Análisis y Reporte sobre el desempeño del modelo

Leonardo Alvarado Menéndez
A01705998

Introducción:

Se realizó un modelo de regresión logística para analizar los datos recibidos de imágenes de pasas, y con estos poder clasificar si las pasas eran de tipo Kecimen o Besni.
Los datos usados fueron:

- Area
- Perimeter
- MajorAxisLength
- MinorAxisLength
- Eccentricity
- ConvexArea
- Extent

Para mayor información o sobre los datos, puede consultar en la siguiente página: *https://archive.ics.uci.edu/ml/datasets/Raisin+Dataset#*

Método empleado

El modelo de regresión logística empleado, para este análisis fue LogisticRegression de Sklearn[1], donde usando como hyperparameters:

- penalty = 'l1'
- solver = 'saga'

Y corriendo nuestro modelo, separando el dataset como train y test, obtenemos un accuracy de ***0.5111*** y un MSE de ***0.4888*** en nuestros datos test y un accuracy de ***0.4962*** y un MSE de ***0.5037*** en los datos de train.

En la Fig 1, se pueden ver los resultados obtenidos en el test:

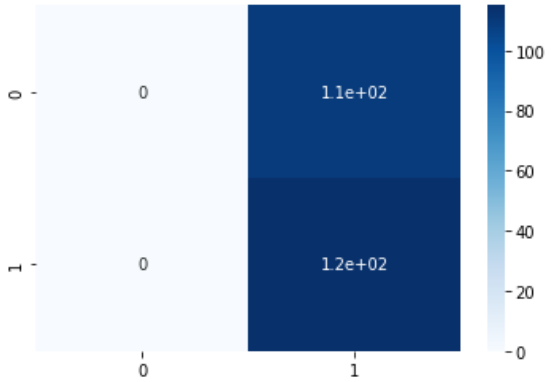

Fig 1: Matriz de confusión modelo1

En esta matriz, el 1 representa las pasas Bensi y el 2 a las Kecimen, como podemos ver, el modelo predice que todas las pasas son de tipo 1 (Bensi), cuando realmente solo 120 de estas son de este tipo y las 110 restantes son de tipo 2 (Kecimen).

Con esta información, podemos ver que nuestro modelo está "**underfitting**" , ya que tenemos un error muy alto tanto en train como en test, por lo que tenemos un **bias** alto, debido a esto, no podemos conocer la **varianza** de nuestro modelo y este únicamente termina el train ya que tiene una limitante de 100 epochs.

Mejora

En los casos de underfitting, para mejorar nuestro modelo tenemos las siguientes opciones:

1. Transformamos los datos
2. Agregamos más parámetros
3. Modelo más complejo

Para la mejora del modelo, se cambió el solver de 'saga', por 'liblinear' el primero es la extensión de 'sag' que también permite el penalty 'l1' y que en general, debería entrenar más rápido que 'sag', el

1 https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
2 https://towardsdatascience.com/dont-sweat-the-solver-stuff-aea7cddc3451

cual es una variación de los enfoques de descenso de gradiente y gradiente incremental agregado que utiliza una muestra aleatoria de los valores de gradiente anteriores. Y el segundo 'liblinear' utiliza un algoritmo de descenso por coordenadas. El descenso por coordenadas se basa en la minimización de una función multivariante mediante la resolución de problemas de optimización univariantes en un bucle .[2]

Este segundo es mejor para el dataset, debido a que funciona mejor con una alta dimensionalidad en los datos.

Resultado

Con estos cambios, volvemos a correr nuestro modelo, y en esta ocasión obtenemos un accuracy de ***0.8933*** y un MSE de ***0.1066*** en nuestros datos test y un accuracy de ***0.8577*** y un MSE de ***0.1422*** en los datos de train.

En la Fig 2, se pueden ver los resultados obtenidos:

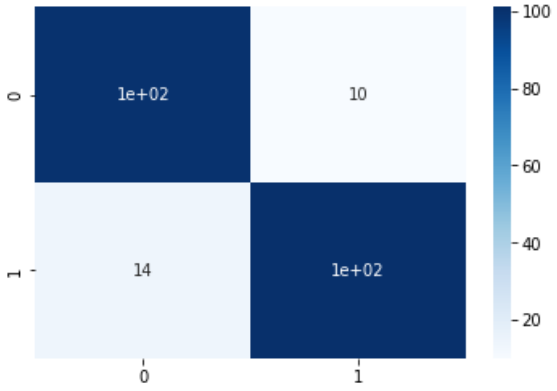

Fig 2:Matriz de confusión modelo 2

En esta ocasión, podemos ver que el modelo únicamente predice mal 10 pasas que llama como tipo 1 cuando realmente son tipo 0 y 14 como tipo 0 cuando son tipo 1.

Conclusión

Comparando ambos modelos, podemos ver una mejora del **174.78%** en accuracy y una disminución en el error de ***0.5111*** a ***0.1066***, por lo que contamos con un mejor modelo, el cual, aunque no llega a tener una precisión de más de 90% ya genera una mejor predicción.

1 https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
2 https://towardsdatascience.com/dont-sweat-the-solver-stuff-aea7cddc3451