 4	65.6250	52.0548	58.8399	207.7297	55.2632	52.1739	53.7285	16.7920	64.7059	46.6667	55.6863	124.9592
Partición 5	70.3125	53.4247	61.8686	316.8861	72.0930	21.7391	46.9161	9.1895	71.4286	17.7778	44.6032	44.4294
Media	67.8125	49.8630	58.8378	254.8268	73.8923	35.6522	54.7722	12.0444	69.8739	37.7778	53.8259	96.4606

Cuadro 9: Resultados Búsqueda Local con K=5