Proyecto 3. Métricas de algoritmos de regresión y clasificación

Santiago López Rojas*
*Maestría en Electrónica, Cartago, Instituto Tecnológico de Costa Rica
email: santiago241996@gmail.com

***Abstract*—This paper presents some of the most use metrics for the clasification and regresion problems using machine learning.**
***Index Terms*—Classification, Regression**

I. Introducción

Luego de seleccionar e implementar un algoritmo de *machine learning* ya sea para la tarea de clasificación o regresión, es necesario definir que tan eficiente es para aproximar el problema. Para esta tarea se utilizan las denominadas **métricas**.

Existen muchas métricas para evaluar modelos de *machine learning* y es importante debido a dependiendo de las que se escojan se pueden comparar el desempeño de los diferentes algoritmos empleados.

En este informe se mencionan algunas de las métricas más usadas, se agrega una breve descripción de cada una de estas y se agrega resultados de ejemplos usando bases de datos triviales, que pueden encontrarse en la biblioteca libre de *sklearn*.

II. Métricas de clasificación

A. Matriz de confusión

Esta es la métrica más sencilla e intuitiva y se usa para clasificaciones binarias y de clases múltiples. Además, es la base para calcular muchas otras métricas como por ejemplo el *Recall* o el *Precision* [2].

Por definición una matriz de confusión C es equivalente a C_{ij}. Esta matriz se puede traducir como el número de observaciones de una clase conocida i y que son predichas como j por el algoritmo de clasificación. El caso más sencillo es el que se muestra en la fig 1 cuando se tiene una clasificación binaria.

En la fig 1, P corresponde a la existencia de la clase en la observación, mientras que N corresponde a la ausencia de esta; y la matiz hace la comparación de lo que se espera que el algoritmo de como resultado (**Actual class**) y lo que se obtiene de él (**Predicted class**). Además, se introducen 4 términos importantes para el tema de clasificación:

		Predicted class	
		P	**N**
Actual class	**P**	**TP**	FN
	N	FP	**TN**

Fig. 1. Matriz de confusión

- **Falsos Positivos** o **False Positives** (FP): Predicciones que dieron positivo cuando en realidad eran negativas (No es deseado).
- **Falsos Negativos** o **False Negatives** (FN): Predicciones que dieron negativo cuando en realidad eran positivos (No es deseado).
- **Verdaderos Positivos** o **True Positives** (TP): Predicciones que dieron positivo cuando en realidad **si** eran positivas (Es deseado).
- **Verdaderos Negativos** o **True Negatives** (TN): Predicciones que dieron negativo cuando en realidad **si** eran negativos (Es deseado).

Con dos clases es bastante intuitivo, no obstante se vuelve ligeramente más complejo cuando se tienen 3 o más clases. En estos casos se puede obtener también que porcentaje de influencia entre clases [2].

Por ejemplo, usando la base de datos *Iris* y el algoritmo de clasificación *C- Support* se obtiene la matriz de confusión 3x3 que se muestra en la figura 2. Donde se ve como las flores Setosas poseen una clasificación perfecta, mientras que las Versicolor no son siempre detectadas y las Virginias poseen falsos positivos por parte de las Versicolor.

B. Recall, Precision, Accuracy y F-score

Recall: Esta métrica nos dice que porcentaje del total de las observaciones que posee una clase fueron detectadas. El calculo se realiza mediante la Ec 1. Esta medida es excluyente e individual por clase y no considera los FP [1].

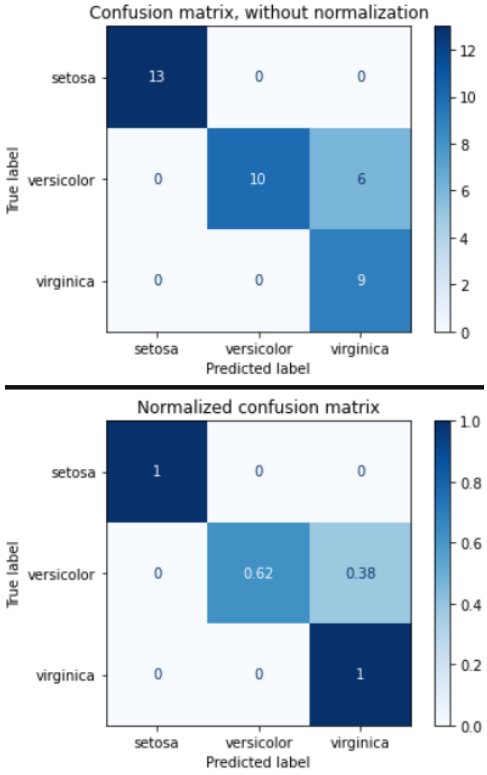

Fig. 2. Matriz de confusión para dataset Iris

$$R = \frac{TP}{TP + FN} \tag{1}$$

Precision: Corresponde a la probabilidad de que una detección positiva corresponda a un Verdadero Positivo. En otras palabras, es la proporción de positivos de una clase, que en realidad corresponden a esa clase. El calculo se realiza mediante la Ec 2. Esta medida es excluyente e individual por clase y no considera los FP [1].

$$P = \frac{TP}{TP + FP} \tag{2}$$

Accuracy: Esta métrica es similar al *recall* pero incluye todos los parametros de la matriz de confusión. Se refiere al número de predicciones correctas sobre el total de predicciones realizadas [2]. El resultado caracteriza al modelo completo, incluyendo todas las clases y se calcula mediante la Ec 3 o la Ec 4.

Se recomienda usar esta métrica cuando se tiene una cantidad balanceada de datos por clase. Por ejemplo si el problema de de clasificación de imágenes de manzana y naranjas y se tiene una cantidad igual de imágenes por clase.

En caso contrario no es recomendable usarlo ya que la clase que posee la mayor cantidad de muestras puede sobreponerse a la o las minoritarias y estar detectando muy bien la clase mayoritaria pero no generalizando el modelo.

$$Accuracy = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(y_{true} = y_{pred}) \tag{3}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{4}$$

F-score: Para facilitar la visualización del *recall* y el *precision* se tiene el F-score que es una relación entre los dos mencionados. El cálculo de esta se realiza con la Ec 5 y al igual que las variables que relaciona el F-score es excluyente e individual para cada clase.

Este cálculo normalmente no es muy útil en muchas situaciones, como por ejemplo cuando el *precision* es muy alto en comparación con el *recall* el *F-score* se verá comprometido y no representará la realidad de la abstracción del modelo.

$$F - score = \frac{2PR}{P + R} \tag{5}$$

Como ejemplo práctico se sigue usando la base de datos Iris y la biblioteca *sklearn* que tiene la opción de obtener todos estos datos en manera simultanea con el comando **classification_report()** o de manera particular, como por ejemplo con el aso del *accuracy* se tiene el comando **accuracy_score()**.

Los resultados del experimento se tienen en la fig 3 obteniendo los resultados esperados teniendo en cuenta la matriz de confusión de la fig 1, teniendo un recall perfecto tanto en las Setosa como en las Virginia y siendo el de las Versicolor las que se ven afectadas.

	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	13
Versicolor	1.00	0.62	0.77	16
Virginica	0.60	1.00	0.75	9
accuracy			0.84	38
macro avg	0.87	0.88	0.84	38
weighted avg	0.91	0.84	0.84	38

Accuracy del modelo: 0.8421052631578947

Fig. 3. Reporte de entrenamiento usando dataset Iris y algoritmo C-Support

C. *ROC Curve y AUC*

Estas métricas nos indican que tanto un algoritmo puede distinguir entre TP de TN en un modelo binario. Siendo una gráfica de TPR (Recall) vrs FPR, esta última no se ha definido aún, debido a que no es muy utilizada y está dado por la Ec 6 [2]. La fig 4 muestra los resultados de esta medición según la capacidad de un algoritmo de aproximar un modelo [4].

$$FPR = \frac{FP}{TN + FP} \tag{6}$$

En la fig 4 a) representa un algoritmo que logra una clasificación perfecta, separando todos los TP de los TN. La figura b) muestra lo que ocurre cuando se da un cierto porcentaje de de FP y por ende FN. Por otra parte, la figura c) muestra un algoritmo que no logra hacer ninguna separación entre TP y TN, teniendo resultados meramente aleatorios [3].

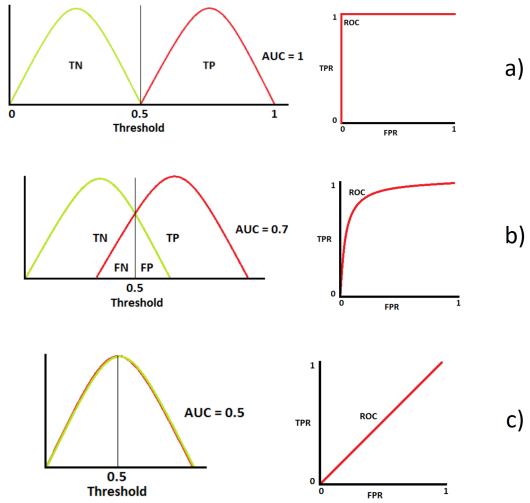

Fig. 4. ROC Curve

Aunque la gráfica ROC es visualmente fácil de leer, el parámetro del área bajo la curva (AUC). es el parámetro utilizado para comparar el desempeño de diferentes algoritmos sobre un mismo dataset. En la figura 4 se muestra como el caso ideal es un AUC cercano a 1, mientras que un AUC cercano a 0.5 es no deseado.

Como ejemplo se usa el set de datos *Breast Cancer*, al ser un problema con resultado binario es posible el ROC y el AUC de manera directa. Se compara el desempeño usando los algoritmos KNN y C-Support. Obtenido que el C-Support es ligeramente mejor para generalizar el problema, como se muestra en la fig 5 [3].

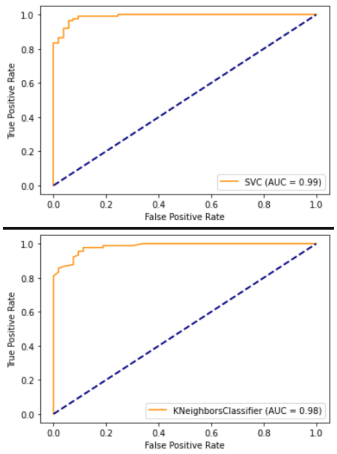

Fig. 5. ROC Curve y AUC para ejemplo con dataset Breast Cancer

III. Métricas de Regresión

Max Error: Esta métrica da como resultado el error residual máximo. En otras palabras, captura el error de la peor predicción dada por el modelo. Se intenta siempre que este error esté lo más cercano a cero posible.

La Ec 10 caracteriza el funcionamiento de esta métrica. Con y correspondiendo al valor esperado y $\ddot{y}$ el valor predicho por el modelo. Esta métrica se usa cuando es necesario que el modelo no supere un error establecido en ninguna muestra individual, aunque el error de MSE, MSLE u otros si esten bajo el límite deseado [5].

$$MaxError(y, \ddot{y}) = max(|y - \ddot{y}|) \tag{7}$$

Mean absolute Error: Esta es una métrica ponderada que da la diferencia entre el valor real y el predicho en valor absoluto. Se calcula mediante la Ec 8. Cabe destacar que MAE es un valor lineal, lo que significa que todos los errores individuales por predicción tienen el mismo peso en el resultado del calculo de la métrica.

Por su naturaleza, con MAE las predicciones con un error muy grande no serán tan penalizadas como si lo hacen otras métricas que se discuten más adelante. En otras palabras, esta medida es usada cuando no es tan importante penalizar a los *outliers* [6] [5].

$$MAE = \frac{1}{n_s} \sum_{i=0}^{n_s-1} |y_i - \ddot{y}_i| \tag{8}$$

Mean Squared Error: Esta es una métrica ponderada que da la diferencia cuadrática entre el valor real y el predicho en valor absoluto. Se calcula mediante la Ec 9. Al ser un cálculo cuadrático este error penaliza los errores individuales grandes pero al mismo tiempo premia los errores individuales pequeños.

Esta métrica es usada cuando es importante notar resultados con errores muy altos pero aislados, que en otras métricas como por ejemplo la MAE no se verían tan representados. Esta medida no se suele usar para comparar el desempeño entre diferentes algoritmos [6].

$$MSE = \frac{1}{n_s} \sum_{i=0}^{n_s-1} (y_i - \ddot{y}_i)^2 \tag{9}$$

Median Absolute Error: Esta métrica nos da el valor de la mediana del error absoluto, esta medición no se ve alterada por los valores de lo *outliers* a menos que haya muchos y es importante para determinar el error sobre el que oscila el modelo, aunque esto no implica que la mayoría de los errores sean este valor o cercano [6] [5].

$$MedAE(y, \ddot{y}) = median(|y_1 - \ddot{y_1}|, ..., |y_n - \ddot{y_n}|) \tag{10}$$

R^2 **Score:** Teniendo un valor entre $-\infty$ y 1, esta métrica indica que tan bien las variables independientes en la regresión logran aproximar el resultado, siendo un 1 como una aproximación perfecta y cercano a cero como una predicción aleatoria. Este parámetro se calcula mediante la Ec 11 [6].

$$R^2 = 1 - \frac{\sum_{i=0}^{n} (y_i - \ddot{y_i})^2}{\sum_{i=0}^{n} (y_i - \bar{y_i})^2} \tag{11}$$

Para ejemplificar el uso de todas estas métricas se usa la base de datos *boston house-prices* que al igual que en los casos anteriores los da la biblioteca *sklearn*. El objetivo es predecir valor de viviendas de entre 5 mil y 50 mil dolares. Se confeccionan 4 modelos de regresión usando diferentes algoritmos y los resultados en todos los casos se muestran en la tabla I.

- **AdaBoost:** Con esta aproximación se logra el mejor MAE en comparación con todos los otros algoritmos, además de que presenta el mejor MSE lo que implica que posee pocos *outliers* aunque no logra el mejor MAD. Finalmente posee un R^2 alto, lo que implica una buena generalización.

- **Decision Tree:** Este es el algoritmo que presenta el segundo mejor desempeño, después del AdaBoost. Es el que logra el mejor MAD pero como muestra el MSE presenta una cantidad más significativa de *outliers*, así como más significativos como se ve en el Max Error. No obstante el R^2 es alto y esta aproximación es buena para generalizar el problema.

- **Gaussian Process:** Este modelo es el que en comparación con los dos anteriores presenta los mayores *outliers*, como se ve en los resutlados, el MAD y el MAE no son demasiado superiores a los de sus predecesores, pero el verdadero problema es el MSE, problema que se ve reflejado en el valor del R^2.

- **KKN:** Este es el algoritmo que tiene el peor desempeño reflejándose directamente en el R^2 teniendo un valor bajo y no aceptable para decir que el modelo logra una generalización.

TABLE I
RESULTADOS REGRESIÓN

Algoritmo	**Max Error**	**MAE**	**MSE**	**MAD**	R^2
AdaBoost	6.83	2.26	7.69	2.06	0.89
Decision Tree	13.40	2.63	13.60	1.80	0.80
Gaussian	19.35	3.28	22.03	2.27	0.68
KNN	24.77	3.99	34.21	2.77	0.5

IV. CONCLUSIONES

Como se ve en los ejemplos y en la teoría, existen muchas métricas para determinar la efectividad de un modelo sobre otro para generalizar un problema; pero de igual manera, se observa como no existe un único parámetro que indique independientemente y con total certeza cual es el mejor algoritmo a utilizar.

Por lo contrario, se deben usar diferentes combinaciones de métricas para poder llegar a conclusiones, y además se tiene que tener claro cual es el objetivo de la generalización del problema para saber cuales métricas se deben optimizar.

REFERENCES

[1] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org.
[2] M. Sunasra, Performance Metrics for Classification problems in Machine Learning. Medium, 2017. https://medium.com/@MohammedS/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b
[3] C. Chan, What is a ROC Curve and How to Interpret It. DisplayR, 2018. https://www.displayr.com/what-is-a-roc-curve-how-to-interpret-it/
[4] S. Narkhede, Understanding AUC - ROC Curve. Medium, 2018. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5
[5] A. Swalin, Evaluating Machine Learning Models — Part 1. Medium, 2018. https://medium.com/usf-msds/choosing-the-right-metric-for-machine-learning-models-part-1-a99d7d7414e4
[6] 3.3.4. Regression metrics. Sklearn, 2020. https://scikit-learn.org/stable/modules/model_evaluation.html#regression-metrics