INSTITUTO POLITÉCNICO NACIONAL

CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN

Tarea 06

Matriz de confusión

Edgar Fernando Espinosa Torres

Clasificación inteligente de patrones

Dr. Cornelio Yáñez Márquez

Ciudad de México, 7 de octubre de 2023.

Índice general

Índice de tablas

Introducción

En el amplio mundo de la clasificación de patrones, la evaluación precisa y clara del rendimiento de un modelo es esencial. Esto ha motivado a la generación de una amplia variedad de métricas y herramientas disponibles para la evaluación.

Una de las herramientas que permite visualizar el desempeño de un clasificador inteligente de patrones que se emplea en aprendizaje supervisado es la matriz de confusión. Su origen está en el campo de la epidemiología, pero uso se ha extendido a una gran cantidad de disciplinas.

La matriz proporciona información sobre los aciertos del clasificador, pero también sobre sus errores y su tipo.

Desarrollo y discusión

Propósito de la tarea

Parte 1

1.1 Aplicar el Clasificador Euclidiano en la misma partición fija de la Tarea 04.

1.2 Reportar la matriz de confusión.

En esta parte se trabajó con un dataset derivado de Haberman's Survival de UCI [1]. Recordemos que al dataset original se le realizó un conjunto de técnicas conocidas como limpieza de datos con la finalidad de aislar al año 1965.

Este dataset tiene 28 patrones y 2 atributos numéricos: age y positive_nodes.

Vamos a definir la clase Positive como aquellos pacientes que murieron dentro de los primeros 5 años y la clase Negative como aquellos pacientes que sobrevivieron 5 años o más.

Después de aplicar el Clasificador Euclidiano en la misma partición fija de la Tarea 04 se obtuvo la siguiente tabla:

Patrón	**Clase real**	**Distancia al centroide centroide$_1$**	**Distancia al centroide centroide$_2$**	**Clase asignada**	**Resultado**
43 0	Negative	9.9	16.2	Negative	ACIERTO
50 4	Negative	2.2	8.2	Negative	ACIERTO
51 0	Negative	3.9	9.6	Negative	ACIERTO
52 0	Negative	3.7	8.9	Negative	ACIERTO
54 23	Positive	19.4	16.5	Positive	ACIERTO
54 5	Positive	2.2	4.2	Negative	ERROR
56 9	Positive	6.5	2.7	Positive	ACIERTO
60 0	Positive	8.6	7.3	Positive	ACIERTO

Tabla 1: Clasificación de los vectores de prueba del dataset Haberman's Survival 1965

Recordemos que la matriz de confusión está construida de la siguiente manera [2]:

TP	FN
FP	TN

Tabla 2: Estructura de una matriz de confusión

Donde:

- True Positive (TP): los valores de la clase Positive clasificados como Positive.
- True Negative (TN): los valores de la clase Negative clasificados como Negative.
- False Negative (FN): los valores de la clase Positive clasificados como Negative.
- False Positive (FP): los valores de la clase Negative clasificados como Positive.

Algunos detalles importantes para notar son:

- Si se suman los valores del renglón superior obtenemos el total de verdaderos Positive.
- Si se suman los valores del renglón inferior obtenemos el total de verdaderos Negative.
- Los patrones de la columna derecha se clasificaron como Negative.
- Los patrones de la columna izquierda se clasificaron como Positive.

A partir de la tabla 1 se puede generar la siguiente matriz de confusión:

3	1
0	4

Tabla 3: Matriz de confusión *para el dataset Haberman's Survival 1965 por partición fija*

Total de verdaderos Positive: $3 + 1 = 4$

Total de verdaderos Negative: $0 + 4 = 4$

Parte 2.

De forma semejante a la parte 1, se trabaja con un dataset derivado de Haberman's Survival [1] del año 1958 pero ahora se aplica el método de validación Hold-Out Estratificado 80-20.

También, vamos a definir la clase Positive como aquellos pacientes que murieron dentro de los primeros 5 años y la clase Negative como aquellos pacientes que sobrevivieron 5 años o más.

A partir de ello, se obtienen los siguientes resultados:

Patrón	**Clase real**	**Clase asignada**	**Resultado**
55 1	Negative	Negative	CORRECTO
42 0	Negative	Negative	CORRECTO
39 0	Negative	Negative	CORRECTO

54 1	Negative	Negative	CORRECTO
46 2	Positive	Negative	ERROR
48 11	Positive	Negative	ERROR

Tabla 4: Clasificación de los vectores de prueba del dataset Haberman's Survival 1958 por Hold-Out estratificado 80-20

La matriz de confusión correspondiente es la siguiente:

0	2
0	4

Tabla 5: Matriz de confusión *para el dataset Haberman's Survival 1958 por Hold-Out estratificado 80-20*

Total de verdaderos Positive: $0 + 2 = 2$

Total de verdaderos Negative: $0 + 4 = 4$

Parte 3

Este dataset se generó a partir del trabajo de Nutt en colaboración con muchos investigadores que estudiaban el cáncer de cerebro. Como resultado de sus investigaciones, generaron un conjunto de 28 patrones de tejido cerebral, de los cuales 14 corresponden a glioblastomas clásicos, mientras que los 14 patrones restantes corresponden a glioblastomas no clásicos [3].

Cada patrón contiene 1070 atributos numéricos, los cuales representan los niveles de intensidad de expresión de los genes.

Vamos a definir la clase Positive como aquellos glioblastomas no clásicos y la clase Negative como aquellos glioblastomas clásicos.

Patrón	**Clase real**	**Clase asignada**	**Resultado**
1	Negative	Negative	ACIERTO
2	Negative	Positive	ERROR
3	Negative	Positive	ERROR
4	Negative	Positive	ERROR
5	Positive	Positive	ACIERTO
6	Positive	Positive	ACIERTO
7	Positive	Positive	ACIERTO
8	Positive	Positive	ACIERTO

Tabla 6: Clasificación de los vectores de prueba del dataset Nutt por Hold-Out estratificado 75-25

La matriz de confusión generada es:

4	0
3	1

Tabla 7: Matriz de confusión *para el dataset Nutt por Hold-Out estratificado 75-25*

Total de verdaderos Positive: $4 + 0 = 4$

Total de verdaderos Negative: $3 + 1 = 4$

Conclusiones

El empleo de la matriz de confusión en el contexto de la clasificación de patrones ha demostrado ser una útil herramienta para entender y visualizar el desempeño de clasificador. A través de las tres partes abordadas en este reporte, se ha evidenciado cómo esta herramienta proporciona una perspectiva clara sobre los aciertos y errores de clasificación en diferentes contextos y conjuntos de datos.

Es esencial destacar la importancia de conocer los tipos de errores que un clasificador comete. Estos errores, clasificados típicamente como False Positive (FP) y False Negative (FN), tienen significados y consecuencias distintas dependiendo de la aplicación. Por ejemplo, en un contexto médico, un FN (predecir que un paciente no tiene una enfermedad cuando realmente la tiene) puede tener consecuencias graves al no proporcionar el tratamiento necesario, mientras que un FP (predecir que un paciente tiene una enfermedad cuando en realidad no la tiene) puede llevar a tratamientos innecesarios y ansiedad en el paciente.

Por lo tanto, la matriz de confusión no solo ofrece una visión global del desempeño del clasificador, sino que también aporta detalles críticos sobre los tipos de errores cometidos, permitiendo una interpretación más informada.

Referencias bibliográficas

[1] *UCI Machine Learning Repository*. (s. f.-c).

https://archive.ics.uci.edu/dataset/43/haberman+s+survival

[2] Yáñez Márquez, C. (2023). RD 06: Matriz de confusión: elementos iniciales [Diapositivas de clase]. Centro de Investigación en Computación.

[3] Nutt, C. L. et al. (2003). Gene expression-based classification of malignant gliomas correlates better with survival than histological classification. Cancer Research, 63(7), 1602–1607.

Anexos

Parte 1

```
import numpy as np

def leer_archivo(nombre_archivo):
    with open(nombre_archivo, 'r') as file:
        data = [list(map(int, line.split())) for line in
file.readlines()]
    return np.array(data)

def calcular_centroide(data):
    return np.mean(data, axis=0)

def distancia_euclidiana(p1, p2):
    return np.linalg.norm(p1 - p2)

def clasificar(vector, centroide1, centroide2):
    distancia_c1 = distancia_euclidiana(vector, centroide1)
    distancia_c2 = distancia_euclidiana(vector, centroide2)

    return 'c1' if distancia_c1 < distancia_c2 else 'c2'

def main():
    # Lectura de los archivos
    entrenamiento_c1 = leer_archivo('entrenamiento_c1.txt')
    print("Datos de entrenamiento_c1:\n", entrenamiento_c1)

    entrenamiento_c2 = leer_archivo('entrenamiento_c2.txt')
    print("\nDatos de entrenamiento_c2:\n", entrenamiento_c2)

    prueba_c1 = leer_archivo('prueba_c1.txt')
    print("\nDatos de prueba_c1:\n", prueba_c1)

    prueba_c2 = leer_archivo('prueba_c2.txt')
    print("\nDatos de prueba_c2:\n", prueba_c2)

    # Cálculo de los centroides
    centroide_c1 = calcular_centroide(entrenamiento_c1)
    centroide_c2 = calcular_centroide(entrenamiento_c2)

    print("Centroide de c1:", centroide_c1)
    print("Centroide de c2:", centroide_c2)
    print()
```

```
    # Clasificación de los vectores de prueba
    resultados_c1 = [clasificar(vector, centroide_c1, centroide_c2)
for vector in prueba_c1]
    resultados_c2 = [clasificar(vector, centroide_c1, centroide_c2)
for vector in prueba_c2]

    print("Resultados para prueba_c1:")
    for i, res in enumerate(resultados_c1):
        print(f"Vector {i+1}: {res}")

    print("\nResultados para prueba_c2:")
    for i, res in enumerate(resultados_c2):
        print(f"Vector {i+1}: {res}")

if __name__ == '__main__':
    main()
```