

### TAREA en clase/casa (reference - pumpkings homework):

Usando alguna herramienta de IA, consultar:

1. Qué es y cómo se obtiene la matriz de confusión.

<https://chatgpt.com/>

La matriz de confusión es una herramienta utilizada en el campo del aprendizaje automático y la estadística para evaluar el rendimiento de un modelo de clasificación. Esta matriz muestra un resumen de la cantidad de predicciones correctas e incorrectas realizadas por el modelo en cada una de las clases.

Aquí hay una descripción de los elementos principales de una matriz de confusión para un problema de clasificación binaria:

- Verdaderos positivos (TP): Las instancias positivas que fueron clasificadas correctamente como positivas por el modelo.
- Falsos positivos (FP): Las instancias negativas que fueron clasificadas incorrectamente como positivas por el modelo.
- Verdaderos negativos (TN): Las instancias negativas que fueron clasificadas correctamente como negativas por el modelo.
- Falsos negativos (FN): Las instancias positivas que fueron clasificadas incorrectamente como negativas por el modelo.

La matriz de confusión se presenta en forma de tabla, donde las filas representan las clases reales y las columnas representan las clases predichas por el modelo.

La obtención de una matriz de confusión generalmente implica realizar predicciones con el modelo en un conjunto de datos de prueba o validación, donde las etiquetas verdaderas son conocidas. Luego, se comparan estas predicciones con las etiquetas verdaderas para calcular los elementos de la matriz de confusión.

En resumen, la matriz de confusión es una herramienta útil para comprender cómo se desempeña un modelo de clasificación en términos de sus errores y aciertos en la clasificación de las diferentes clases.

2. Analizar el resultado de la matriz de confusión del anterior ejemplo/ejercicio.

Importamos las funciones y clases necesarias de `scikit-learn` para cargar el conjunto de datos, dividirlo en conjuntos de entrenamiento y prueba, entrenar el modelo de regresión logística y calcular las métricas de evaluación.

### **Carga y división del conjunto de datos:**

Utilizamos `load_digits()` para cargar el conjunto de datos de dígitos escritos a mano. Este conjunto de datos contiene imágenes de dígitos del 0 al 9.

Dividimos los datos en conjuntos de entrenamiento (`x_train, y_train`) y prueba (`x_test, y_test`) utilizando `train_test_split()`. El conjunto de prueba representará el 20% del conjunto de datos original.

### **Entrenamiento del modelo:**

Creamos un objeto de modelo de regresión logística con `LogisticRegression()`. Establecemos `max_iter` en un número grande para asegurarnos de que el modelo converja.

Luego, ajustamos (entrenamos) el modelo utilizando los datos de entrenamiento (`x_train, y_train`) mediante `fit()`.

### **Predicciones:**

Utilizamos el modelo entrenado para hacer predicciones en el conjunto de prueba (`x_test`) utilizando `predict()`.

### **Cálculo de métricas de evaluación:**

Calculamos varias métricas de evaluación del modelo, incluyendo:

Exactitud (`accuracy_score`): Proporción de predicciones correctas sobre el total de predicciones.

Precisión (`precision_score`): Proporción de predicciones positivas correctas sobre el total de predicciones positivas.

Recuperación o sensibilidad (`recall_score`): Proporción de instancias positivas que fueron correctamente identificadas por el modelo.

Puntuación F1 (`f1_score`): Media armónica entre precisión y recuperación.

Utilizamos el parámetro `average='weighted'` para calcular estas métricas, lo que significa que se toma en cuenta el desequilibrio de clases.

### **Impresión de métricas:**

Imprimimos las métricas de evaluación para mostrar el rendimiento del modelo en el conjunto de prueba.

En resumen, este código carga un conjunto de datos de dígitos escritos a mano, entrena un modelo de regresión logística sobre él, realiza predicciones en un conjunto de prueba y luego calcula y muestra varias métricas de evaluación del modelo.

### 3. Obtener las formulas de TP, TN, FP, FN, precision, recall, F1-score, support.

```
from sklearn.metrics import confusion_matrix
```

```
import numpy as np
```

```
# Datos de ejemplo (clasificación binaria)
```

```
y_true = np.array([1, 0, 1, 1, 0, 1, 0, 0, 1, 0])
```

```
y_pred = np.array([1, 1, 0, 1, 0, 1, 1, 0, 1, 1])
```

```
# Calculando la matriz de confusión
```

```
conf_matrix = confusion_matrix(y_true, y_pred)
```

```
# Extrayendo los valores de la matriz de confusión
```

```
TN = conf_matrix[0, 0]
```

```
FP = conf_matrix[0, 1]
```

```
FN = conf_matrix[1, 0]
```

```
TP = conf_matrix[1, 1]
```

```
# Calculando las métricas de evaluación

accuracy = (TP + TN) / (TP + TN + FP + FN)

precision = TP / (TP + FP)

recall = TP / (TP + FN)

f1 = 2 * (precision * recall) / (precision + recall)


# Mostrando la matriz de confusión y las métricas de evaluación

print("Matriz de Confusión:")

print(conf_matrix)

print("\nMétricas de Evaluación:")

print("Exactitud:", accuracy)

print("Precisión:", precision)

print("Recuperación o Sensibilidad (recall):", recall)

print("Puntuación F1:", f1)
```

4. Apartir de la matriz de confusión, interpretar los resultados del item 3.

- Los Falsos Positivos son las instancias negativas que fueron clasificadas incorrectamente como positivas por el modelo.
  - Fórmula:  $FP = \text{Número de instancias negativas clasificadas como positivas}$ .

5. Con base en lo anterior, calcular, graficar e interpretar: FPR, TPR, ROC, AUC.