Práctica #1: Evaluación de clasificadores.

José Andrés Reyna Espinoza

Centro de Investigación y de Estudios Avanzados del IPN,
Unidad Tamaulipas.

1 Introducción

La evaluación de clasificadores presenta un reto adicional en el proceso del análisis de datos pues es crucial para determinar la selección de un clasificador adecuado y por tanto su implementación debe ser, en algunos casos, muy concreta.

El presente reporte presenta la implementación de métodos de muestreo, índices de desempeño y evaluación de los resultados simulados de un clasificador, con el fin de observar y comparar los resultados en distintos contextos(dos clases y múltiples clases).

2 Materiales y Métodos

Con el fin de probar los métodos de evaluación de desempeño con resultados de un clasificador, se implementaron funciones para realizar muestre. Además de funciones para calcular medidas f1, de precisión, sensibilidad, especificidad, coeficiente de correlación de matthews. Todos estos métodos fueron implementados en python, en su tercera versión.

Tres archivos de python son los principales: ***evaluation_ methods.py***, ***sampling_ methods.py***, y ***performance_ metrics.py***. Estos tres archivos contienen las funciones principales para realizar las tareas de muestreo, medidas de desempeño y evaluación.

Para realizar una prueba es necesario ejecutar el archivo ***binary_ evaluation.py*** o ***multiclass_ evaluation.py***, para pruebas de evaluación con dos clases y múltiples clases, respectivamente. Estas pruebas

se realizan utilizando tanto holdout como k-folds para realizar el muestreo y evaluación. Los resultados de la ejecución son los valores de desempeño de la evaluación con cada índice, además de un conjunto de gráficas para la matriz de confusión y dos diagramas de barras, uno para el desempeño con holdout y otro para k-folds.

El proceso general para la realización de los experimentos tanto en holdout como en k-folds se describe en la Figura 1, que muestra un diagrama de flujo donde se generaliza el proceso de experimentación.

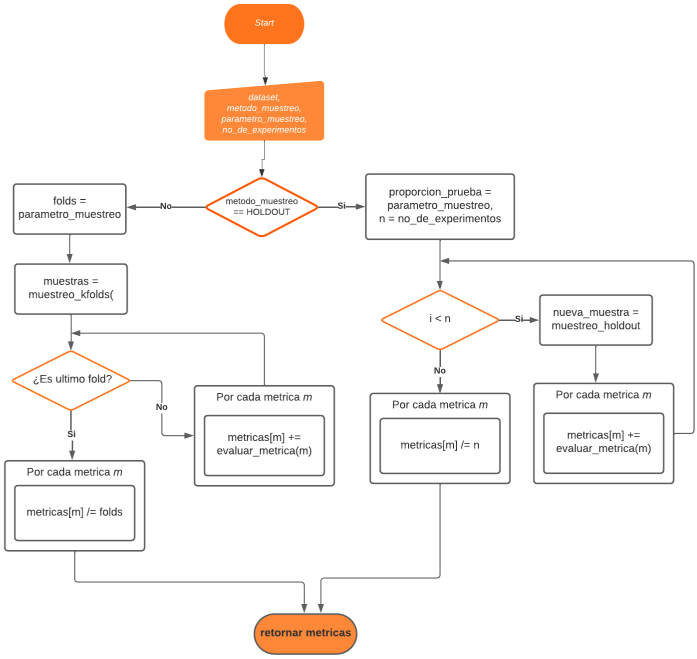

Figura 1 Flujo general de experimentación.

2.1 Muestreo

Para realizar el muestreo estratificado, se utilizan los procedimientos mostrados en la Figura 2 y Figura 3, para el muestreo holdout y kfolds, respectivamente.

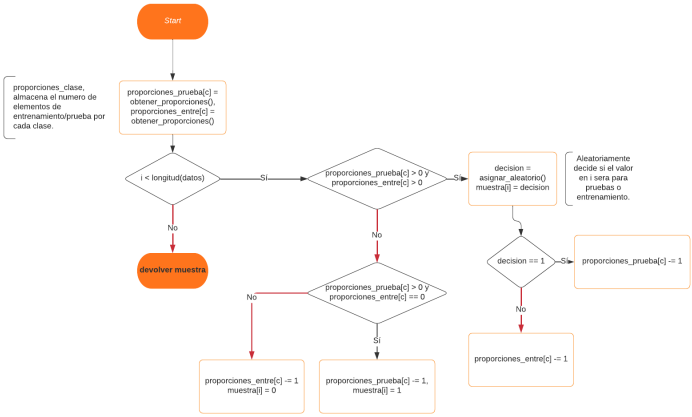

Figura 2 Flujo general de muestreo holdout.

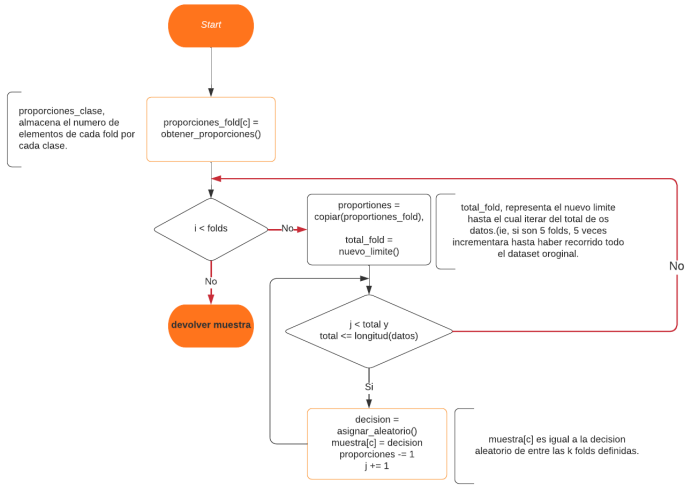

Figura 3 Flujo general de muestreo kfolds.

2.2 Matriz de confusión

La forma en que se genero la matriz de confusión, fue haciendo uso de las herramientas de python para análisis numérico provistas por numpy. Particularmente, haciendo una comparación logica simple entre las clases reales y predichas, y la clase en particular.

El Algoritmo 1 muestra la parte esencial de la función para obtener la matriz de confusión. Particularmente deben hacerse antes de ese algoritmo, un proceso para darle "forma.ª la matriz a construir indicando el identificador para filas y columnas o, en caso de clases binarias, si es negativa o positiva.

Algoritmo 1 Obtener matriz de confusión

```
for i in clases do
    for j in clases do
        matriz[i][j] ⇐ sum((real == i)&(pred == j))
```

2.2.1 Valores de predicción

Para poder aplicar los cálculos de las medidas de desempeño fue necesario obtener antes el total de valores verdaderos positivos(VP), verdaderos negativos(VN), falsos positivos(FP), y falsos negativos(FN). Estos de obtuvieron en los métodos ***get_ bi_ counts()***, y ***get_ tri_ counts()***, para el caso biclase y multiclase respectivamente.

Estos valores de prediccion, se obtienen a partir de la matriz de confusión de dos formas distintas, para las clases binarias y multiclases. En el caso binario simplemente se obtienen los índices de la clase positiva y negativa de acuerdo a como esta en la literatura definido. La Figura 2 muestra como obtener los valores de perdición en una matriz de confusión de dos clases.

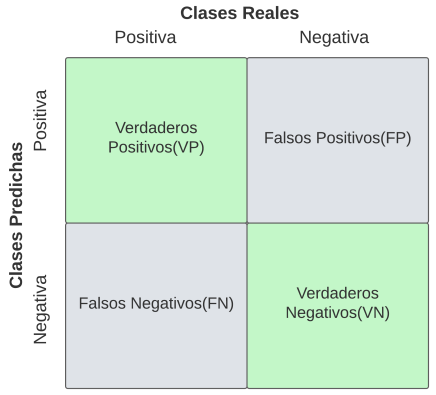

Figura 4 Valores de predicción en matriz de confusión.

Estos valores de predicción son utilizados en las medidas de desempeño, tanto para binarias como multiclase.

2.3 Medidas de desempeño

Las ecuaciones para medir el desempeño se utilizaron para comparar entre los conjuntos de prueba. Se tomaron los datos de prueba obtenidos por el remuestreo(holdout o k-folds) y se compararon con los valores reales en los mismos índices. Se aplicaron las mismas métricas en los de evaluación con dos clases, y con múltiples clases. Cada medida se calcula de forma diferente para la cantidad de clases. Independientemente de la cantidad de clases se

necesita de la matriz de confusión, para obtener los valores de predicción(VP, VN, FP, FN). Debajo se muestran las formulas para el calculo de las medidas de desempeño.

1. Precision
   (a) Binaria: $\frac{VP}{VP+FP}$
   (b) Multiclase: $\frac{1}{K}\sum_k^K \frac{VP_k}{VP_k+FP_k}$ donde K = Numero de clases.
2. Exactitud
   (a) Binaria $\frac{VP+VN}{VP+VN+FP+FN}$
   (b) Multiclase
      (I) Global $\frac{\sum_k^K VP_k}{\sum_i{}_K \sum_j{}_K C_{ij}}$ Donde C_{ij} = Cada celda de la matriz.
      (II) Promedio $\frac{\sum_k^K VP_k+VN_k}{(\sum_i^K \sum_j^K C_{ij})*K}$
3. Sensibilidad
   (a) Binaria $\frac{VP}{VP+FN}$
   (b) Multiclase: $\frac{1}{K}\sum_k^K \frac{VP_k}{VP_k+FN_k}$ donde K = Numero de clases.
4. Especificidad
   (a) Binaria $\frac{VN}{FP+VN}$
   (b) Multiclase: $\frac{1}{K}\sum_k^K \frac{VN_k}{VN_k+FP_k}$ donde K = Numero de clases.
5. Medida F1
   (a) Binaria $\frac{2*VP}{2*VP+FP+FN}$
   (b) Multiclase: $\frac{2*Precision*sensibilidad}{precision+sensibilidad}$
6. Coeficiente de Correlación de Mattews(MCC, por sus siglas en inglés)
   (a) Binaria $\frac{VP*VN-FP*FN}{\sqrt{(VN+FN)(FP+VP)(VN+FP)(FN+VP)}}$
   (b) Multiclase $\frac{\sum_k^K VP_k(\sum_i^K \sum_j^K C_{ij})-\sum_k^K \sum_l^k C_k\dot{C}_l}{\sqrt{((\sum_i^K \sum_j^K C_{ij})^2-(\sum_k^K \sum_l^k (C^T)_k\dot{C}_l)*((\sum_i^K \sum_j^K C_{ij})^2-(\sum_k^K \sum_l^k C_k\dot{(C^T)}_l))}}$

Todas estas medidas de desempeño y la matriz de confusión se encuentran en el código $performance\_metrics.py$.

3 Resultados

Despues de multiples evaluaciones, los resultados muestran que el desempeño de las evaluaciones cuando se realiza el muestreo mediante K-Folds es,

en general, mejor que cuando se utiliza el muestreo Holdout. Como se muestra en las figuras 4 y 5, las graficas son mas altas en los casos donde se utilizo el muestreo k-folds, sin embargo la diferencia puede ser poco notable.

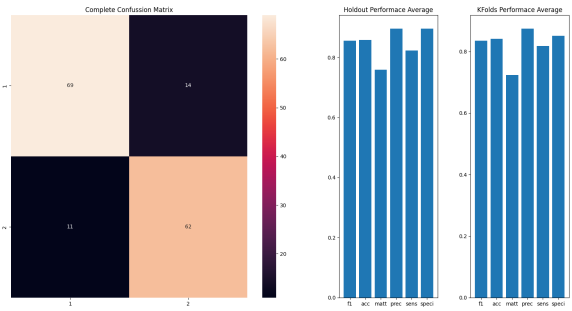

Figura 5 CASO BINARIO: Matriz de confusion datos originales / Desempeño evaluacion con holdout / Desempeño evaluacion Kfolds

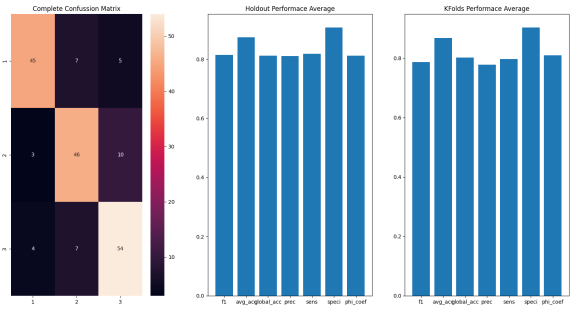

Figura 6 CASO MULTI CLASE: Matriz de confusion datos originales / Desempeño evaluacion con holdout / Desempeño evaluacion Kfolds

4 Conclusiones

El presente reporte muestra el procedimiento y evaluación de un conjunto de datos simulado de los resultados de un clasificador. Estos resultados muestran que el desempeño de ambos algoritmos es bastante similar. En el caso binario de forma general el holdout muestra mejor desempeño. Por el contrario, para la evaluación multiclase, utilizando el muestreo de k-folds parece proporcionar mejores resultados. Este desempeño puede ser debido a la variedad en

el desbalanceo de los folds generados. Al realizar múltiples pruebas siempre tienen un buen promedio, sin embargo si llegan a tener grandes variaciones en algunos casos.

Apéndice A Ejecución de código fuente

La ejecución del código se realiza a través de los archivos ***multiclass_evaluation.py*** y ***binary_evaluation.py*** ejecutandose con python 3.

```
python3 multiclass_evaluation.py
python3 binary_evaluation.py
```

Ambos codigos imoprtan el codigo necesario para ejecutar la evaluacion.