Métricas

Aprendizaje Automático

Hasta ahora, en prácticas hemos utilizado para cuantificar la bondad del modelo funciones sencillas, como el MSE en problemas de regresión o la precisión en problemas de clasificación. Si bien en problemas de regresión las funciones se basan en un error calculado de una u otra manera (error medio, error cuadrático medio, etc.), en problemas de clasificación se pueden derivar otro tipo de métricas en función de cómo es el problema que se está resolviendo. Muchas de estas métricas, al menos las que se van a usar en prácticas, se basan en el cálculo de la matriz de confusión.

Una matriz de confusión es una matriz cuadrada, con tantas filas y columnas como clases, donde se muestra la distribución de los patrones en clases, y la clasificación que realiza el modelo. Habitualmente en las filas se pone cómo ha efectuado el modelo la clasificación, y en las columnas los valores de clasificación real, aunque esto pueda variar según la fuente que se consulte.

El caso más sencillo se corresponde con 2 clases, en las que una se considera "negativa" y otra "positiva". Una matriz de confusión de dos clases sería la siguiente:

		Predicción	
		Negativo	Positivo
Real	Negativo	VN	FP
	Positivo	FN	VP

Esta matriz de confusión contiene 4 valores, que podemos dividir

- Según la salida que da el modelo: positivos o negativos.
- Según si el modelo se equivoca o no: verdaderos o falsos.

De esta forma, estos 4 valores reciben por nombre verdaderos negativos (VN), falsos positivos (FP), falsos negativos (FN) y verdaderos positivos (VP). Por ejemplo, los falsos negativos sería la cantidad de patrones que el sistema ha clasificado en negativo, y se ha equivocado porque en realidad eran positivos.

A partir de esta matriz de confusión se pueden calcular distintas métricas. Dependiendo del

problema en que se esté trabajando, será más interesante seguir una u otra. Algunas de las métricas más usadas son:

- Precisión (*accuracy*, no confundir con el término inglés *precision*). Ratio de patrones en los que acierta en la predicción. Se calcula como (VN + VP)/(VN + VP + FN + FP)
- Tasa de error (*error rate*). Ratio de patrones en los que falla en la predicción. Se calcula como (FN + FP)/(VN + VP + FN + FP)
- Sensibilidad (*recall* o *sensitivity*). Indica la probabilidad de que para un caso positivo se obtenga en el clasificador un resultado positivo. Se calcula como VP/(FN+VP)
    - En una prueba médica, la sensibilidad de la prueba representa la probabilidad de que un sujeto enfermo (positivo) tenga un resultado positivo en la prueba.
- Especificidad (*specificity*). Indica la probabilidad de que para un caso negativo se obtenga en el clasificador un resultado negativo. Se calcula como VN/(FP+VN)
    - La especificidad de una prueba representa la probabilidad de que un sujeto sano (negativo) tenga un resultado negativo en la prueba.
- Valor predictivo positivo (*precision* o *positive predictive value*). Ratio de casos con valor positivo que han sido correctamente clasificados. Se calcula como VP/(VP+FP).
- Valor predictivo negativo (*negative predictive value*). Ratio de casos con valor negativo que han sido correctamente clasificados. Se calcula como VN/(VN+FN).
- F_1-*score*, se define como la media **harmónica** entre VPP y sensibilidad.

Es conveniente aclarar que estas métricas, así como otras vistas en clase de teoría (curva ROC, índice Kappa) se utilizan para valorar clasificadores ya entrenados, no para realizar el proceso de entrenamiento. Para entrenarse, cada modelo tiene su propia función para cuantificar el error (también llamado *loss*) o la bondad, como puede ser la función *cross-entropy* en el caso de las redes de neuronas.

Posiblemente, el valor más utilizado es el de precisión (*accuracy*), puesto que indica, de una forma sencilla, la tasa de éxito del clasificador. Sin embargo, dependiendo del problema con el que se está trabajando, podría no ser la métrica más adecuada. Por ejemplo, en un test masivo en una población sobre una enfermedad en el que se sabe que la mayoría de la gente no padece esa enfermedad, un modelo que clasifique a todas las personas como negativo (sano) tendrá una precisión muy alta,

aunque el modelo realmente no hace nada.

Por este motivo, es necesario valorar, entre todas estas métricas, cuál o cuáles son las más utilizadas, en función del problema. En muchos problemas en los que las distintas clases tienen la misma importancia, el valor de precisión puede ser suficiente, dependiendo del problema. Sin embargo, en otros problemas puede interesar más evaluar las situaciones en las que se produce o debería producir una respuesta positiva por parte del modelo, puesto que podría indicar algo crítico, como detectar una enfermedad o dar algún tipo de alarma. Por este motivo, muchas veces se tienen en cuenta, aparte de la precisión, los valores de sensibilidad y valor predictivo positivo. En los apuntes de teoría aparece una discusión más extensa sobre esto, pero una posible guía informal podría ser la siguiente:

- Si se desea minimizar el número de positivos clasificados incorrectamente como negativos (por ejemplo, maximizar el número de sujetos enfermos diagnosticados como sanos, o maximizar el número de alarmas que se dan ante situaciones de riesgo), la métrica indicada es la sensibilidad.

- Si se desea minimizar el número de positivos detectados incorrectamente (falsos positivos, es decir, clasificados como negativos, por ejemplo, sujetos sanos diagnosticados como enfermos, o situaciones en las que se debería haber dado una alarma pero no se dio), la métrica indicada es el valor predictivo positivo.

Por lo tanto, depende del problema escoger la métrica más adecuada en función de la importancia relativa de su salida y de su comportamiento. En este tipo de problemas, F_1-*score* es una métrica que puede ser de más utilidad que la precisión.

Otra cuestión a tener en cuenta es el posible desbalanceo de los datos. La precisión es una métrica que ofrece una visión "global", sin tener en cuenta que puede ser engañosa cuando la distribución en clases está desbalanceada, y en estos casos F_1-*score* es una mejor métrica. El tener bases de datos desbalanceadas es algo muy común, lo cual aporta un argumento extra para utilizar F_1-*score* antes que precisión.

Finalmente, si se tienen más de dos clases, es posible construir una matriz de confusión de una forma similar teniendo una fila y columna por clase, por ejemplo:

		Predicción		
		A	B	C
Real	A			
	B			
	C			

En estos casos, ya no se puede hablar de patrones positivos o negativos, puesto que hay más de dos clases, ni tomar valores de sensibilidad o valor predictivo positivo. Sin embargo, esta matriz de confusión puede ofrecer información muy interesante a la hora de comprender el funcionamiento del modelo, viendo cuáles son las clases entre las que el modelo encuentra mayor facilidad y dificultad en su separación.

- Si se ha dividido el conjunto de patrones en entrenamiento y test, ¿en cuál de los dos habría que calcular la matriz de confusión?

La razón detrás de esto es que la matriz de confusión proporciona una forma de evaluar la calidad de las predicciones del modelo en datos no vistos, es decir, datos que no se han utilizado durante el entrenamiento.

Porque con los datos entrenados ya sabemos que va a clasificar bien, ahora necesitamos probar con datos externos al entrenamiento

En esta práctica, se pide:

- Desarrollar una función llamada *confusionMatrix* que acepte dos vectores de igual longitud (igual al número de patrones), el primero con las salidas obtenidas por un modelo *outputs* y el segundo con las salidas deseadas *targets*, ambos de tipo *AbstractArray{Bool,1}* y devuelva una tupla con los siguientes valores, por este orden:
  - Valor de precisión.
  - Tasa de fallo.
  - Sensibilidad.
  - Especificidad.
  - Valor predictivo positivo.
  - Valor predictivo negativo.
  - F_1-*score*.
  - Matriz de confusión, como un objeto de tipo *Array{Int64,2}* con dos filas y dos columnas.

  Dado que se le están dando vectores de valores booleanos, esta función se aplicará en

problemas de dos clases (casos positivos y negativos).

En el cálculo de estos valores, es necesario tener en cuenta los siguientes casos particulares:

- Si no hay patrones que pertenezcan a la clase positiva y ninguna instancia ha sido clasificada como positiva, el valor de sensibilidad (*recall*) no se podrá calcular (0/0). En este caso, no se habrá fallado ningún caso de instancias positivas, por lo que el valor de sensibilidad será igual a 1. Esto se puede saber fácilmente, si VP = FN = 0.

- De igual manera, si VP = FP = 0, no se puede calcular el valor predictivo positivo (*precision*), pero, sabiendo que no se ha fallado ninguna predicción hecha como positiva, esta métrica debería tomar el valor de 1.

- El mismo razonamiento se puede seguir para las métricas especificidad (*specificity*) (TN = FP = 0) y VPN (*NPV*) (VN = FN = 0).

- Para el cálculo de F_1, si la sensibilidad y el valor predictivo positivo son iguales a 0, esta no se puede calcular. En ese caso, el valor de F_1 será igual a 0.

  - ¿Por qué debería de ser igual a 0 en este caso?

F1 es una medida que combina la precisión y la sensibilidad del modelo. Si tanto la sensibilidad como el valor predictivo positivo son iguales a 0, significa que el modelo no ha identificado correctamente ninguna instancia positiva de la clase en cuestión.

No utilizar bucles en el desarrollo de esta función.

- Muchos modelos (por ejemplo, RR.NN.AA.) no van a devolver una salida categórica, sino que asignarán un valor de probabilidad a la clase "positivo". Por este motivo, se pide desarrollar una función de nombre igual que la anterior, cuyo primer parámetro, en lugar de ser un vector de valores booleanos, sea un vector de valores reales (de tipo *AbstractArray{<:Real}*), y con un tercer parámetro opcional que tenga un umbral, con un valor por defecto, y los utilice para aplicar la función anterior y devolver, por lo tanto, los mismos valores. Por el mismo motivo, no se permite el uso de bucles en esta función.

- Desarrollar dos funciones del mismo nombre, *printConfusionMatrix*, que reciban las salidas del modelo y las salidas deseadas, llamen a las funciones anteriores y muestren por pantalla los resultados obtenidos, incluida la matriz de confusión. Una de estas funciones recibirá como vector de salidas del modelo *outputs* un vector de tipo *AbstractArray{Bool,1}*, mientras que para la otra este parámetro será un vector de valores reales (de tipo *AbstractArray{<:Real,1}*). Estas funciones realizarán llamadas a las funciones anteriores. Estas funciones no serán evaluadas.

Al final de este documento se incluyen las firmas de las funciones a realizar en estos ejercicios.

Firmas de las funciones:

A continuación se muestran las firmas de las funciones a realizar en los ejercicios propuestos. Tened en cuenta que, dependiendo de cómo se defina la función, esta puede contener o no la palabra reservada *function* al principio:

```
function confusionMatrix(outputs::AbstractArray{Bool,1}, targets::AbstractArray{Bool,1})
function confusionMatrix(outputs::AbstractArray{<:Real,1},
        targets::AbstractArray{Bool,1}; threshold::Real=0.5)

function printConfusionMatrix(outputs::AbstractArray{Bool,1},
        targets::AbstractArray{Bool,1})
function printConfusionMatrix(outputs::AbstractArray{<:Real,1},
        targets::AbstractArray{Bool,1}; threshold::Real=0.5)
```

Matriz de confusión que devuelve todas las métricas,coge los targets y nos da las métricas