# BITACORA DE DESARROLLO - APRENDIZAJE DE MAQUINA

**JONATAN GALLO – JUAN CAMILO TORRES - JUAN DAVID CLAVIJO**

**MAESTRIA EN INTELIGENCIA ARTIFICIAL**

**MAESTRIA EN INGIENERIA DE SISTEMAS Y COMPUTACION**

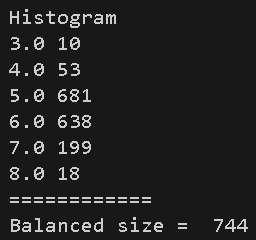
**2025**

## INTRODUCCIÓN

El objetivo de este proyecto fue implementar K-Fold y Leave On Out desde cero para clasificar cuatro datasets. Durante el desarrollo, nos enfrentamos a varios desafíos para construir los datasets de manera adecuada para que los modelos pudieran evaluar correctamente los datos y la validación de los resultados.

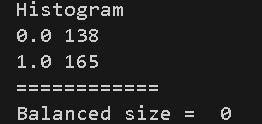
## DESARROLLO

* 1. Red wine quality:



* 1. Heart Disease:

Este dataset lo normalizamos y le aplicamos shuffle a partir del histograma. No había mucho problema con los datos y estaba completa, con lo que procedimos a avanzar con la normalización y la mezcla de los datos.



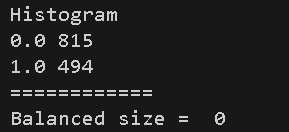
Generamos el data con los valores normalizados en el archivo heart\_balanced.csv

* 1. Titanic:

Para generar completamente el tercer dataset de Titanic. Usamos el código de Python: join\_columns\_handle\_nulls.py. Donde revisamos el gender\_submission.csv y lo unimos a test.csv revisando las ids de los integrantes y colocándole el survive al final de la columnadel archivo test.csv, usando la función unirSurv\_test, guardamos los valores en un csv con la data llamado: data\_con\_sobrevivientes.csv y este csv lo evaluamos con el train, revisando que la columna de survive teníamos que pasarla al final para que quedara como (*y).* Con la función unificar\_csvs, unificamos los dos datasets y cambiamos el survive a la posición final.

Los datos faltantes en edad, los generamos con la media de la edad de las personas que tenían edad en el data y le agregamos esa media en números flotantes, porque había edades de 28.5. Pensamos en redondear las edades, pero finalmente decidimos dejarlos como aparecían para no alterar los datos originales. En cabina, como también teníamos datos faltantes, colocamos DES de desconocida. Al final teníamos los datos completos que se guardaron como data\_completa.csv y este csv ya quedo listo para normalizar, aplicarle shuffle en balance\_normalize\_titanic.py se realizó dicha normalización y mezcla de datos y finalmente lo ejecutamos con los modelos.

Revisamos en el histograma y nos quedo con que no habían sobrevivido 815 lo que sobrepasaba por mucho a los que si, 494.



* 1. TelcoCustomer:

2.5 K-Fold:

En la implementación de K-Fold tuvimos que revisar como se separaba el grupo que generaba de lo que nos quedaba por fuera del grupo, encontramos que para realizar la función de K-Fold, teníamos que dividir la muestra total enteramente entre el grupo de divisiones que asignábamos. Y si quedaba un sobrante empezar a introducirlos entre los grupos iniciales de la división. Es decir, si teníamos una muestra de 10 y dividíamos entre 3 nos daba 3.33 ese 0.33 teníamos que ubicarlos en los grupos iniciales para que, en el grupo final, solo quedara el sobrante y no se alterara o faltaran datos en los grupos según la división asignada a los datos.

## CONCLUSIONES:

## REFERENCIAS:

* **Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009).** Red wine quality dataset. *Kaggle.* <https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009>
* **PAVAN KUMARD - Heart Disease UCI, (2021)** <https://www.kaggle.com/code/mragpavank/heart-disease-uci/input>
* **Titanic – Machine Learning from Disaster ()-** <https://www.kaggle.com/competitions/titanic/data>
* **Telco Customer Churn, (2018).**

<https://www.kaggle.com/code/farazrahman/telco-customer-churn-logisticregression>

* Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2