PEC 2

Presentación

Segunda actividad de evaluación continua del curso. En esta PEC se pretende conocer y desarrollar distintas técnicas de machine learning.

Competencias

Competencias de grado

- Capacidad de utilizar los fundamentos matemáticos, estadísticos y físicos y comprender los sistemas TIC.

- Capacidad para analizar un problema en el nivel de abstracción adecuado a cada situación y aplicar las habilidades y conocimientos adquiridos para abordarlo y resolverlo.

- Capacidad para conocer las tecnologías de comunicaciones actuales y emergentes y saberlas aplicar convenientemente, para diseñar y desarrollar soluciones basadas en sistemas y tecnologías de la información.

- Capacidad para proponer y evaluar diferentes alternativas tecnológicas y resolver un problema concreto.

Competencias específicas

- Capacidad para utilizar la tecnología de aprendizaje automático más adecuada para resolver un determinado problema.

- Capacidad para evaluar el rendimiento de los diferentes algoritmos de resolución de problemas mediante técnicas de validación cruzada.

Objetivos

El objetivo de esta prueba de evaluación es clasificar los objetos astronómicos del conjunto de datos proporcionado en función de su clase y evaluar el rendimiento de los modelos utilizados.

Descripción de la PEC

Se proporciona un conjunto de datos en formato CSV que contiene información sobre objetos astronómicos observados por el Sloan Digital Sky Survey (SDSS). Las características incluyen medidas fotométricas en diferentes filtros, coordenadas astronómicas, información de observación, etc. La variable objetivo es la clase de objeto, que puede ser galaxia, estrella u objeto quasar

Ejercicio 1

a) Realiza el preprocesamiento de los datos si es necesario, justificando cada paso y mostrando el dataset tratado.

El preprocesamiento de los datos se hará sobre el dataset TRAIN y TEST que nos ha proporcionado el equipo docente. Así entonces, tenemos un conjunto de datos de 13 objetos con 18 características cada uno, siendo una de ellas la clase que representa el objeto. Tenemos 3 clases distintas (GALAXY, STAR o QSO), y tenemos 4 objetos de cada una excepto de STAR, que tenemos 5. Los datos se muestran a continuación (con un índice para identificar los objetos):

Colum	obj_ID	alpha	delta	u	g	r	i	z	run_ID	rerun_I	cam_co	field_ID	spec_o	class	redshif	plate	MJD	fiber_ID
0	1,24E+32	1,87E+14	5,87E+14	2614481	2456541	2097581	1958867	1923176	2826	301	6	300	7,85E+32	GALAXY	6822903	6968	56443	323
1	1,24E+32	5,43E+13	7,7E+14	2323521	2204661	2021183	192951	1869015	4849	301	5	817	2,97E+32	GALAXY	4948072	2639	54465	337
2	1,24E+32	1,98E+14	1,92E+14	1913823	180857	1764759	1737495	1715765	5314	301	3	136	2,95E+32	GALAXY	9203831	2617	54502	114
3	1,24E+31	3,54E+14	6,33E+14	2013128	1892085	1848558	1827021	1814316	4188	301	5	68	4,74E+32	STAR	-7793586	4212	55447	802
4	1,24E+32	3,57E+14	-8,4E+14	1773818	1691551	1662435	1652587	1651668	4263	301	2	94	9,87E+32	STAR	-3497355	8767	57310	6
5	1,24E+32	8,81E+14	2,82E+14	2220962	2024953	1870976	1727218	1645631	7712	301	4	388	8,84E+32	STAR	78,25889	7855	57011	908
6	1,24E+32	1,2E+14	4,12E+14	2283221	2174318	2119896	2083009	205248	2076	301	6	113	4,14E+31	STAR	32,99982	3680	55210	74
7	1,24E+32	1,92E+14	4,19E+14	2162058	210798	206557	2042108	2003778	3840	301	3	208	9,44E+32	QSO	8466634	8386	57518	511
8	1,24E+31	1,9E+14	4,63E+13	1978255	1861807	1851826	1851242	1830961	3698	301	1	194	7,47E+32	QSO	2766249	6635	56370	602
9	1,24E+32	1,29E+14	1,42E+13	2334927	2156899	2167272	2174917	2159274	5194	301	6	108	5,07E+32	QSO	2263579	4500	55543	304
10	1,24E+32	3,52E+14	-9,7E+14	2003712	1904514	1877032	1865098	187190	1729	301	4	28	3,54E+32	STAR	1687093	3145	54801	279
11	1,24E+32	1,8E+14	5,12E+14	1959604	1754836	1651074	1609856	1572806	2830	301	1	397	9,93E+30	GALAXY	1189402	882	52370	289
12	1,24E+31	1,82E+14	5,1E+14	1970178	192986	1896274	1866815	1855648	2964	301	6	309	1,09E+32	QSO	1655222	969	52442	567

Se decide realizar una selección de características univariante para evaluar la función de bonanza de las características. Posteriormente, analizaremos si alguna de los atributos puede ser omitido. En cualquier caso, una vez decididas las características, normalizaremos los datos para su posterior tratamiento.

Para el cálculo de la función de bonanza, haremos un cálculo de la distancia en el centro de masas, de forma que a cada punto se le asignará la clase asociada al centro de masas más cercano. Ejemplificaremos el cálculo paso a paso:

1. Calculamos la media de las clases según la característica, calculada como la media de los valores de la característica cuando pertenece a la clase.

Column1	obj_ID	alpha	delta	u	g	r	i	z	run_ID	rerun_ID	cam_col	field_ID	spec_obj	class	redshift	plate	MJD	fiber_ID
	1,23766E+32	1,55E+14	5,15E+14	2202857	1649224	1883649	1374792	1770191	3954,75	301	3,75	412,5	3,47E+32	GALAXY	5541052	3276,5	54445	265,75
	1,01488E+32	4,13E+14	-9,6E+13	2058968	1937484	1875779	1830987	1100811	3993,6	301	4,2	138,2	5,48E+32	STAR	-1920747	5531,8	55955,8	413,8
	6,81E+31	1,73E+14	2,48E+14	2,11E+06	1,11E+06	1,53E+06	1,98E+06	1,96E+06	3,92E+03	3,01E+02	4,00E+00	2,05E+02	5,77E+32	QSO	3,79E+06	5,12E+03	5,55E+04	4,96E+02

Ahora, calculamos la distancia al centro de masas. Para ello, calculamos las distancias absolutas entre los centros de las clases y la característica, y clasificamos el objeto en el mínimo de distancia. Para el objeto 0:

$$alpha: \begin{cases} d(A_0, \overline{GALAXY}) = |186564000000000 - 154547000000000| = 3{,}2e13 \\ d(A_0, \overline{STAR}) = |186564000000000 - 412730000000000| = 2{,}26e14 \\ d(A_0, \overline{QSO}) = |186564000000000 - 173000000000000| = 1{,}32e13 \end{cases}$$

$$\min\left(d(A_0, \overline{GALAXY}), d(A_0, \overline{STAR}), d(A_0, \overline{QSO})\right) = d(A_0, \overline{QSO}) \in QSO$$

Así entonces, según la característica *Alpha*, el objeto 0 pertenece a la clase QSO (en este caso, el objeto realmente pertenece a la clase GALAXY). A continuación dejo una tabla con la clasificación de todos los objetos según la distancia al centro de masas(0 es GALAXY, 1 es STAR y 2 es QSO):

Column1	obj_ID	alpha	delta	u	g	r	i	z	run_ID	rerun_ID	cam_col	field_ID	spec_obj_class	redshift	plate	MJD	fiber_ID
0	0	2	0	0	1	0	2	2	2	0	1	2	2	0	0	1	1	0
1	0	0	0	0	1	0	0	2	1	0	1	0	0	0	0	0	0	0
2	0	2	2	1	2	1	1	0	1	0	0	1	0	0	0	0	0	0
3	2	1	0	1	1	1	1	0	1	0	1	1	1	1	1	2	2	2
4	0	1	1	1	0	2	1	0	1	0	0	1	2	1	1	1	1	0
5	0	1	2	0	1	1	1	0	1	0	2	0	2	1	1	1	1	2
6	0	0	0	0	1	0	2	1	2	0	1	1	0	1	1	0	2	0
7	0	2	0	0	2	2	2	2	2	0	0	2	2	2	0	1	1	2
8	2	2	1	1	1	1	1	0	2	0	0	2	2	2	2	1	1	2
9	0	0	1	0	1	0	2	2	1	0	1	1	1	2	2	2	2	0
10	0	1	1	1	1	1	1	1	2	0	2	1	0	1	2	0	0	0
11	0	2	0	1	0	2	1	0	2	0	0	0	0	0	2	0	0	0
12	2	2	0	1	2	0	1	0	2	0	1	0	0	2	2	0	0	2

Su función de bondad asociada es calculada de la siguiente forma: se suman los aciertos y se divide por el total. En el caso de la característica obj_ID, tiene 4 aciertos, así que su función de bondad es $\frac{6}{13} = 0{,}46 = 46\%$.

Se acompaña con todas las funciones de bonanza:

Característica	obj_ID	alpha	delta	u	g	r	i	z	run_ID	rerun_ID	cam_col	field_ID	spec_obj_	redshift	plate	MJD	fiber_ID
Función de bonanza	46%	62%	38%	38%	54%	46%	54%	46%	46%	31%	31%	62%	46%	77%	46%	46%	54%

Como puede observarse, hay bastantes características que o bien nos dan valores aleatorios, o son contraproducentes para el análisis objetivo. Proponemos tomar todos los atributos cuyas funciones de bonanza sean mayor de 50%. Así entonces, nos quedaremos con *Alpha, g,i, field_ID, redshift* y *fiber_ID.*

Column1	alpha	g	i	field_ID	redshift	fiber_ID	class
0	1,86564E+14	2456541	1958867	300	6822903	323	GALAXY
1	5,42846E+13	2204661	192951	817	4948072	337	GALAXY
2	1,978E+14	180857	1737495	136	9203831	114	GALAXY
3	3,5369E+14	1892085	1827021	68	-7793586	802	STAR
4	3,56735E+14	1691551	1652587	94	-3497355	6	STAR
5	8,8136E+14	2024953	1727218	388	78,25889	908	STAR
6	1,20203E+14	2174318	2083009	113	32,99982	74	STAR
7	1,91727E+14	210798	2042108	208	8466634	511	QSO
8	1,9041E+14	1861807	1851242	194	2766249	602	QSO
9	1,29339E+14	2156899	2174917	108	2263579	304	QSO
10	3,51662E+14	1904514	1865098	28	1687093	279	STAR
11	1,79538E+14	1754836	1609856	397	1189402	289	GALAXY
12	1,82036E+14	192986	1866815	309	1655222	567	QSO

Finalmente, normalizaremos los datos. La decisión del tipo de normalización a aplicar ha sido tomada en base a los tipos de datos que tenemos. Al ser datos numéricos, utilizaremos la estandarización (aunque podría ser válido usar ranging e incluso útil para según que algoritmos queramos aplicar). Recordemos que la estandarización se lleva a cabo mediante esta fórmula:

$$v' = \frac{v - media\left(A(x)\right)}{\sigma\left(A(x)\right)}$$

Calcularemos la media y la desviación de cada característica. La desviación es representada por:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{N}}$$

	alpha	g	i	field_ID	redshift	fiber_ID
media	2,59642E+14	1592831,231	1737630	243,0769	2131704	393,5385
desviación	2,00552E+14	790406,4478	473586,5	202,1182	4481619	261,5815

Con la media y la desviación calculadas, se dan los valores de las características normalizadas:

Column1	alpha	g	i	field_ID	redshift	fiber_ID	class
0	-0,3643844	1,0927413	0,467153	0,281633	1,046764	-0,26966	GALAXY
1	-1,02395959	0,77406981	-3,26166	2,839542	0,628426	-0,21614	GALAXY
2	-0,30835914	-1,78639007	-0,00028	-0,52977	1,578029	-1,06865	GALAXY
3	0,468943882	0,378607449	0,188754	-0,86621	-2,21467	1,561508	STAR
4	0,484126945	0,124897475	-0,17957	-0,73757	-1,25603	-1,48152	STAR
5	3,100026612	0,546708305	-0,02198	0,717021	-0,47564	1,966735	STAR
6	-0,69527545	0,735680701	0,729285	-0,64357	-0,47565	-1,22156	STAR
7	-0,3386405	-1,74850956	0,64292	-0,17355	1,413536	0,449044	QSO
8	-0,34520736	0,340300576	0,239898	-0,24281	0,141588	0,796928	QSO
9	-0,64972127	0,713642672	0,923353	-0,66831	0,029426	-0,3423	QSO
10	0,458831675	0,394332271	0,269156	-1,06411	-0,09921	-0,43787	STAR
11	-0,39941926	0,204963876	-0,2698	0,76155	-0,21026	-0,39964	GALAXY
12	-0,38696213	-1,7710448	0,272781	0,326161	-0,10632	0,663126	QSO

Finalmente, separaremos los conjuntos de TRAIN y TEST nuevamente para su prueba con los algoritmos.

TRAIN:

Column1	alpha	g	i	field_ID	redshift	fiber_ID	class
0	-0,3643844	1,0927413	0,467153	0,2816326	1,0467643	-0,269662	GALAXY
1	-1,02395959	0,77406981	-3,26166	2,8395419	0,6284264	-0,216141	GALAXY
2	-0,30835914	-1,78639007	-0,00028	-0,529774	1,5780294	-1,068648	GALAXY
3	0,468943882	0,378607449	0,188754	-0,866211	-2,214666	1,5615079	STAR
4	0,484126945	0,124897475	-0,17957	-0,737573	-1,256033	-1,481521	STAR
5	3,100026612	0,546708305	-0,02198	0,7170214	-0,475637	1,9667354	STAR
6	-0,69527545	0,735680701	0,729285	-0,643569	-0,475648	-1,221564	STAR
7	-0,3386405	-1,74850956	0,64292	-0,173547	1,413536	0,4490438	QSO
8	-0,34520736	0,340300576	0,239898	-0,242813	0,1415883	0,7969278	QSO
9	-0,64972127	0,713642672	0,923353	-0,668307	0,0294257	-0,342297	QSO

TEST:

Column1	alpha	g	i	field_ID	redshift	fiber_ID	class
10	0,458831675	0,394332271	0,269156	-1,064115	-0,099208	-0,437869	STAR
11	-0,39941926	0,204963876	-0,2698	0,7615498	-0,210259	-0,39964	GALAXY
12	-0,38696213	-1,7710448	0,272781	0,326161	-0,106319	0,6631263	QSO

b) Utiliza un algoritmo de clasificación de tu elección para categorizar los objetos astronómicos en las tres clases mencionadas. Calcula y presenta la exactitud del modelo.

Utilizaremos el método del *k-nearest neighbour*. El método de clasificación determina la clase del objeto a partir de los k-objetos más próximos. En nuestro caso haremos una k=3. Entonces, tenemos un conjunto de ejemplos TRAIN, y queremos clasificar los objetos del conjunto TEST. Ejemplificaremos el cálculo para el primer objeto. Primeramente, calcularemos las distancias entre todos los puntos de TRAIN y el objeto 1. Para ello, utilizaremos la distancia euclidiana, definida de esta manera:

$$d(x_j, x_k) = \left(\sum_{i=0}^{M} \left(A_i(x_j) - A_i(x_k)\right)^2\right)^{\frac{1}{2}}$$

Las distancias entre los objetos quedan tal como se muestran a continuación:

	d(10,x)	d(11,x)	d(12,x)
0	2,087401	1,777582417	3,231291
1	5,534138	3,836701734	5,196701
2	2,985508	3,059327694	2,578696
3	2,918673	3,392319471	3,469093
4	1,67591	2,303623872	3,407477
5	4,022457	4,253993998	4,427925
6	1,610262	2,021808228	3,349024
7	3,039005	2,980636138	1,656908
8	1,705138	1,686820407	2,205379
9	1,393258	1,962204112	2,946901

Una vez calculadas, procedemos a escoger la clase de los objetos según su cercanía con los tres objetos más próximos. Para el objeto 10 (el primer objeto de TEST), los objetos más cercanos son el 9, el 6 y el 4, que pertenecen a QSO, STAR y STAR, respectivamente. Como la clase mayoritaria de los objetos cercanos es STAR, el objeto 10 pertenecerá a STAR (y realmente pertenece a STAR).

El objeto 11 tiene como objetos más cercanos el 1, el 8 y el 9, que pertenecen a GALAXY, QSO y QSO. Así entonces, el objeto 11 pertenece a QSO (aunque realmente pertenece a GALAXY).

El objeto 12 tiene como objetos más cercanos el 7, el 8 y el 2, que pertenecen a QSO, QSO y GALAXY. Así entonces, el objeto 12 pertenece a QSO (y realmente pertenece a QSO).

La exactitud del modelo se puede calcular así:

$$Exactitud = \frac{Predicciones\ correctas}{Predicciones\ totales} = \frac{2}{3} = 0{,}67$$

c). ¿Qué métricas son necesarias para poder tener una idea precisa del funcionamiento del modelo de clasificación?

Podríamos considerar métricas como la precisión, el fall-out o el recall. Comentar que los datasets que hemos clasificado son muy pequeños y para métricas así dan poca información, pero las definiremos y calcularemos igualmente.

- Precisión: La precisión para cada clase mide la proporción de verdaderos positivos sobre el total de predicciones positivas. Esto nos puede indicar si el modelo es más preciso para unos objetos u otros.

$$Precision_{STAR} = \frac{TP}{TP + FP} = \frac{1}{1 + 0} = 1$$
$$Precision_{GALAXY} = \frac{0}{0} = indefinido\ (lo\ trataremos\ como\ 0)$$
$$Precision_{QSO} = \frac{1}{2} = 0{,}5$$

- Recall: mide la tasa de verdaderos positivos sobre el total de verdaderos positivos y falsos negativos. Esta métrica nos indica, para cada clase, como de bueno es el modelo para detectar verdaderos positivos.

$$Recall_{STAR} = \frac{TP}{TP + FN} = \frac{1}{1 + 0} = 1$$
$$Recall_{GALAXY} = \frac{0}{0 + 1} = 0$$
$$Recall_{QSO} = \frac{1}{1 + 0} = 1$$

- Fall-out: mide la tasa de falsos positivos. Se calcula como la división entre el número de falsos positivos y el número de ejemplos negativos. Nos puede dar información útil sobre como se clasifican incorrectamente los objetos.

$$Fall - out_{STAR} = \frac{FP}{FP + TN} = \frac{0}{0 + 2} = 0$$
$$Fall - out_{GALAXY} = \frac{0}{0 + 3} = 0$$
$$Fall - out_{QSO} = \frac{1}{1 + 1} = 0{,}5$$

Ejercicio 2

a) Construye un dendrograma utilizando un método de agrupamiento jerárquico. Muestra los pasos involucrados, incluyendo el dendrograma resultante y explica las decisiones tomadas.

Crearemos el agrupamiento jerárquico del dataset de entrenamiento TRAIN.

Column1	alpha	g	i	field_ID	redshift	fiber_ID	class
0	-0,3643844	1,0927413	0,467153	0,2816326	1,0467643	-0,269662	GALAXY
1	-1,02395959	0,77406981	-3,26166	2,8395419	0,6284264	-0,216141	GALAXY
2	-0,30835914	-1,78639007	-0,00028	-0,529774	1,5780294	-1,068648	GALAXY
3	0,468943882	0,378607449	0,188754	-0,866211	-2,214666	1,5615079	STAR
4	0,484126945	0,124897475	-0,17957	-0,737573	-1,256033	-1,481521	STAR
5	3,100026612	0,546708305	-0,02198	0,7170214	-0,475637	1,9667354	STAR
6	-0,69527545	0,735680701	0,729285	-0,643569	-0,475648	-1,221564	STAR
7	-0,3386405	-1,74850956	0,64292	-0,173547	1,413536	0,4490438	QSO
8	-0,34520736	0,340300576	0,239898	-0,242813	0,1415883	0,7969278	QSO
9	-0,64972127	0,713642672	0,923353	-0,668307	0,0294257	-0,342297	QSO

Para ello, seguiremos un algoritmo de generación de particiones y la selección de la mejor partición. Empecemos por los criterios de generación de particiones. Al tener atributos numéricos, construiremos nodos con un valor de corte de manera que se parten los objetos entre los que están por debajo y los que están por encima de ese valor. Primeramente, ordenaremos los valores de cada atributo forma creciente, y consideraremos los valores de corte entre cada valor sucesivo como $\frac{v_i+v_{i+1}}{2}$:

Column1	alpha	Dist. Corte	class
5	3,100026612	1,792076779	STAR
4	0,484126945	0,476535414	STAR
3	0,468943882	0,08029237	STAR
2	-0,30835914	-0,32349982	GALAXY
7	-0,3386405	-0,34192393	QSO
8	-0,34520736	-0,35479588	QSO
0	-0,3643844	-0,50705283	GALAXY
9	-0,64972127	-0,67249836	QSO
6	-0,69527545	-0,85961752	STAR
1	-1,02395959		GALAXY

Column1	g	Dist. Corte	class
0	1,0927413	0,933406	GALAXY
1	0,7740698	0,754875	GALAXY
6	0,7356807	0,724662	STAR
9	0,7136427	0,630175	QSO
5	0,5467083	0,462658	STAR
3	0,3786074	0,359454	STAR
8	0,3403006	0,232599	QSO
4	0,1248975	-0,81181	STAR
7	-1,74851	-1,76745	QSO
2	-1,78639		GALAXY

Column1	i	Dist. Corte	class
9	0,923352818	0,826319	QSO
6	0,7292848	0,686103	STAR
7	0,642920436	0,555037	QSO
0	0,467153192	0,353526	GALAXY
8	0,239897998	0,214326	QSO
3	0,18875423	0,094235	STAR
2	-0,00028408	-0,01113	GALAXY
5	-0,02198445	-0,10078	STAR
4	-0,17957128	-1,72062	STAR
1	-3,26166059		GALAXY

Column1	field_ID	Dist. Corte	class
1	2,839542	1,778282	GALAXY
5	0,717021	0,499327	STAR
0	0,281633	0,054043	GALAXY
7	-0,17355	-0,20818	QSO
8	-0,24281	-0,38629	QSO
2	-0,52977	-0,58667	GALAXY
6	-0,64357	-0,65594	STAR
9	-0,66831	-0,70294	QSO
4	-0,73757	-0,80189	STAR
3	-0,86621		STAR

Column1	redshift	Dist. Corte	class
2	1,578029	1,495782713	GALAXY
7	1,413536	1,230150138	QSO
0	1,046764	0,83759535	GALAXY
1	0,628426	0,385007347	GALAXY
8	0,141588	0,085506989	QSO
9	0,029426	-0,22310589	QSO
5	-0,47564	-0,47564253	STAR
6	-0,47565	-0,86584005	STAR
4	-1,25603	-1,73534937	STAR
3	-2,21467		STAR

Column1	fiber_ID	Dist. Corte	class
5	1,966735	1,764122	STAR
3	1,561508	1,179218	STAR
8	0,796928	0,622986	QSO
7	0,449044	0,116451	QSO
1	-0,21614	-0,2429	GALAXY
0	-0,26966	-0,30598	GALAXY
9	-0,3423	-0,70547	QSO
2	-1,06865	-1,14511	GALAXY
6	-1,22156	-1,35154	STAR
4	-1,48152		STAR

Una vez calculadas nuestras particiones, debemos seleccionar la mejor partición. Para cada valor, tenemos que separar los objetos en los dos grupos que crea el valor (en nuestro caso, más que ese valor, o menos que ese valor), y mirar qué objetos quedan bien clasificados cuando seleccionamos como clase la que aparece de manera más frecuente en el grupo. Ejemplificaremos un caso:

El valor que cogemos como distancia de corte es el valor v_{corte} =-0.22310589 del atributo redshift (distancia de corte entre el objeto 9 y 5). Creamos las dos particiones:

Para $v_i > v_{corte} = \{2,7,0,1,8,9\}$

Para $v_i < v_{corte} = \{5,6,4,3\}$

Los separamos en tabla para que se más gráfico:

Column1	redshift	Dist. Corte	class
2	1,578029	1,495782713	GALAXY
7	1,413536	1,230150138	QSO
0	1,046764	0,83759535	GALAXY
1	0,628426	0,385007347	GALAXY
8	0,141588	0,085506989	QSO
9	0,029426	-0,22310589	QSO

Column1	redshift	Dist. Corte	class
5	-0,47564	-0,47564253	STAR
6	-0,47565	-0,86584005	STAR
4	-1,25603	-1,73534937	STAR
3	-2,21467		STAR

Nuestro valor de medida, en este caso, será la suma de los valores correctos cogiendo el valor más común de cada grupo, dividido por el nombre total de objetos. Así entonces:

$$medida_{v_{corte}}^{redshift} = \frac{3+4}{10} = 0{,}7$$

A continuación, se presentan las medidas para cada distancia y atributo.

Column1	alpha	Dist. Corte	class	Medida
5	3,100026612	1,792076779	STAR	0,4
4	0,484126945	0,476535414	STAR	0,5
3	0,468943882	0,08029237	STAR	0,6
2	-0,30835914	-0,32349982	GALAXY	0,6
7	-0,3386405	-0,34192393	QSO	0,5
8	-0,34520736	-0,35479588	QSO	0,5
0	-0,3643844	-0,50705283	GALAXY	0,4
9	-0,64972127	-0,67249836	QSO	0,4
6	-0,69527545	-0,85961752	STAR	0,4
1	-1,02395959		GALAXY	

Column1	g	Dist. Corte	class	Medida
0	1,092741	0,933406	GALAXY	0,5
1	0,77407	0,754875	GALAXY	0,6
6	0,735681	0,724662	STAR	0,5
9	0,713643	0,630175	QSO	0,5
5	0,546708	0,462658	STAR	0,4
3	0,378607	0,359454	STAR	0,5
8	0,340301	0,232599	QSO	0,4
4	0,124897	-0,81181	STAR	0,5
7	-1,74851	-1,76745	QSO	0,5
2	-1,78639		GALAXY	

Column1	i	Dist. Corte	class	Medida
9	0,923353	0,826319	QSO	0,5
6	0,729285	0,686103	STAR	0,4
7	0,64292	0,555037	QSO	0,5
0	0,467153	0,353526	GALAXY	0,5
8	0,239898	0,214326	QSO	0,6
3	0,188754	0,094235	STAR	0,5
2	-0,00028	-0,01113	GALAXY	0,5
5	-0,02198	-0,10078	STAR	0,4
4	-0,17957	-1,72062	STAR	0,5
1	-3,26166		GALAXY	

Column1	field_ID	Dist. Corte	class	Medida
1	2,839542	1,778282	GALAXY	0,5
5	0,717021	0,499327	STAR	0,4
0	0,281633	0,054043	GALAXY	0,5
7	-0,17355	-0,20818	QSO	0,5
8	-0,24281	-0,38629	QSO	0,5
2	-0,52977	-0,58667	GALAXY	0,6
6	-0,64357	-0,65594	STAR	0,5
9	-0,66831	-0,70294	QSO	0,5
4	-0,73757	-0,80189	STAR	0,4
3	-0,86621		STAR	

Column1	redshift	Dist. Corte	class	Medida
2	1,578029	1,495782713	GALAXY	0,5
7	1,413536	1,230150138	QSO	0,5
0	1,046764	0,83759535	GALAXY	0,6
1	0,628426	0,385007347	GALAXY	0,7
8	0,141588	0,085506989	QSO	0,7
9	0,029426	-0,22310589	QSO	0,7
5	-0,47564	-0,47564253	STAR	0,6
6	-0,47565	-0,86584005	STAR	0,5
4	-1,25603	-1,73534937	STAR	0,4
3	-2,21467		STAR	

Column1	fiber_ID	Dist. Corte	class	medida
5	1,966735	1,764122	STAR	0,4
3	1,561508	1,179218	STAR	0,5
8	0,796928	0,622986	QSO	0,5
7	0,449044	0,116451	QSO	0,5
1	-0,21614	-0,2429	GALAXY	0,5
0	-0,26966	-0,30598	GALAXY	0,4
9	-0,3423	-0,70547	QSO	0,4
2	-1,06865	-1,14511	GALAXY	0,5
6	-1,22156	-1,35154	STAR	0,4
4	-1,48152		STAR	

Escogeremos la medida máxima. Tenemos tres medidas iguales a 0,7. Cogeremos la que nos da 0,7 con un conjunto más discriminado (es decir, un conjunto más pequeño entre los dos (preferimos un ratio 6-4 que uno 5-5)). Así entonces, el valor de corte para la primera iteración será $v_{corte} = -0{,}22310589$ del atributo *redshift*. Ahora, consideraremos las clases de los dos conjuntos discriminados, para ver como proseguiremos en las posteriores iteraciones. El primer conjunto solo incluye objetos de GALAXY o QSO, mientras que el

segundo conjunto solo incluye objetos de STAR (mostrados anteriormente en el ejemplo). Así entonces, tendremos una hoja que será marcada como la etiqueta STAR, y ahora tenemos que discriminar entre GALAXY y QSO. Así pues, el nuevo conjunto para la próxima iteración será el que incluya únicamente GALAXY y QSO.

Column1	alpha	g	i	field_ID	redshift	fiber_ID	class
8	-0,34520736	0,340300576	0,239898	-0,242813	0,1415883	0,7969278	QSO
7	-0,3386405	-1,74850956	0,64292	-0,173547	1,413536	0,4490438	QSO
1	-1,02395959	0,77406981	-3,26166	2,8395419	0,6284264	-0,216141	GALAXY
0	-0,3643844	1,0927413	0,467153	0,2816326	1,0467643	-0,269662	GALAXY
9	-0,64972127	0,713642672	0,923353	-0,668307	0,0294257	-0,342297	QSO
2	-0,30835914	-1,78639007	-0,00028	-0,529774	1,5780294	-1,068648	GALAXY

Volvemos a calcular las distancias de corte y las medidas:

Column1	alpha	Dist. Corte	class	Medida
2	-0,308359	-0,3235	GALAXY	0,666667
7	-0,338641	-0,341924	QSO	0,5
8	-0,345207	-0,354796	QSO	0,666667
0	-0,364384	-0,507053	GALAXY	0,5
9	-0,649721	-0,83684	QSO	0,666667
1	-1,02396		GALAXY	

Column1	g	Dist. Corte	class	Medida
0	1,092741	0,933405555	GALAXY	0,666667
1	0,77407	0,743856241	GALAXY	0,833333
9	0,713643	0,526971624	QSO	0,666667
8	0,340301	-0,70410449	QSO	0,5
7	-1,74851	-1,76744982	QSO	0,666667
2	-1,78639		GALAXY	

Column1	i	Dist. Corte	class	Medida
9	0,923353	0,783137	QSO	0,666667
7	0,64292	0,555037	QSO	0,833333
0	0,467153	0,353526	GALAXY	0,666667
8	0,239898	0,119807	QSO	0,833333
2	-0,00028	-1,63097	GALAXY	0,666667
1	-3,26166		GALAXY	

Column1	field_ID	Dist. Corte	class	Medida
1	2,839542	1,560587	GALAXY	0,666667
0	0,281633	0,054043	GALAXY	0,833333
7	-0,17355	-0,20818	QSO	0,666667
8	-0,24281	-0,38629	QSO	0,5
2	-0,52977	-0,59904	GALAXY	0,666667
9	-0,66831		QSO	

Column1	redshift	Dist. Corte	class	Medida
2	1,578029	1,495783	GALAXY	0,666667
7	1,413536	1,23015	QSO	0,5
0	1,046764	0,837595	GALAXY	0,666667
1	0,628426	0,385007	GALAXY	0,833333
8	0,141588	0,085507	QSO	0,666667
9	0,029426		QSO	

Column1	fiber_ID	Dist. Corte	class	Medida
8	0,796928	0,622986	QSO	0,666667
7	0,449044	0,116451	QSO	0,833333
1	-0,21614	-0,2429	GALAXY	0,666667
0	-0,26966	-0,30598	GALAXY	0,5
9	-0,3423	-0,70547	QSO	0,666667
2	-1,06865		GALAXY	

Aquí tenemos varios valores máximos en distintos atributos. Se puede escoger cualquiera. En nuestro caso, escogeremos el valor 0,116451 del atributo fiber_ID. Así pues, nos fijamos en los dos conjuntos que nos deja:

Column1	fiber_ID	class
8	0,796928	QSO
7	0,449044	QSO

Column1	fiber_ID	class
1	-0,21614	GALAXY
0	-0,26966	GALAXY
9	-0,3423	QSO
2	-1,06865	GALAXY

Podemos ver que en el primer conjunto se discrimina totalmente QSO, así que marcaremos con QSO la hoja resultante. En el otro conjunto tenemos GALAXY y QSO. La próxima iteración se centrará otra vez en estas clases, ahora con el conjunto de datos reducido a:

Column1	alpha	g	i	field_ID	redshift	fiber_ID	class
0	-0,3643844	1,0927413	0,467153	0,2816326	1,0467643	-0,269662	GALAXY
1	-1,02395959	0,77406981	-3,26166	2,8395419	0,6284264	-0,216141	GALAXY
2	-0,30835914	-1,78639007	-0,00028	-0,529774	1,5780294	-1,068648	GALAXY
9	-0,64972127	0,713642672	0,923353	-0,668307	0,0294257	-0,342297	QSO

Calculamos las distancias de corte y las medidas:

	alpha	Dist.Corte	class	Medida
2	-0,30835914	-0,33637177	GALAXY	0,75
0	-0,3643844	-0,50705283	GALAXY	0,75
9	-0,64972127	-0,83684043	QSO	0,75
1	-1,02395959		GALAXY	

Column1	g	Dist.Corte	class	Medida
0	1,092741	0,933406	GALAXY	0,75
1	0,77407	0,743856	GALAXY	0,75
9	0,713643	-0,53637	QSO	0,75
2	-1,78639		GALAXY	

Column1	i	Dist.Corte	class	Medida
9	0,923353	0,695253	QSO	1
0	0,467153	0,233435	GALAXY	0,75
2	-0,00028	-1,63097	GALAXY	0,75
1	-3,26166		GALAXY	

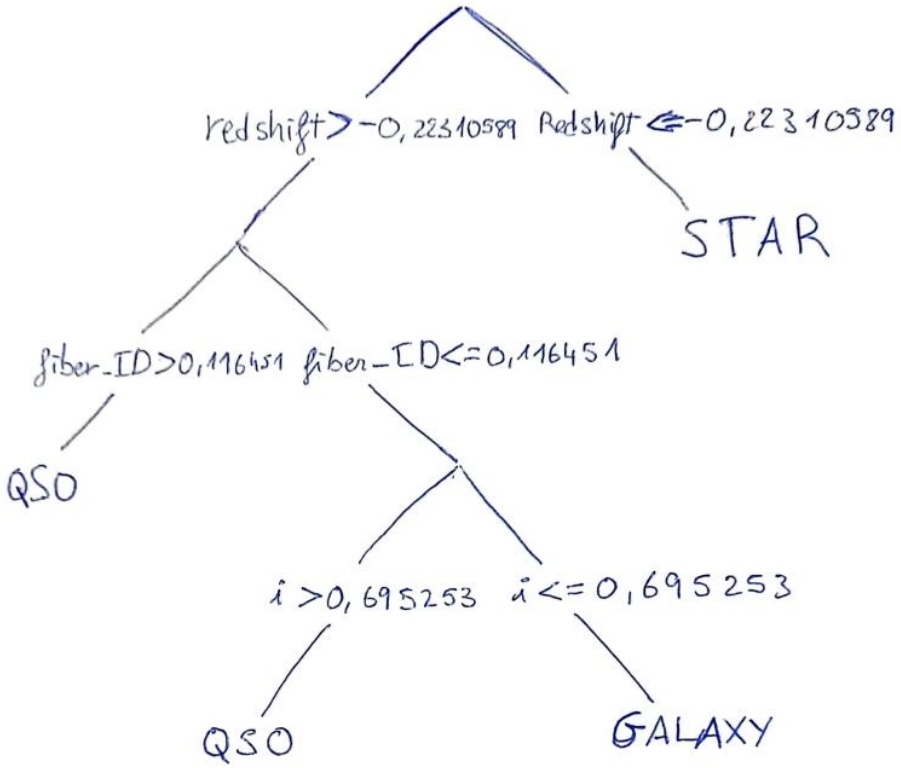
Column1	field_ID	Dist.Corte	class	Medida
1	2,839542	1,560587	GALAXY	0,75
0	0,281633	-0,12407	GALAXY	0,75
2	-0,52977	-0,59904	GALAXY	1
9	-0,66831		QSO	

Column1	redshift	Dist.Corte	class	Medida
2	1,578029	1,312396866	GALAXY	0,75
0	1,046764	0,83759535	GALAXY	0,75
1	0,628426	0,328926052	GALAXY	1
9	0,029426		QSO	

Column1	fiber_ID	Dist.Corte	class	Medida
1	-0,21614	-0,2429	GALAXY	0,75
0	-0,26966	-0,30598	GALAXY	0,75
9	-0,3423	-0,70547	QSO	0,75
2	-1,06865		GALAXY	

Podemos ver que tenemos valores 1 para unos cuantos atributos. En este caso, hemos escogido el valor de corte $v_{corte} = 0{,}695253$ del atributo i.

Como ya hemos discriminado totalmente todo el conjunto de datos, damos por finalizada la construcción del dendrograma. El dendrograma resultante se puede representar gráficamente así:



b) Calcula la exactitud del dendrograma propuesto.

Para ello, determinaremos las clases de los objetos de TEST:

Column1	alpha	g	i	field_ID	redshift	fiber_ID	class
10	0,458831675	0,394332271	0,269156	-1,064115	-0,099208	-0,437869	STAR
11	-0,39941926	0,204963876	-0,2698	0,7615498	-0,210259	-0,39964	GALAXY
12	-0,38696213	-1,7710448	0,272781	0,326161	-0,106319	0,6631263	QSO

Para el objeto 10, en primera instancia nos fijaremos en redshift. $A_{redshift} = -0.099208$. $A_{redshift} > -0{,}2231089$, así que nos fijaremos en el atributo fiber_ID. $A_{fiber_{ID}} = -1{,}064115$. $A_{fiber_{ID}} < 0{,}116451$. Finalmente, nos fijamos en el objeto i. $A_i = 0{,}269156$. $A_i < 0{,}695253$, así que el árbol de decisión nos indica que el objeto 10 es GALAXY.

Siguiendo el mismo procedimiento con los otros objetos, encontramos que:

$$10 \in GALAXY$$
$$11 \in GALAXY$$
$$12 \in QSO$$

La exactitud del modelo se puede calcular así:

$$Exactitud = \frac{Predicciones\ correctas}{Predicciones\ totales} = \frac{2}{3} = 0{,}67$$

Ejercicio 3

a) Realiza el mismo tratamiento de datos que en el Ejercicio 1, utilizando herramientas que consideres adecuadas. Muestra los primeros 10 ejemplos ya tratados.

En primera instancia, cargaremos los datos y analizaremos brevemente de qué están compuestos. Concretamente, veremos el número de ejemplos, de atributos y de clases. Mostraré los 10 primeros ejemplos para ver como están formulados los datos:

```
# Cargamos los datos
my_data = pd.read_csv('star_classification.csv')
# Creamos un array para los nombres de los atributos
attribute_names = [ "obj_ID","alpha","delta","u","g","r","i","z","run_ID","rerun_ID","cam_col","field_ID","spec_obj_ID","class","redshift","plate","MJD","fiber_ID" ]
# Cuantos ejemplos hay (shape[0] nos indica la primera dimensión del array Numpy, que equivale a los ejemplos)
print(my_data.head(10))
sample_num = my_data.shape[0]
print("número de ejemplos: ", sample_num)
# Cuantos atributos hay (shape[1] nos indica la segunda dimensión del array Numpy, que equivale a los atributos)
attribute_num = my_data.shape[1]
print("Número de atributos: ", attribute_num)
# Se busca el índice correspondiente a class y se indican cuantas ocurrencias únicas hay
class_num = len(np.unique(my_data['class']))
print("Número de clases: ",class_num)
```

```
         obj_ID     alpha     delta        u        g        r        i        z  run_ID  rerun_ID  cam_col  field_ID     spec_obj_ID   class  redshift  plate    MJD  fiber_ID
0  1.237661e+18  135.689107  32.494632  23.87882  22.27530  20.39501  19.16573  18.79371    3606       301        2        79  6.543777e+18  GALAXY  0.634794   5812  56354       171
1  1.237665e+18  144.826101  31.274185  24.77759  22.83188  22.58444  21.16812  21.61427    4518       301        5       119  1.176014e+19  GALAXY  0.779136  10445  58158       427
2  1.237661e+18  142.188790  35.582444  25.26307  22.66389  20.60976  19.34857  18.94827    3606       301        2       120  5.152200e+18  GALAXY  0.644195   4576  55592       299
3  1.237663e+18  338.741038  -0.402828  22.13682  23.77656  21.61162  20.50454  19.25010    4192       301        3       214  1.030107e+19  GALAXY  0.932346   9149  58039       775
4  1.237680e+18  345.282593  21.183866  19.43718  17.58028  16.49747  15.97711  15.54461    8102       301        3       137  6.891865e+18  GALAXY  0.116123   6121  56187       842
5  1.237680e+18  340.995121  20.589476  23.48827  23.33776  21.32195  20.25615  19.54544    8102       301        3       110  5.658977e+18     QSO  1.424659   5026  55855       741
6  1.237679e+18   23.234926  11.418188  21.46973  21.17624  20.92829  20.60826  20.42573    7773       301        2       462  1.246262e+19     QSO  0.586455  11069  58456       113
7  1.237679e+18    5.433176  12.065186  22.24979  22.02172  20.34126  19.48794  18.84999    7773       301        2       346  6.961443e+18  GALAXY  0.477009   6183  56210        15
8  1.237661e+18  200.290475  47.199402  24.40286  22.35669  20.61032  19.46490  18.95852    3716       301        5       108  7.459285e+18  GALAXY  0.660012   6625  56386       719
9  1.237671e+18   39.149691  28.102842  21.74669  20.03493  19.17553  18.81823  18.65422    5934       301        4       122  2.751763e+18    STAR -0.000008   2444  54082       232
número de ejemplos:  100000
Número de atributos:  18
Número de clases:  3
```

Posteriormente, miramos cuantos objetos tenemos de cada clase y normalizamos mediante estandarización (he considerado que no hay diferencia entre el orden de selección univariante y normalización).

```python
# Mostramos cuantos objetos tenemos de cada clase
print(my_data['class'].value_counts())

# Normalizamos mediante estandarización. Para ello, separaremos la clase del dataset, normalizaremos, y la uniremos otra vez.
my_data.pop('class')
scaler = StandardScaler()
scaler.fit(my_data)
my_data_normalized = scaler.transform(my_data)

# Para añadir los datos, usaremos los dataframes de Panda
normalized_dataframe = pd.DataFrame(my_data_normalized, columns=attribute_names[:-1])
normalized_dataframe['class'] = class_data

# Mostramos los datos normalizados
print(normalized_dataframe.head(10))
```

```
class
GALAXY    59445
STAR      21594
QSO       18961
Name: count, dtype: int64
     obj_ID     alpha     delta         u         g         r         i         z    run_ID  rerun_ID   cam_col  field_ID  spec_obj_ID  redshift     plate       MJD  fiber_ID   class
0 -0.445634 -0.434604  0.425529  0.059755  0.054926  0.403962  0.046007  0.003937 -0.445535       0.0 -0.952553 -0.718947     0.228609  0.079557  0.228633  0.423203 -1.021342  GALAXY
1  0.018740 -0.339921  0.363402  0.088045  0.072456  1.584406  1.185097  0.092835  0.018646       0.0  0.937920 -0.450589     1.797912  0.277096  1.797924  1.420729 -0.081883  GALAXY
2 -0.445633 -0.367251  0.582713  0.103327  0.067165  0.519745  0.150019  0.008808 -0.445535       0.0 -0.952553 -0.443798    -0.190037  0.092423 -0.190025  0.001854 -0.551612  GALAXY
3 -0.147311  1.669523 -1.249105  0.004921  0.102210  1.059904  0.807610  0.018321 -0.147278       0.0 -0.322395  0.187031     1.358962  0.486770  1.358942  1.354927  1.195196  GALAXY
4  1.842768  1.737310 -0.150242 -0.080055 -0.092948 -1.697421 -1.767887 -0.098468  1.842792       0.0 -0.322395 -0.329712     0.333328 -0.630267  0.333297  0.330860  1.441070  GALAXY
5  1.842767  1.692881 -0.180499  0.047461  0.088389  0.903727  0.666309  0.027630  1.842792       0.0 -0.322395 -0.510908    -0.037577  1.160523 -0.037601  0.147280  1.070424     QSO
6  1.675255 -1.599911 -0.647361 -0.016077  0.020310  0.691483  0.866612  0.055374  1.675341       0.0 -0.952553  1.851345     2.009247  0.013403  2.009285  1.585508 -1.234189     QSO
7  1.675254 -1.784381 -0.614425  0.008477  0.046939  0.374982  0.229301  0.005710  1.675341       0.0 -0.952553  1.072875     0.354260 -0.136378  0.354298  0.343578 -1.593826  GALAXY
8 -0.389456  0.234828  1.174070  0.076250  0.057490  0.520847  0.216195  0.000131 -0.389548       0.0  0.937920 -0.524330     0.504032  0.114070  0.504012  0.440898  0.989689  GALAXY
9  0.739380 -1.434994  0.201966 -0.007359 -0.015636 -0.253529 -0.151673 -0.000460  0.739346       0.0  0.307763 -0.430376    -0.912190 -0.789196 -0.912177 -0.833104 -0.797487    STAR
```

Realizamos la selección de características. Se ha decidido escoger las características con una puntuación superior a 100. En este caso son 10 atributos seleccionados.

```python
'''
    Se decide realizar una selección de características univariante para evaluar la función de bonanza de las características.
    Posteriormente, analizaremos si alguna de las variables puede ser eliminada. En cualquier caso, una vez decididas las características,
    normalizaremos los datos para su posterior tratamiento.
'''
# Usaremos sklearn para ello. Crearemos un objeto SelectKBest y escogeremos las mejores características
# (serán todas con una puntuación superior a 100, en este caso son 10)
# Documentación: https://scikit-learn.org/stable/modules/feature_selection.html#removing-features-with-low-variance
# separamos las características de la clase

Y= normalized_dataframe.pop('class')
X= normalized_dataframe
X_new = SelectKBest(f_classif, k=10).fit(X,Y)
feature_scores = X_new.scores_
selected_features_indices = X_new.get_support()


# Mostraremos las puntuaciones del algoritmo, y las características seleccionadas.
selected_features = np.array(attribute_names[:-1])[selected_features_indices]
print("Puntuaciones de las características:")
for feature, score in zip(attribute_names[:-1], feature_scores):
    print(f"{feature}: {score}")

print("\nCaracterísticas seleccionadas:")
for feature in selected_features:
    print(feature)

# Finalmente, transformamos el array y comprobamos que las dimensiones sean las adecuadas
my_data_transformed = X_new.transform(X)
print(my_data_transformed.shape)
transformed_dataframe = pd.DataFrame(my_data_normalized, columns=attribute_names[:-1])
transformed_dataframe['class'] = class_data
print(transformed_dataframe.head(10))
```

```
Puntuaciones de las características:
obj_ID: 122.54245490821458
alpha: 21.948822226127003
delta: 217.58835661128475
u: 30.445338960932055
g: 25.96252257517714
r: 4584.533363526937
i: 8282.343545240854
z: 32.32830785749901
run_ID: 122.55034045316454
rerun_ID: nan
cam_col: 26.96483886445349
field_ID: 79.69839973437045
spec_obj_ID: 5169.594514534656
redshift: 83429.41896710458
plate: 5169.585516628427
MJD: 4427.629370741775
fiber_ID: 436.7950274413851

Características seleccionadas:
obj_ID
delta
r
i
run_ID
spec_obj_ID
redshift
plate
MJD
fiber_ID
(100000, 10)
```

	obj_ID	delta	r	i	run_ID	spec_obj_ID	redshift	plate	MJD	fiber_ID	class
0	-0.445634	0.425529	0.403962	0.046007	-0.445535	0.228609	0.079557	0.228633	0.423203	-1.021342	GALAXY
1	0.018740	0.363402	1.584406	1.185097	0.018646	1.797912	0.277096	1.797924	1.420729	-0.081883	GALAXY
2	-0.445633	0.582713	0.519745	0.150019	-0.445535	-0.190037	0.092423	-0.190025	0.001854	-0.551612	GALAXY
3	-0.147311	-1.249105	1.059904	0.807610	-0.147278	1.358962	0.486770	1.358942	1.354927	1.195196	GALAXY
4	1.842768	-0.150242	-1.697421	-1.767887	1.842792	0.333328	-0.630267	0.333297	0.330860	1.441070	GALAXY
5	1.842767	-0.180499	0.903727	0.666309	1.842792	-0.037577	1.160523	-0.037601	0.147280	1.070424	QSO
6	1.675255	-0.647361	0.691483	0.866612	1.675341	2.009247	0.013403	2.009285	1.585508	-1.234189	QSO
7	1.675254	-0.614425	0.374982	0.229301	1.675341	0.354260	-0.136378	0.354298	0.343578	-1.593826	GALAXY
8	-0.389456	1.174070	0.520047	0.216195	-0.389548	0.504032	0.114070	0.504012	0.440898	0.989689	GALAXY
9	0.739380	0.201966	-0.253529	-0.151673	0.739346	-0.912190	-0.789196	-0.912177	-0.833104	-0.797487	STAR

Finalmente, separaremos el conjunto de datos en un conjunto de train y uno de test. Hemos decidido una proporción 80-20 para los conjuntos de train y test respectivamente.

```python
# Creamos una partición de entreno y de test (80-20). Para cada una, tendremos la clase y los atributos separados. Si es conveniente unirlos posteriormente, se hará
# Documentación: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

X_train, X_test, Y_train, Y_test = train_test_split(my_data_transformed, Y, test_size = 0.2, random_state = 42)

# Miramos la medida de los ejemplos
print("Ejemplos X_train : ", X_train.shape[0])
print("Ejemplos de X_test: ", X_test.shape[0])
```

```
Ejemplos X_train :  80000
Ejemplos de X_test:  20000
```

b) Utiliza un algoritmo de clasificación de vecinos más cercanos (K-Nearest Neighbors) sobre los datos tratados. Calcula y presenta la exactitud, precisión, recall, fall-out y matriz de confusión.

```python
'''
    Se aplicará una clasificación de vecinos más cercanos (K-Neares Neighbors)
    Documentación: https://scikit-learn.org/stable/modules/neighbors.html#nearest-neighbors-classification
    https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier
'''
# Aplicamos clasificador al conjunto de datos de entrenamiento. Decidimos utilizar 5 vecinos próximos para la decisión. Dejamos los pesos de los atributos como uniformes.

KNN = KNeighborsClassifier(n_neighbors = 5)
KNN.fit(X_train,Y_train)

# Predecimos sobre el conjunto de entrenamiento
KNN_pred = KNN.predict(X_test)

# Calculamos las métricas. Para ello, usaremos las metricas de sklearn
# https://scikit-learn.org/stable/modules/model_evaluation.html#classification-metrics

# Cálculo de la exactitud (accuracy)
knn_accuracy = accuracy_score(Y_test,KNN_pred)
# Cálculo de la precisión
knn_precision = precision_score(Y_test,KNN_pred, average = 'weighted')
# Cálculo del recall
knn_recall = recall_score(Y_test,KNN_pred, average = 'weighted')
# Cálculo de la matriz de confusión
knn_conf_matrix = confusion_matrix(Y_test,KNN_pred)

# Cálculo del fall-out
# https://stackoverflow.com/questions/31324218/scikit-learn-how-to-obtain-true-positive-true-negative-false-positive-and-fal
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html#sklearn.metrics.confusion_matrix
TN = knn_conf_matrix[0][0]
FN = knn_conf_matrix[1][0]
TP = knn_conf_matrix[1][1]
FP = knn_conf_matrix[0][1]
FPR = FP/(FP+TN)
#Exponemos las métricas calculadas
print("Exactitud: ", knn_accuracy)
print("Precisión: ", knn_precision)
print("Recall: ", knn_recall)
print("Fall-out: ", FPR)
print("Matriz de Confusión: ")
print(knn_conf_matrix)
```

```
Exactitud:  0.9231
Precisión:  0.9243090996671368
Recall:  0.9231
Fall-out:  0.009363620599965 32
Matriz de Confusión:
[[11426   108   326]
 [  531  3238    28]
 [  544     1  3798]]
```

c) Repite el paso anterior utilizando un clasificador de árbol de decisión.

```python
'''
    Repetimos el mismo proceso usando un árbol de decisión
    Documentación: https://scikit-learn.org/stable/modules/tree.html#classification
    https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier
'''
# Aplicamos el DecisionTreeClassifier sobre el conjunto de entrenamiento
dtc = DecisionTreeClassifier(random_state = 0)
dtc.fit(X_train,Y_train)

#Predecimos sobre el conjunto de test
dtc_pred = dtc.predict(X_test)

# Calculamos las métricas. Para ello, usaremos las metricas de sklearn
dtc_accuracy = accuracy_score(Y_test,dtc_pred)
dtc_precision = precision_score(Y_test,dtc_pred, average ='weighted')
dtc_recall = recall_score(Y_test,dtc_pred, average = 'weighted')
dtc_conf_matrix = confusion_matrix(Y_test,dtc_pred)
TN = dtc_conf_matrix[0][0]
FN = dtc_conf_matrix[1][0]
TP = dtc_conf_matrix[1][1]
FP = dtc_conf_matrix[0][1]
FPR = FP/(FP+TN)
#Exponemos las métricas calculadas
print("Exactitud: ", dtc_accuracy)
print("Precisión: ", dtc_precision)
print("Recall: ", dtc_recall)
print("Fall-out: ", FPR)
print("Matriz de Confusión: ")
```

```
Exactitud:  0.9606
Precisión:  0.9606798338778372
Recall:  0.9606
Fall-out:  0.03209188413140782
Matriz de Confusión:
[[11461   380    19]
 [  361  3436     0]
 [   27     1  4315]]
```

Todo el código y los datos pueden encontrarse en: https://github.com/moosemaniac/PEC2MLUOC

Ejercicio 4

a) ¿Dirías que el conjunto de datos se puede considerar balanceado? Nuestro conjunto de datos se nos da con una cantidad de datos por clase así:

```
class
GALAXY    59445
STAR      21594
QSO       18961
```

Como se puede observar, tenemos muchísimos más ejemplos de GALAXY que de STAR y QSO, así que estamos ante un conjunto desbalanceado.

b) ¿Cómo crees que afecta el desbalanceo a la Exactitud (accuracy)?

Un desbalanceo en los datos nos puede afectar negativamente a la exactitud, dado que nos puede dar una exactitud alta simplemente dando como ejemplo la clase mayoritaria (recordamos que la exactitud no distingue entre falsos positivos y falsos negativos). En nuestro caso no parece afectar dadas las matrices de confusión de ambos clasificadores implementados.

c) Investiga y explica brevemente otras métricas de evaluación de modelos que puedan ser más adecuadas para conjuntos de datos desbalanceados.

Para un conjunto de datos desbalanceado, la precisión nos puede ser útil para detectar la cantidad de falsos positivos que se nos dan para cada clase. De las métricas ya usadas, el recall o el fall-out nos pueden dar buena información sobre la tasa de verdaderos positivos o falsos negativos. Otras métricas no usadas que pueden ser útiles son:

- F1_score: se define como la media armónica entre la precisión y el recall. Puede ser útil para ver la media entre verdaderos positivos y falsos positivos. Se calcula así:

$$F1 = \frac{2TP}{2TP + FP + FN}$$

Se puede encontrar más información en: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html#sklearn.metrics.f1_score

- ROC AUC score: Calcula el área debajo la curva ROC (Reciever Operating Characteristic) para clases predecidas. Se puede encontrar más información en: https://scikit-learn.org/stable/modules/model_evaluation.html#roc-metrics
- Balanced accuracy: es la media aritmética de la sensibilidad (Recall) y la especifidad (tasa de verdaderos negativos). Este tipo de exactitud evita los estimados inflados por datasets desbalanceados. Se calcula así:

$$balanced - accuracy = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right)$$

Se puede encontrar más información en: https://scikit-learn.org/stable/modules/model_evaluation.html#balanced-accuracy-score

Recursos

Básicos

Para realizar esta PEC disponéis de los datos contenidos en *star_classification.csv*, así como los apuntes de la asignatura.

Criterios de valoración

Cada uno de los cuatro ejercicios se valorará con 2.5 puntos como máximo. No

No existe una puntuación o un peso específico para cada apartado de los ejercicios, sino que cada ejercicio se valorará globalmente. Recordad que debéis justificar todas las respuestas: una respuesta sin justificar se valorará con cero puntos.

Formato y fecha de entrega

Tenéis que entregar la PEC en un fichero zip con el pdf de la memoria al registro de actividades de evaluación continua.

El nombre del archivo tiene que ser ApellidosNombre_AC_PEC1 con extensión .zip (ZIP).

Fecha límite: 0**7/06/2024**

Para dudas o aclaraciones sobre el enunciado, dirigíos al consultor responsable de vuestra aula.

Nota: **Propiedad intelectual**

A menudo es inevitable, al producir una obra multimedia, hacer uso de recursos creados por terceras personas. Es por tanto comprensible hacerlo en el marco de una práctica de los estudios del Grado de Informática, siempre que esto se documente claramente y no suponga plagio en la práctica.

Por lo tanto, al presentar una práctica que haga uso de recursos ajenos, se presentará junto con ella un documento en el que se detallen todos ellos, especificando el nombre de cada recurso, su autor, el lugar donde se obtuvo y el su estatus legal: si la obra está protegida por copyright o se acoge a alguna otra licencia de uso (Creative Commons, licencia GNU, GPL ...). El estudiante deberá asegurarse de que la licencia que sea no impide específicamente su uso en el marco de la práctica. En caso de no encontrar la información correspondiente deberá asumir que la obra está protegida por copyright.

Deberán, además, adjuntar los archivos originales cuando las obras utilizadas sean digitales, y su código fuente si corresponde.

.