



ALGORITMOS DE CLASIFICACIÓN

DIFERENCIAS ENTRE UN PROBLEMA DE CLASIFICACIÓN BINARIA Y UN PROBLEMA DE CLASIFICACIÓN MULTICLASE



clasificador binario: Algoritmo para determinar la probabilidad de que un mensaje de correo electrónico sea spam y no spam



clasificador multiclase: Algoritmo que busque predecir en qué país de doce posibles un cliente va a realizar la siguiente reserva

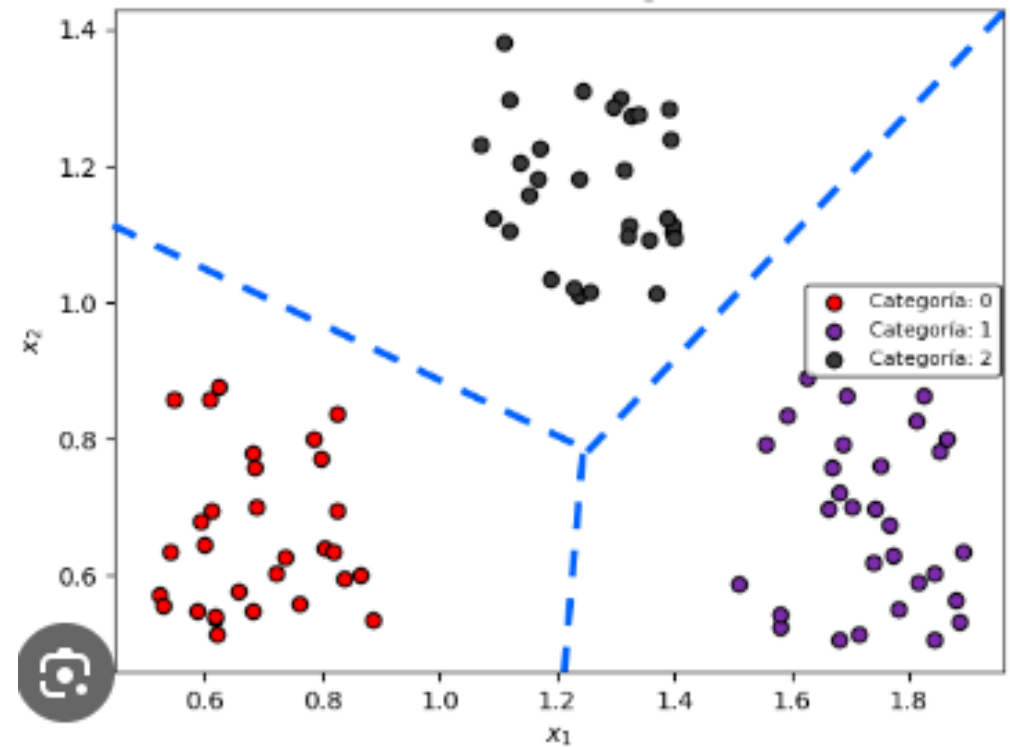
VENTAJA COMPETITIVA

- Capacidad para adelantarte a situaciones futuras de la forma más adecuadas y gestionarlas mejor
- Cualquier evaluación corre el riesgo de ser subjetiva, por tanto, es necesario establecer los criterios de evaluación más objetivos y rigurosos posibles



PROPÓSITO

- Objetivo: Obtener la clase más probable para cada una de sus instancias
- Se pueden utilizar para predecir o estimar la probabilidad de pertenencia de una clase de entre 2 posibles (Clasificación binaria)
- Predecir o estimar la probabilidad de una clase de entre varias posibles También esta definida para un vector de (Clasificación multiclase)



EVALUANDO EL RENDIMIENTO



Cualquier evaluación corre el riesgo de ser subjetiva



Es necesario establecer los métodos de evaluación más objetivos en cualquier escenario



Los algoritmos de los diferentes modelos pueden tener fortalezas y debilidades y es necesario disponer de la posibilidad de evaluarlos en las mismas condiciones

EVALUANDO EL RENDIMIENTO

- La mejor métrica de rendimiento de un clasificador es ver si un clasificador tiene éxito para su propósito
- Es muy útil utilizar una serie de medidas a partir de la matriz de confusión
- Existen diferentes tipos de datos que se pueden utilizar para evaluar un clasificador: (1) los valores reales de las clases, (2) los valores predichos y (3) la probabilidad estimada de la predicción



EN LA PRÁCTICA



Lo habitual es mantener dos vectores de datos: (1) contiene la verdad de la clasificación y (2) contiene los valores predichos. Ambos vectores deben tener la misma longitud y los datos en el mismo orden



La clase verdadera se conoce de antemano y es la variable a predecir del conjunto de datos



Los valores predichos se obtienen por medio de la aplicación del modelo. En la mayoría de los paquetes de ML de Python con la función `predict()` sobre un objeto de un modelo entrenado y un `data.frame` de test.

EN LA PRÁCTICA

Incluso en el caso de los clasificadores que predicen una sola clase, es útil saber con que certeza o confianza predicen cada clase. Por ejemplo: un mensaje puede ser Spam con una prob de 0.9 ó de 0.51.



Si dos modelos cometen los mismos errores pero uno es más capaz de tener en cuenta la incertidumbre, es mejor modelo.

- Es ideal encontrar un modelo que tenga mucha confianza en las predicciones correctas y sea temeroso en aquellas dudosas

EJEMPLO: CLASIFICACIÓN BINARIA

- Hay que tener cuidado en usar la probabilidad correcta para la categoría. En una clasificación binaria es común utilizar solo una de las clases

```
> head(predicted_prob)
      no      yes
1 0.0808272 0.9191728
2 1.0000000 0.0000000
3 0.7064238 0.2935762
4 0.1962657 0.8037343
5 0.8249874 0.1750126
6 1.0000000 0.0000000
```

EJEMPLO: CLASIFICACIÓN BINARIA

- Cuando los valores predichos son de la clase ham, los valores de spam son muy cercanos a 0. Cuando los valores predichos son Spam el valor es 1. Esto sugiere que el modelo tiene mucha confianza en sus clasificaciones.

```
> head(sms_results)
  actual_type predict_type   prob_spam
1         ham          ham 2.560231e-07
2         ham          ham 1.309835e-04
3         ham          ham 8.089713e-05
4         ham          ham 1.396505e-04
5        spam          spam 1.000000e+00
6         ham          ham 3.504181e-03
```

¿QUÉ OCURRE CUANDO LAS PREDICCIONES DIFIEREN?

- Observe que los valores son menos extremos. Por ejemplo la instancia 73 tiene un 35% de ser ham y se ha clasificado como Spam

```
> head(subset(sms_results, actual_type != predict_type))
```

	actual_type	predict_type	prob_spam
53	spam	ham	0.0006796225
59	spam	ham	0.1333961018
73	spam	ham	0.3582665350
76	spam	ham	0.1224625535
81	spam	ham	0.0224863219
184	spam	ham	0.0320059616







MÉTRICAS DE ERROR












MATRIZ DE CONFUSIÓN

- Una de las dimensiones de la tabla hace referencia a las categorías de los valores predichos, la otra dimensión a las categorías reales
- Las instancias clasificadas correctamente caen en la diagonal de la matriz. Los valores fuera de la diagonal indican las instancias clasificadas incorrectamente

Two Classes

		Predicted Class	
		A	B
Actual Class	A		
	B		

Three Classes

		Predicted Class		
		A	B	C
Actual Class	A			
	B			
	C			

MATRIZ DE CONFUSIÓN

		Condition (as determined by "Gold standard")		
		Condition Positive	Condition Negative	
Test Outcome	Test Outcome Positive	True Positive	False Positive (Type I error)	Positive predictive value = $\frac{\Sigma \text{ True Positive}}{\Sigma \text{ Test Outcome Positive}}$
	Test Outcome Negative	False Negative (Type II error)	True Negative	Negative predictive value = $\frac{\Sigma \text{ True Negative}}{\Sigma \text{ Test Outcome Negative}}$
		Sensitivity = $\frac{\Sigma \text{ True Positive}}{\Sigma \text{ Condition Positive}}$	Specificity = $\frac{\Sigma \text{ True Negative}}{\Sigma \text{ Condition Negative}}$	

- 1. Tasa de verdaderos positivos (true positives): se trata de las clasificaciones correctas de las instancias que corresponden a la clase positiva.
- 2. Tasa de falsos positivos (false positives): se trata de las clasificaciones de la clase negativa que han sido incorrectamente clasificadas como clase positiva.
- 3. Tasa de verdaderos negativos (true negatives): se trata de las clasificaciones correctas de las instancias que corresponden a la clase negativa.
- 4. Tasa de falsos negativos (false negatives): se trata de las clasificaciones de la clase positiva que han sido incorrectamente clasificadas como clase negativa.

ACCURACY / ERROR RATE

- Esta métrica también se conoce como el ratio de éxito.
- Representa la proporción del número de predicciones correctas entre el número total de predicciones.
- El ratio de Error indica la proporción de predicciones incorrectas

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{error rate} = \frac{FP + FN}{TP + TN + FP + FN} = 1 - \text{accuracy}$$

SENSIVITY / SPECIFICITY

- La sensibilidad de un modelo es el ratio de los ejemplos positivos correctamente clasificados.
- La especificidad de un modelo indica la proporción de los ejemplos negativos correctamente clasificados.
- Estas métricas tienen valores de 0 a 1, siendo 1 lo más deseable

$$\text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

$$\text{sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

PRECISION / RECALL

- Precision (Positive Predictive Value PPV): Indica la proporción de ejemplos que son verdaderamente positivos. Cuando un modelo predice la clase positiva, ¿cuántas veces está en lo cierto?
- Recall: Indica que tan completos son los resultados (== sensivity)

$$\text{precision} = \frac{TP}{TP + FP}$$

$$\text{recall} = \frac{TP}{TP + FN}$$

F-MEASURE

- También conocida como F1, es una métrica que combina precisión y recall
- Se utiliza con mucha frecuencia puesto que reduce el rendimiento de un clasificador con una solo métrica.
- La métrica estándar asume que el precision y recall tienen la misma importancia, pero es posible ponderarlos para dar mayor peso a uno u otro.

$$F - \text{measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{recall} + \text{precision}} = \frac{2 \times TP}{2 \times TP + FP + FN}$$



CLASIFICACIÓN CON NÄIVE BAYES



INTRODUCCIÓN

- El clasificador Näive Bayes utiliza las probabilidades a-priori para estimar la probabilidad de los eventos futuros
- Utiliza datos de entrenamiento para calcular la probabilidad observada de cada evento en función del vector de características.
- Cuando el clasificador es utilizado con datos sin etiquetar, utiliza las probabilidades observadas para estimar la clase más probable.
- Ejemplo: Utilizar la frecuencia de las palabras en los correos spam para identificar nuevos correos spam en el futuro.



INTRODUCCIÓN

- Se utiliza principalmente para: clasificar texto, detección de intrusos en redes, diagnósticos médicos, etc.
- Si todos los eventos fueran independientes, sería imposible predecir ningún evento con los datos observados por el otro
- Los eventos dependientes son la base del modelado predictivo.
- Ejemplo, la presencia de nubes suele ser un evento predictivo de un día lluvioso. La presencia de la palabra viagra suele ser un evento predictivo de spam



NÄIVE BAYES

- La relación entre los eventos dependientes se puede describir utilizando el teorema de Bayes: $P(A|B) = \{P(B|A)P(A)\} / P(B)$
- Supongamos que queremos estimar la probabilidad de que un mensaje sea spam. Sin tener evidencias adicionales es un 0.2 (probabilidad a priori).
- Si tenemos la evidencia que el mensaje contiene la palabra viagra. La probabilidad de que viagra haya usado en mensajes de spam previos se llama verosimilitud (likelihood) y la probabilidad de que aparezca en cualquier mensaje se llama verosimilitud marginal (marginal likelihood).
- Aplicando el teorema de bayes, se puede calcular la probabilidad a-posteriori (posterior probability). Si es mayor que 0.5, es más probable que sea spam que ham y se debería filtrar.

$$P(\text{spam} | \text{Viagra}) = \frac{P(\text{Viagra} | \text{spam}) P(\text{spam})}{P(\text{Viagra})}$$

Diagram illustrating the Naïve Bayes formula for spam classification:

- $P(\text{spam} | \text{Viagra})$ is labeled as the **posterior probability**.
- $P(\text{Viagra} | \text{spam})$ is labeled as the **likelihood**.
- $P(\text{spam})$ is labeled as the **prior probability**.
- $P(\text{Viagra})$ is labeled as the **marginal likelihood**.

VIDEO

NAIVE

BAYES

TEORÍA

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Es naïve porque trata a todas las variables como independientes e igualmente importantes.



Esto no es así en el mundo real. Por ejemplo, la persona que envía el email puede ser más importante que el propio texto.



Además, la aparición de las palabras no es independientes. Si aparece la palabra viagra, es muy probable que la palabra droga aparezca cerca.



Sin embargo, aunque no se tengan en cuenta estas situaciones, el clasificador funciona relativamente bien.



CARACTERÍSTICAS

EJEMPLO SPAM

- ▶ Para obtener cada uno de los componentes hay que construir una tabla de frecuencias, que indica el número de veces que la palabra viagra ha aparecido en los mensajes de spam. Esta tabla de frecuencias se puede utilizar para calcular una tabla de verosimilitud.

Frequency	Viagra		Total
	Yes	No	
spam	4	16	20
ham	1	79	80
Total	5	95	100

Likelihood	Viagra		Total
	Yes	No	
spam	4 / 20	16 / 20	20
ham	1 / 80	79 / 80	80
Total	5 / 100	95 / 100	100

- ▶ Para calcular la probabilidad posteriori, $P(\text{Spam}|\text{Viagra}) = (4/20) * (20/100) / (5/100) = 0.8$
- ▶ La probabilidad de que un mail que contenga la palabra viagra sea spam es del 0.8.

EJERCICIO APRENDIZAJE SUPERVISADO CON NAIVE BAYES

- Analizar el archivo house-votes-84.NAMES, donde se indica que existen 435 instancias correspondientes a las votaciones de EEUU, las cuales están conformadas por 267 demócratas y 168 republicanos con un total de 435 registros. Estas votaciones están clasificadas por 16 atributos que corresponde a los sectores de los Votantes más "Class Name", que corresponde a la clase que se desea.
- Para el análisis de la Data, se recupera el archivo house-votes-84.DATA y se incorpora las columnas correspondientes a los atributos obtenidos de house-votes-84.NAMES

ANÁLISIS DE DATOS PASO I

```
#Paso 1: Revisión de datos
# Importar la libreria Pandas
import pandas as pd

#Recupero el archivo de data etiquetada
atributos = ['Class-Name', 'handicapped-infants', 'water-project-cost-sharing',
             'adoption-of-the-budget-resolution', 'physician-fee-freeze',
             'el-salvador-aid', 're-groups-in-schools', 'anti-satellite-test-ban',
             'aid-to-nicaraguan-contras', 'mx-missile', 'immigration',
             'synfuels-corporation-cutback', 'education-spending', 'superfund-right-to-sue',
             'crime', 'duty-free-exports', 'export-administration-act-south-africa']
df_datos=pd.read_csv('house-votes-84.data', sep=',', index_col='Class-Name', names=atributos)

print(df_datos.head(6))
```

ANÁLISIS DE DATOS PASO 2

```
[ ] #Paso 2
    #Reemplazo de datos correspondientes de y,n y ?. y significa que votaron por ese candidato (1)
    #n significa que no votaron por el candidato(0)
    #? significa que votaron en blanco (3), en el texto se indica que (?) no significa desconocido, signifca otro valor,
    df_datos = df_datos.replace('n','0')
    df_datos = df_datos.replace('y','1')
    df_datos = df_datos.replace('?', '3')

    df_datos.describe()
```

ENTRENAMIENTO PASO 3

```
#Paso 3
#Se Determina el conjunto de modelización y el de validación, para esto se utiliza train_test_split indicando que
#se preserve las proporciones de Dataframe original
#El bloque de entrenamiento es el 75% de los registros, y al bloque de pruebas el 25% restante
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB

X_train, X_test, y_train, y_test = train_test_split(df_datos, df_datos.index, random_state=1, stratify = df_datos.index)

#Se verifica los tamaños de los datos originales, de entrenamiento y test
print("\nNumero de datos total", df_datos.shape[0])
print("Numero de datos para Entrenamiento 75% ", X_train.shape[0])
print("Numero de datos para Test 25%", X_test.shape[0])

#Se analiza que se mantengan los porcentajes de clasificación de los datos del original, entrenamiento y test agrupado por Clase
print("_____Porcentaje Original_____")
print(df_datos.groupby("Class-Name").count()/len(df_datos))
print("_____Porcentaje entrenamiento_____")
print(X_train.groupby("Class-Name").count()/len(X_train))
print("_____Porcentaje Test_____")
print(X_test.groupby("Class-Name").count()/len(X_test))
```


PASO 4 PREDICCIÓN


```
#Paso 5: Predicción
## Se usa MultinomialNB por ser valores discretos

naive_bayes = MultinomialNB()
naive_bayes.fit(X_train,y_train)

#Probar con predicciones
predictions = naive_bayes.predict(X_test)
predictions
```

MÉTRICAS DE EVALUACIÓN

```
[ ] ##Paso 6 Evaluar Algoritmo
    ## Se calculan las siguientes métricas y matriz de confusión.
    from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
    print('Accuracy score: ', format(accuracy_score(y_test, predictions)))
```

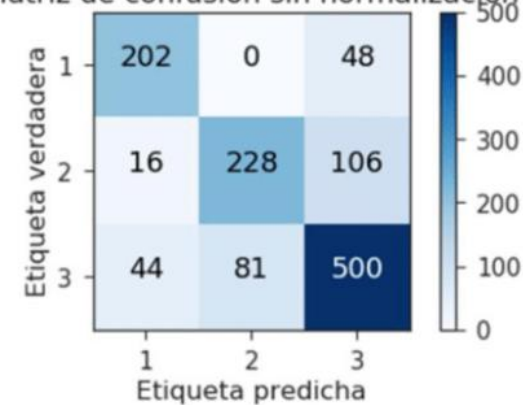
 Accuracy score: 0.8623853211009175

EJERCICIO

- Realizar la matriz confusión

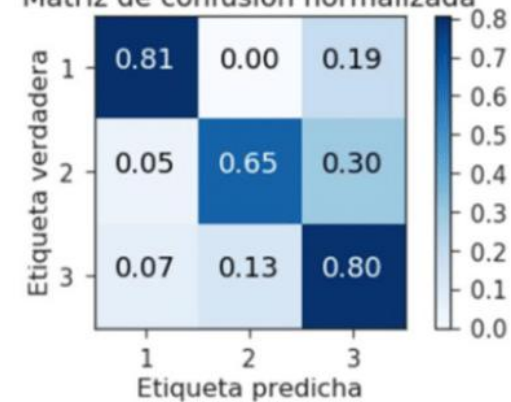
Análises de Resultados e Informe

Matriz de confusión sin normalización



(a)

Matriz de confusión normalizada



(b)



GRACIAS

ALGUIEN@EJEMPLO.COM