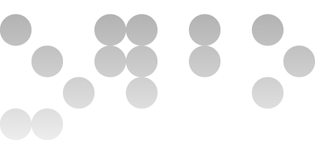
75.583 · Aprendizaje Computacional · PEC2 · 2023-24 · Ingeniería Informática · Estudios de Informática Multimedia y Telecomunicaciones



**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)](https://www.sdss.org/). 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):



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.



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:

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):



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 .

Se acompaña con todas las funciones de bonanza:



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.*



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:

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



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



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

TRAIN:



TEST:



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:

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



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:

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.
* 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.
* 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.

**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.



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 :

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 -0.22310589 del atributo redshift (distancia de corte entre el objeto 9 y 5). Creamos las dos particiones:

Para

Para

Los separamos en tabla para que se más gráfico:



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:

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

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 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.



Volvemos a calcular las distancias de corte y las medidas:





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:



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:



Calculamos las distancias de corte y las medidas:



Podemos ver que tenemos valores 1 para unos cuantos atributos. En este caso, hemos escogido el valor de corte 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:

A diagram of a star

Description automatically generated with medium confidence

b) Calcula la exactitud del dendrograma propuesto.

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



Para el objeto 10, en primera instancia nos fijaremos en redshift. . , así que nos fijaremos en el atributo fiber\_ID. . Finalmente, nos fijamos en el objeto i. , así que el árbol de decisión nos indica que el objeto 10 es GALAXY.

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

La exactitud del modelo se puede calcular así:

**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:

A black screen with text

Description automatically generated A black screen with white text

Description automatically generated with medium confidence

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).

A computer screen with text on it

Description automatically generatedA screen shot of a computer screen

Description automatically generated

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

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated A screenshot of a computer screen

Description automatically generated

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.

A computer screen shot of a computer screen

Description automatically generated 

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.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

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

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

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:

A screenshot of a computer screen

Description automatically generated

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:

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:

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.  . |