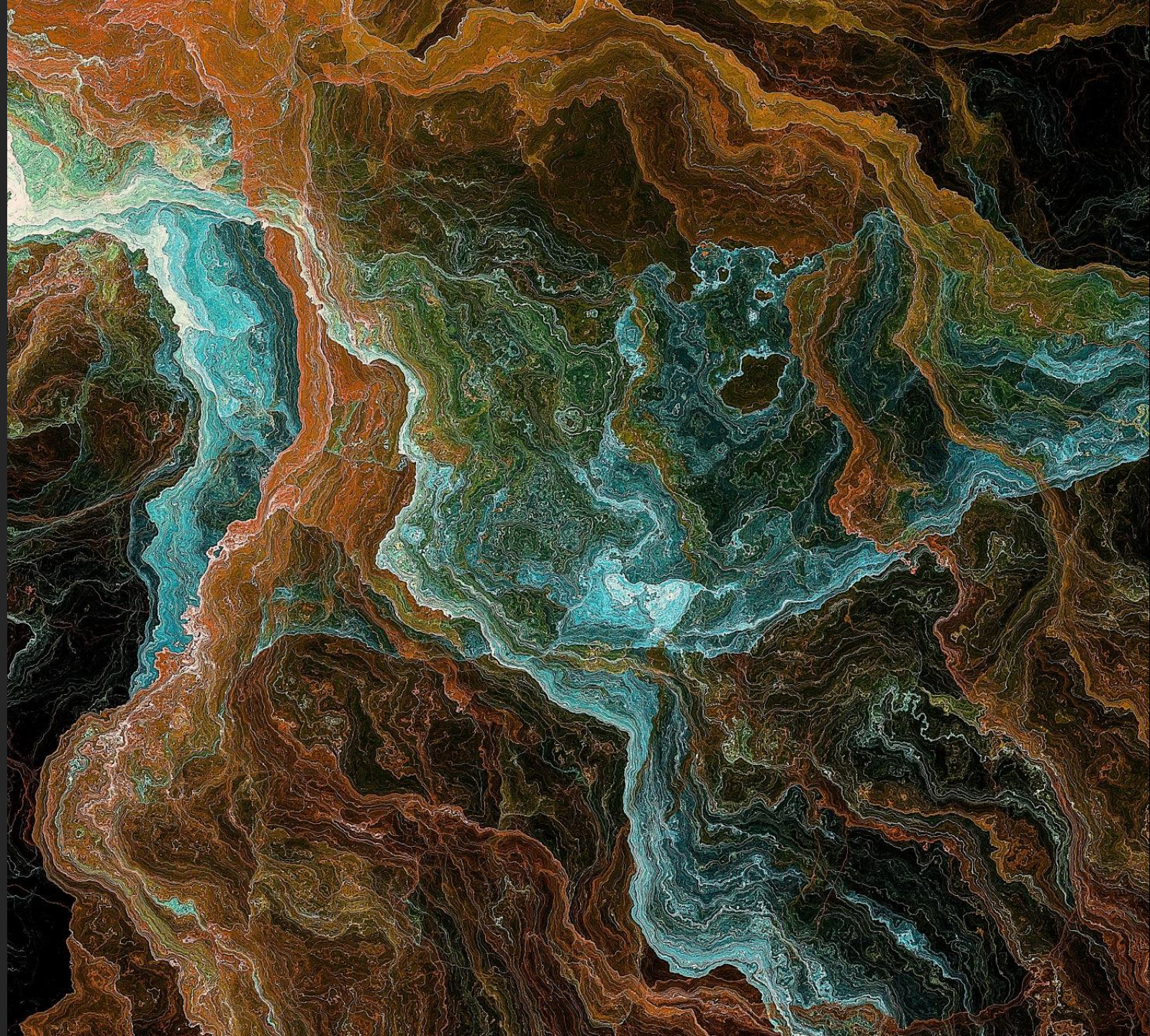
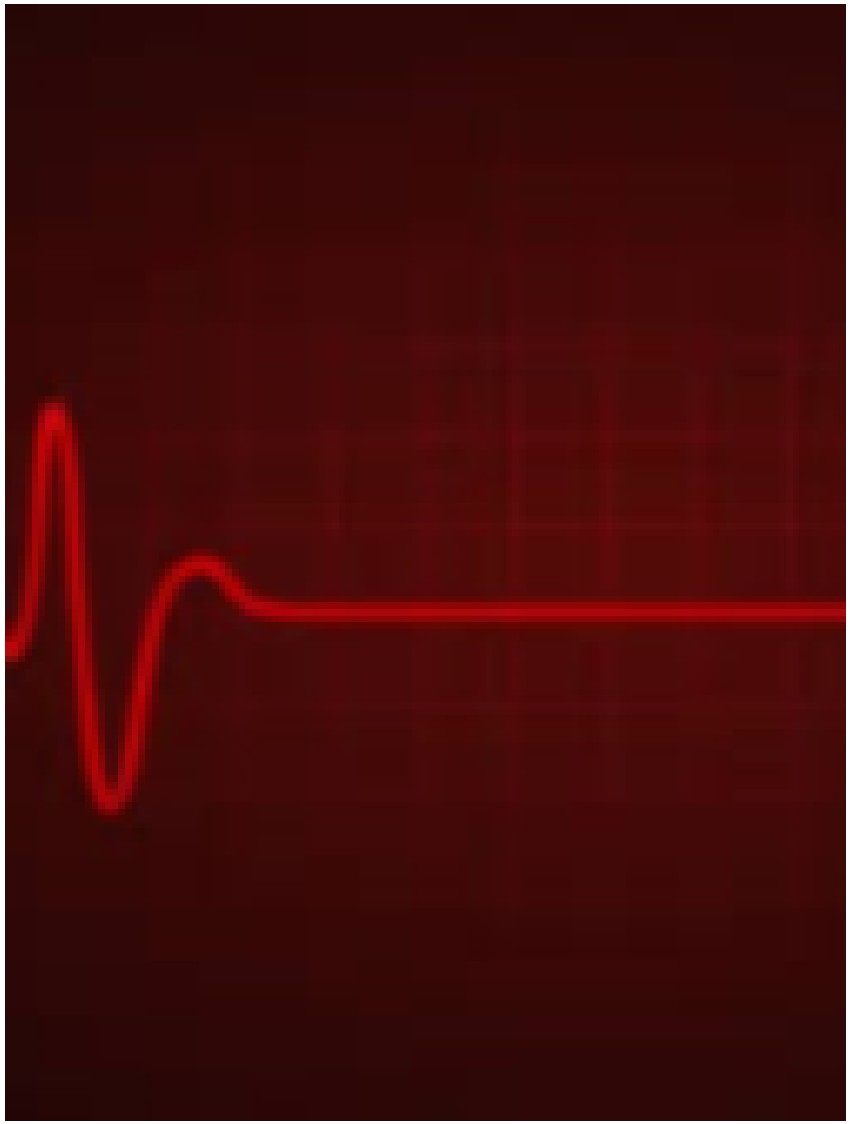
Machine Learning

Supervised learning determining survivability of patients with HCC diagonosis

Beatriz Seabra 202303729
Pedro Amaro 202106568
Rui Jorge 202305059





Introduction to the Machine Learning Problem:

This assignment involves developing a machine learning pipeline capable of determining patient survivability at 1 year following the diagnosis of Hepatocellular Carcinoma (HCC), a form of liver cancer, using clinical data obtained from the Coimbra Hospital and University Center (CHUC).

To know ahead of time the possible outcome of a patient is crucial for personalized treatment planning and improving overall survival rates.

By the end of this project, we should have a sophisticated machine learning pipeline adept at accurately forecasting patient survivability.

.

Related Work

Hepatocellular carcinoma (HCC) is among the leading causes of cancer incidence and death, and despite decades of research and development of new treatment options, the overall outcomes of patients with HCC continues to remain poor. More recently we have been seeing an explosive growth in the application of artificial intelligence (AI) technology in medical research, with the field of HCC being no exception.

These articles contain crucial information concerning the aspects of creating a machine learning using supervised learning in patients with HCC, being, that way, our base for the assignment.

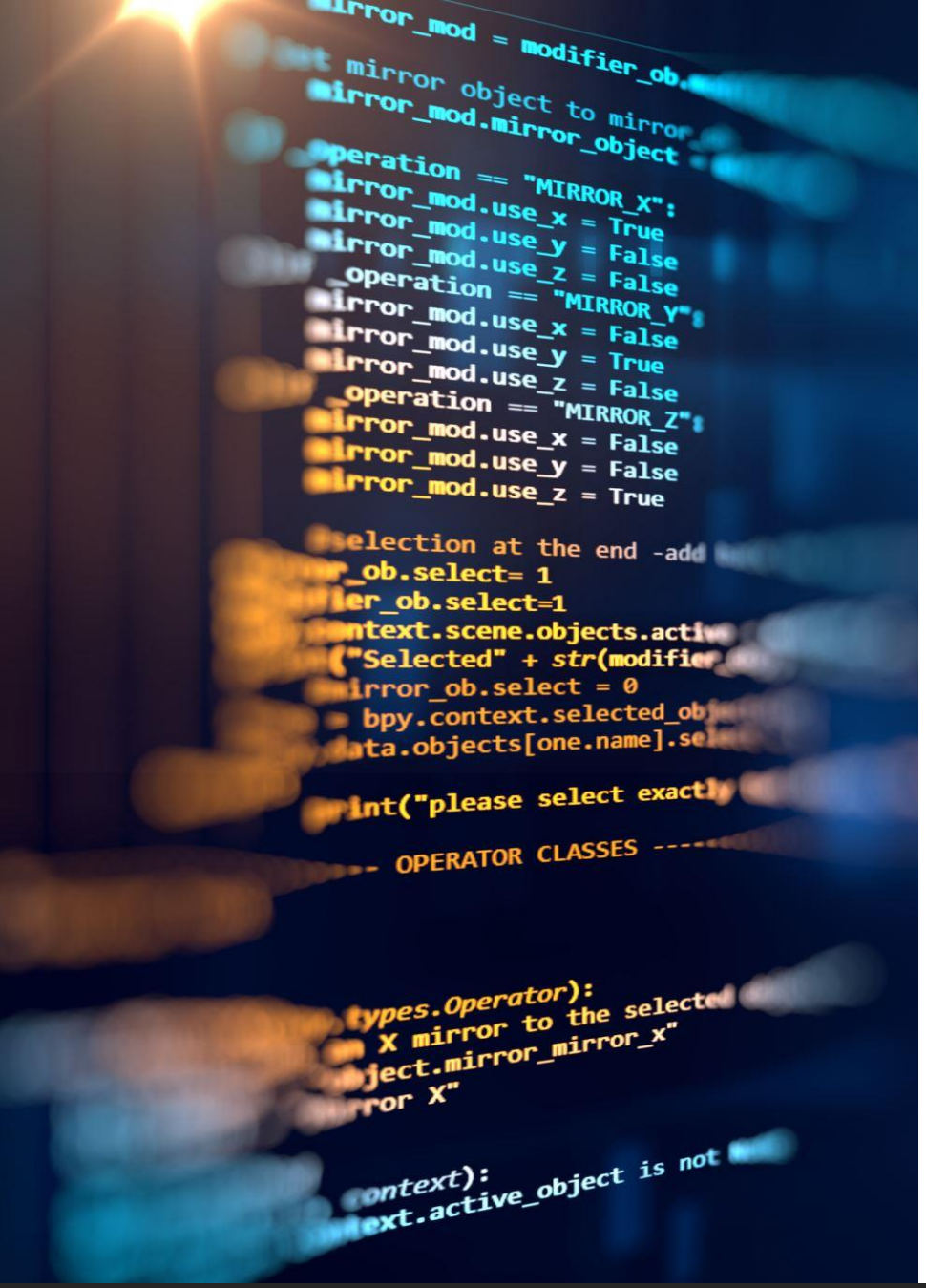
https://www.ajmc.com/view/machine-learning-model-predicts-hepatocellular-carcinoma-risk-in-patients-with-masld
w.ncbi.nlm.nih.gov/pmc/articles/PMC8727204/

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8727204/

https://fortuneonline.org/articles/supervised-machine-learning-techniques-for-the-prediction.pdf

https://www.nature.com/articles/s41598-024-51265-7

https://www.sciencedirect.com/science/article/pii/S2405844023096664



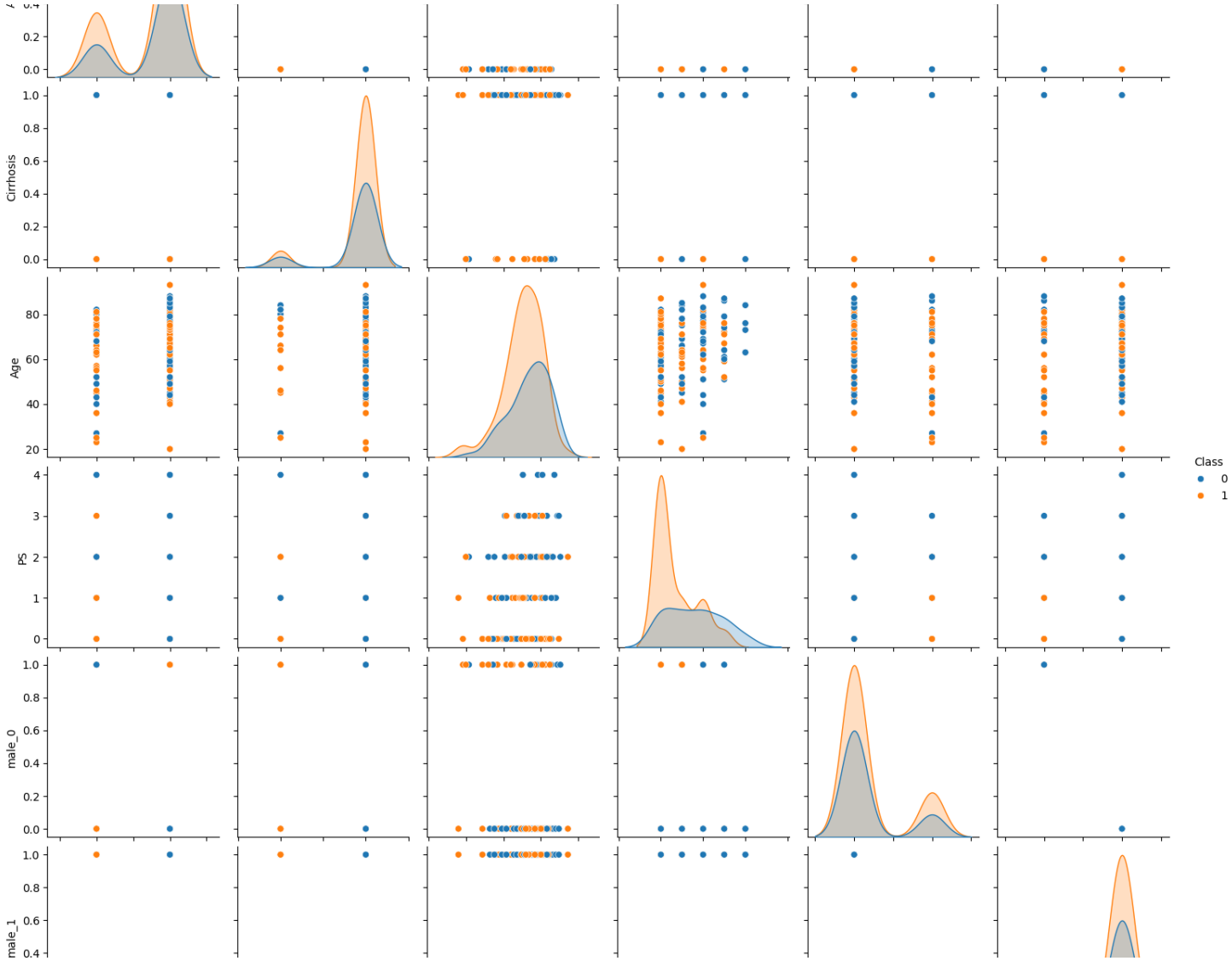
Tools and Algorithms

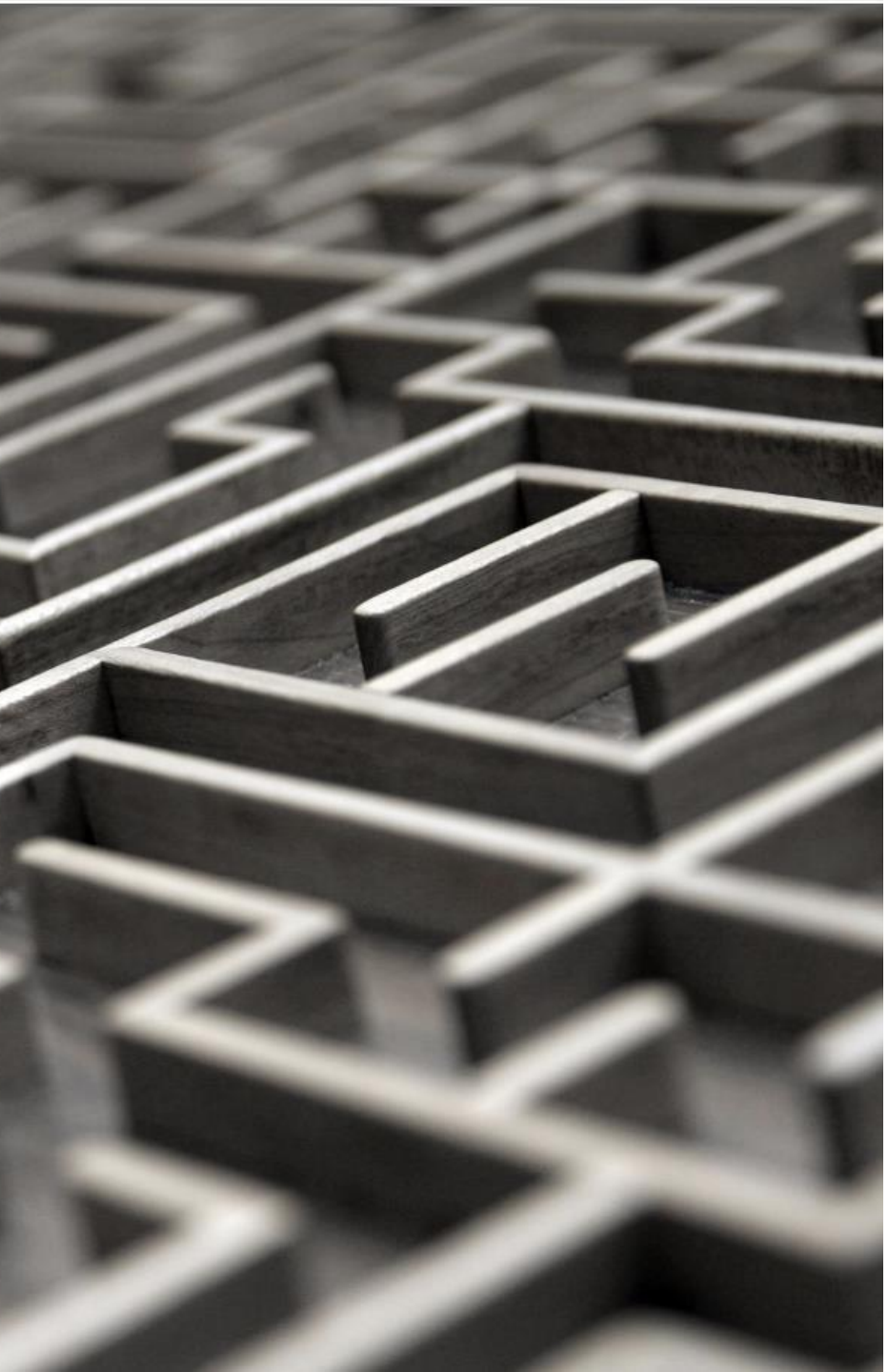
For this assignment, we are going to work with Jupyter notebooks as our development environment. Our implementation will be in Python, due to its vast array of resources for machine learning. We will focus on and work with core libraries such as pandas, numpy/scipy, scikit-learn, and matplotlib/seaborn within the Jupyter environment. Our workflow will involve a variety of tools and algorithms, starting from data exploration and preprocessing techniques like exploratory data analysis, feature preprocessing, and engineering. For supervised learning, we will deploy decision trees and K-nearest neighbors (KNN) algorithms using Scikit-learn.

Data preprocessing

To effectively process our data, we first performed the following preprocessing steps:

- **Replace Missing/Null Values**: replace any missing or null entries with imputacion and Knn to ensure data integrity.
- **identified Outliers**: Detected and handled outliers to maintain data consistency.
- **verified Duplicate and Ambiguous Values**: Ensured the data did not contain duplicates or ambiguous entries.
- **transformed Categorical Values into Continuous**: Converted categorical data into a continuous format for better analysis.





Methods and approaches used

Classifiers:

- Decision tree
- k-Nearest Neighbors
- Random Forest
- Gradient boosting
- Naive bayes
- Adaboost

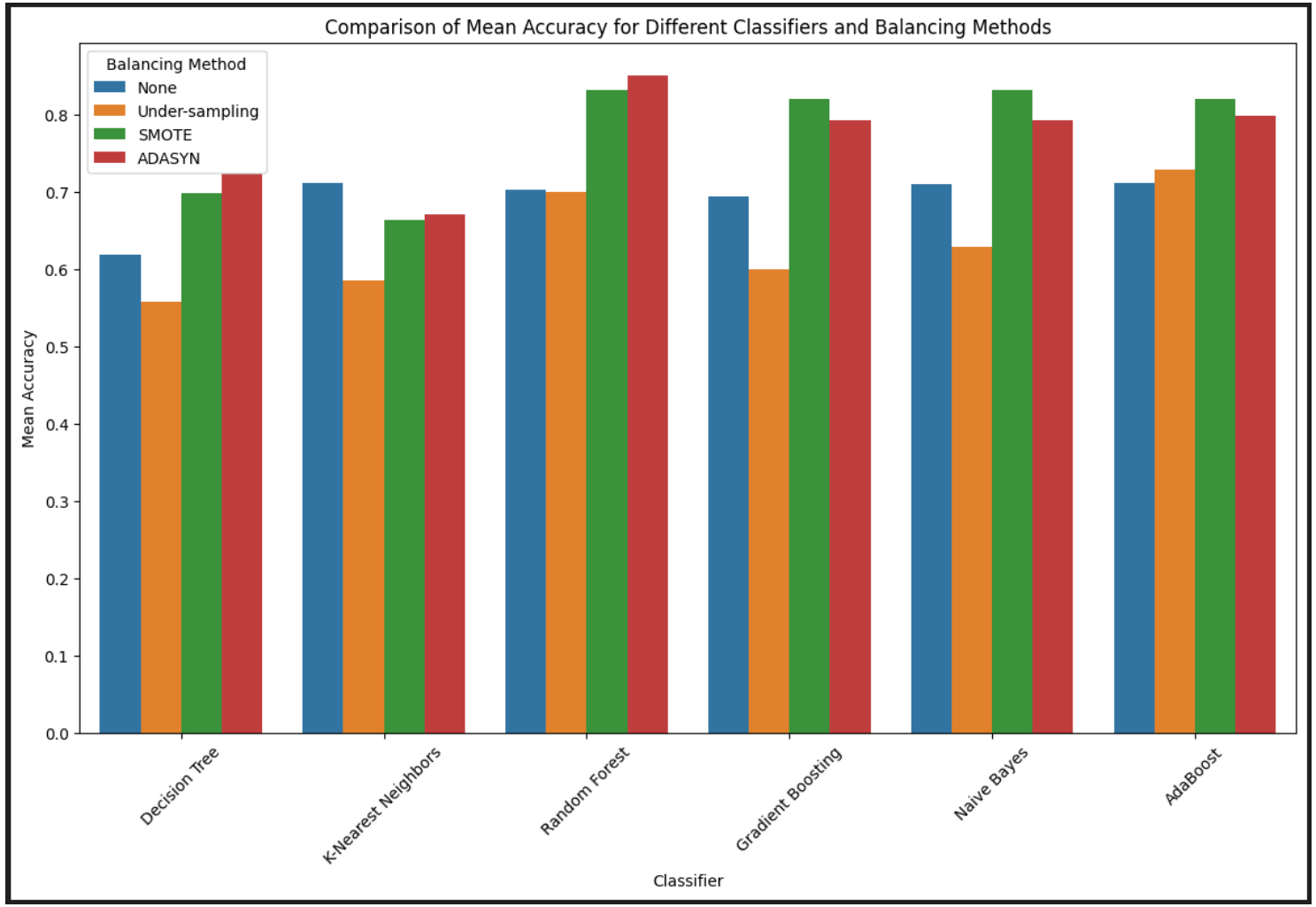
Balancing methods:

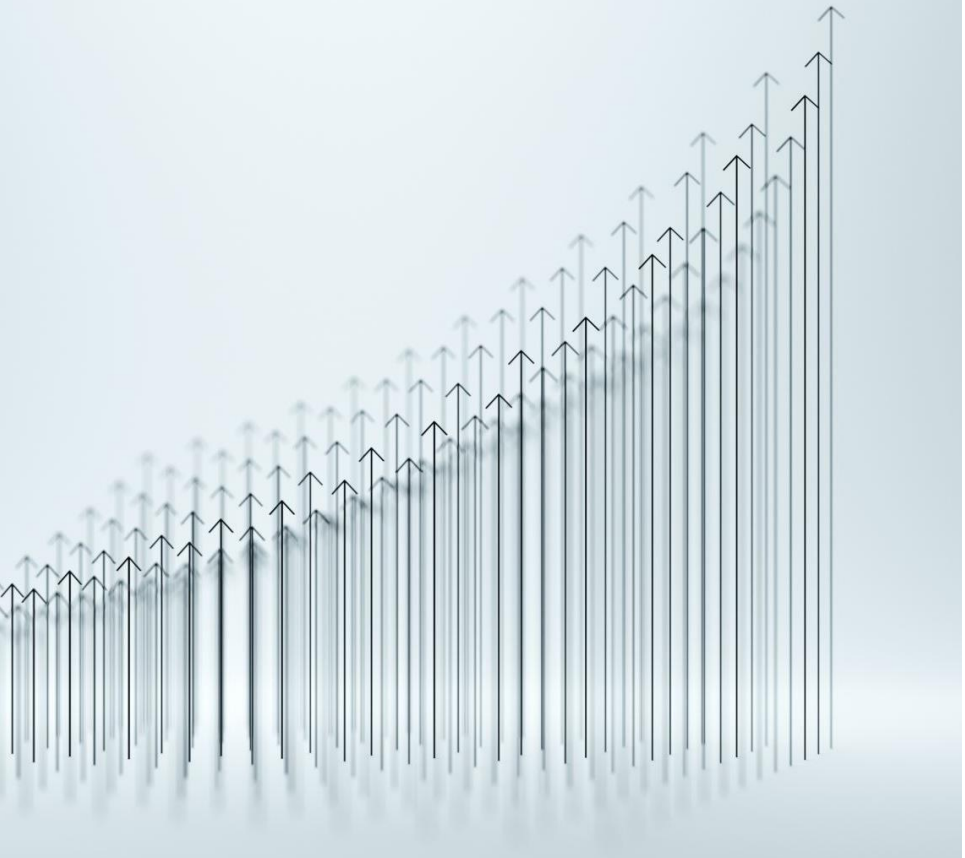
- None
- Under-sampling
- SMOTE
- ADASYN

Models comparation

Best scores regarding mean accuracy:

- Random Forest with SMOTE: 0.826
- Gradient Boosting with SMOTE: 0.814
- Random Forest with ADASYN: 0.834
- AdaBoost with SMOTE: 0.821

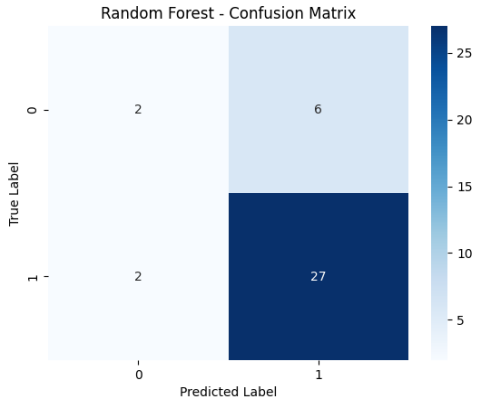


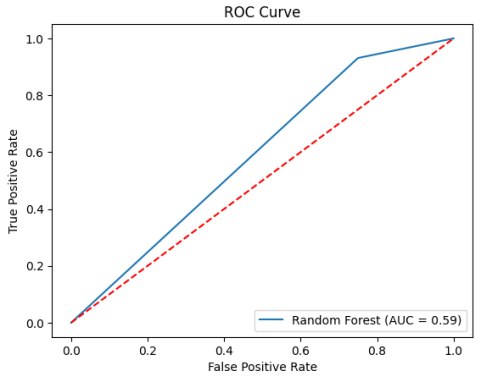


Developed models

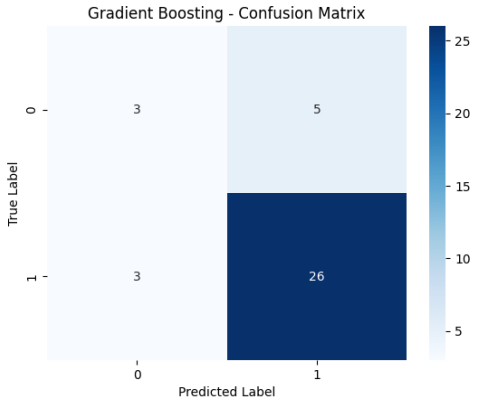
- Firstly, the Random Forest models show the best overall performance, especially with the ADASYN and SMOTE balancing techniques. With SMOTE, it achieved a mean precision of 0.859 and a mean accuracy of 0.823, while with ADASYN it achieved a mean precision of 0.875 and a mean accuracy of 0.834 (the highest result we managed to achieve). Other models that provided good results were Gradient Boosting with SMOTE that achieved a mean precision of 0.858 and a mean accuracy of 0.814 and AdaBoost with SMOTE, which achieved a mean precision of 0.872 and a mean accuracy of 0.821.
- With this results we saw that data balancing significantly impacts model performance. Comparing results without balancing to those using techniques such as "SMOTE" and "ADASYN" shows substantial variations in both mean precision and accuracy. Some models are more sensitive to balancing. These results highlight the importance of data balancing when developing and evaluating classification models, showing that the choice of the ideal model may depend on the balancing technique used and the specific characteristics of the dataset.

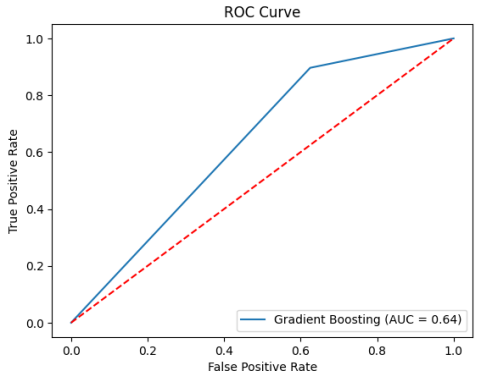
Random Forest with SMOTE



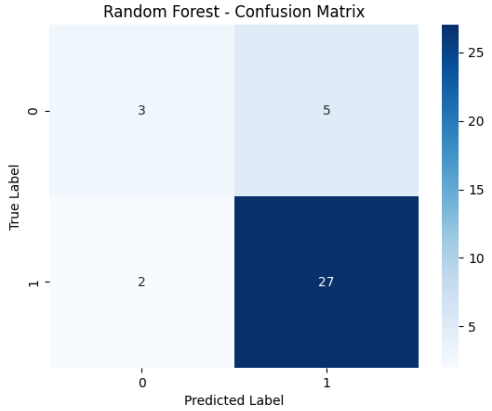


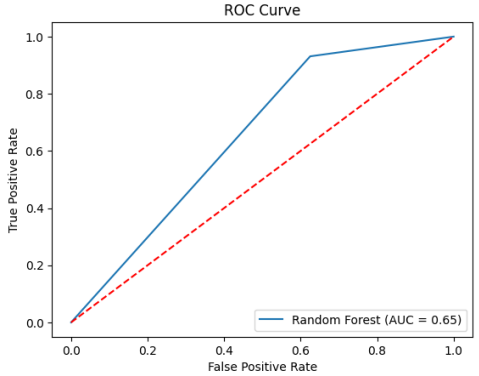
Gradient Boosting with SMOTE



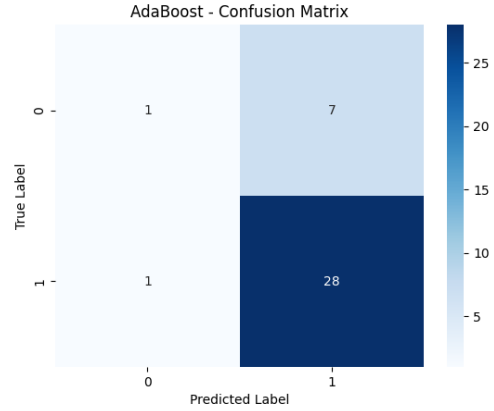


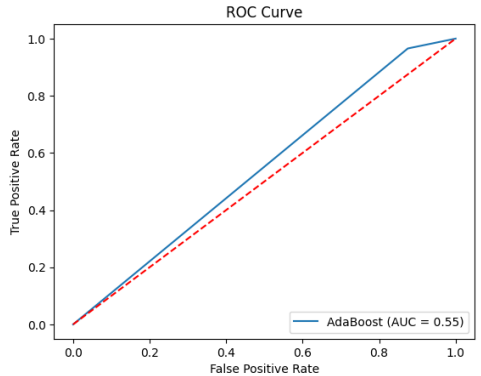
Random Forest with ADASYN





AdaBoost with SMOTE





Conclusion

In our project, we developed six distinct Machine Learning models to determine which one was the most suitable for our problem and dataset. The findings indicated that Random Forest with ADASYN consistently achieved the highest mean accuracy and mean precision scores overall. We can also find that overall machine learning models have a better mean accuracy and mean precision when combined with the balancing methods SMOTE and ADASYN.

We had some overfitting problems and to resolve them we used cross-validation, which made the results more down to earth.
We are satisfied with the models and the outcomes we obtained. This work helped us gain a better understanding of the workflow of a data science and machine learning project, as well
as improve our knowledge of various Python data science libraries.