Goal

This assignment aimed to create a model for which we could reduce the costs associated with heart attacks in the national healthcare system by 20%. For that task, we were given a dataset with data related to people's health and a binary label that indicates if the person had a heart attack.

We had all the data needed to perform the assignment, except for the probability that a person adheres to the proposed plan by the doctor. Without that information, we wouldn't be able to predict how much the savings would be for the healthcare system. However, we can estimate the minimum percentage of people adhering to the plan so that we can save 20% of the associated costs.

Calculation of adherence to the plan

Current cost

In order to save 20% of the costs, we would need to know the current cost structure, which is the following.

Current cost = (1-HA) * (CostNHA) + HA * (CostHA) = 4709.3€

HA = Heart attack probability = 0.094186

$CostHA$ = Cost of one person having a heart attack = 50000

$CostNHA$ = Cost of one person not having a heart attack = 0

After-plan cost

The cost per person after applying this program has several branches:

- People that won't have a HA but were diagnosed to have one and accept to follow the plan -> Cost: 1.000€
  C1 = FPP * 1.000€

- People that will have a heart attack but were diagnosed not to have one -> Cost: 50.000€
  C2 = FNP * 50.000€

- People that will have a heart attack, are diagnosed to have one and decide not to take the plan -> Cost: 50.000€
  C3 = TPP * (1- P(takesplan)) * 50.000€

- People that will have a heart attack, are diagnosed to have one and decide to take the plan, and they don't adhere to it -> Cost: 51.000€
  C4 = TPP * P(takesplan) * (1-P(adheresplan)) * 51.000€

- People that will have a heart attack, are diagnosed to have one and decide to take the plan, they adhere to it, and it doesn't work -> Cost: 51.000€
  C5 = TPP * P(takesplan) * P(adheresplan) * (1-P(planworks)) * 51.000€

- People that will have a heart attack, are diagnosed to have one and decide to take the plan, they adhere to it, and it works -> Cost: 1.000€
  C6 = TPP * P(takesplan) * P(adheresplan) * P(planworks) * 1.000€

FPP = False Positive Percentage = TN / Total Cases

FNP = False Negative Percentage = FN / Total Cases

TPP = True Positive Percentage = TP / Total Cases

$P(takesplan)$ = Probability that someone takes the plan when is offered to him/her = 0.85

$P(adheresplan)$ = Probability that someone adheres to the plan = Unknown

$P(planworks)$ = Probability that the plan works = 0.75

Formula to calculate the minimum adherence plan

We need to do a cost analysis to check the minimum percentage of adherence that we need in order to reduce the cost by 20%, that means, we need to get the value of P(adheresplan) so that the cost is equal than 80% of the current cost.

$$Currentcost * 0.8 = Newcost$$

$$Currentcost * 0.8 - (c1 + c2 + c3) = c4 + c5 + c6$$

$$(Currentcost * 0.8 - (c1 + c2 + c3))/(TPP * P(takesplan)) =$$

$$= ((1 - P(adheresplan)) * 51000) + (P(adheresplan) * (1 - (P(planworks)) * 51000) + (P(adherespl$$

$$= 51000 - 51000 * P(adheresplan) + P(adheresplan) * (1 - P(planworks)) * 51000) + (P(adherespla$$

$$((Currentcost * 0.8 - (c1 + c2 + c3))/(TPP * P(takesplan)) - 51000)/P(adheresplan) =$$
$$= -51000 + (1 - P(planworks)) * 51000) + P(planworks) * 1000)$$

$$P(adheresplan) =$$
$$= ((Currentcost * 0.8 - (c1 + c2 + c3))/(TPP * P(takesplan)) - 51000)/(-51000 + (1 - P(planwork$$
$$= ((Currentcost * 0.8 - (c1 + c2 + c3))/(TPP * P(takesplan)) - 51000)/(-P(planworks)) * 51000) +$$
$$= ((Currentcost * 0.8 - (c1 + c2 + c3))/(TPP * P(takesplan)) - 51000)/(-P(planworks)) * 50000)))$$

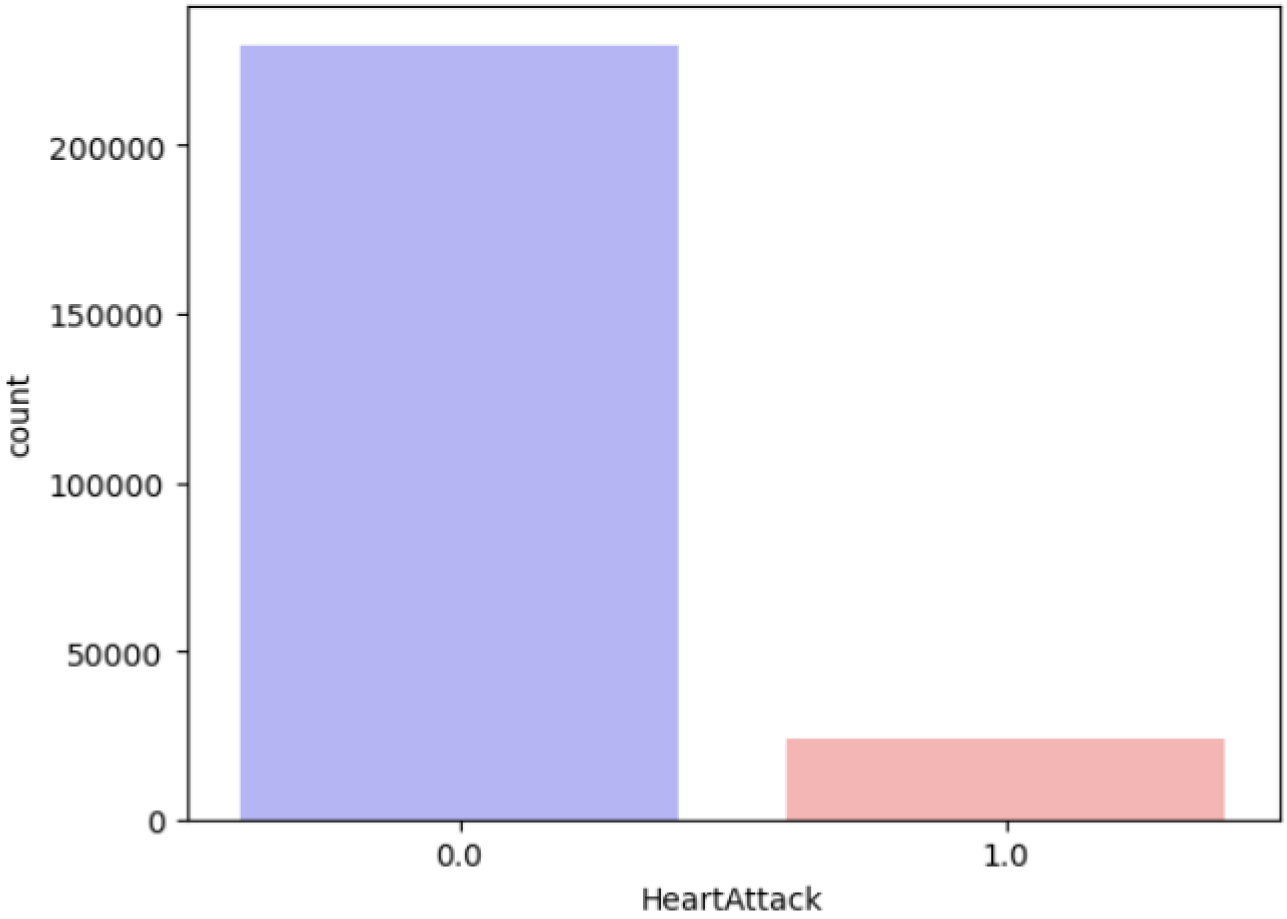
$$P(adheresplan) =$$
$$= ((Currentcost * 0.8 - (c1 + c2 + c3))/(TPP * P(takesplan)) - 51000)/(-P(planworks)) * 50000)))$$

Having this formula, we can now start analysing and cleaning the data so that we can build a heart attack prediction model.

Data analysis and cleaning

Before starting to apply models we proceeded to inspect the data, looking for missing values and evaluating how the data is distributted.

First, we evaluated the distribution of the target variable which shows that the negative (no heartattack) samples largely exceed the positive samples (heartattack) in approxcimate ratio of 1:9. This help us identify a possible problem with the model to develop, as we will have to keep it in check so it does not overly favors negative predictions. In Figure 1, the proportion of negative and positive samples can be observed. Moreover, this unbalance extedns to other attributes in the dataset, altough in a more less drastic manner, like in the case of the `Sex` attribute.



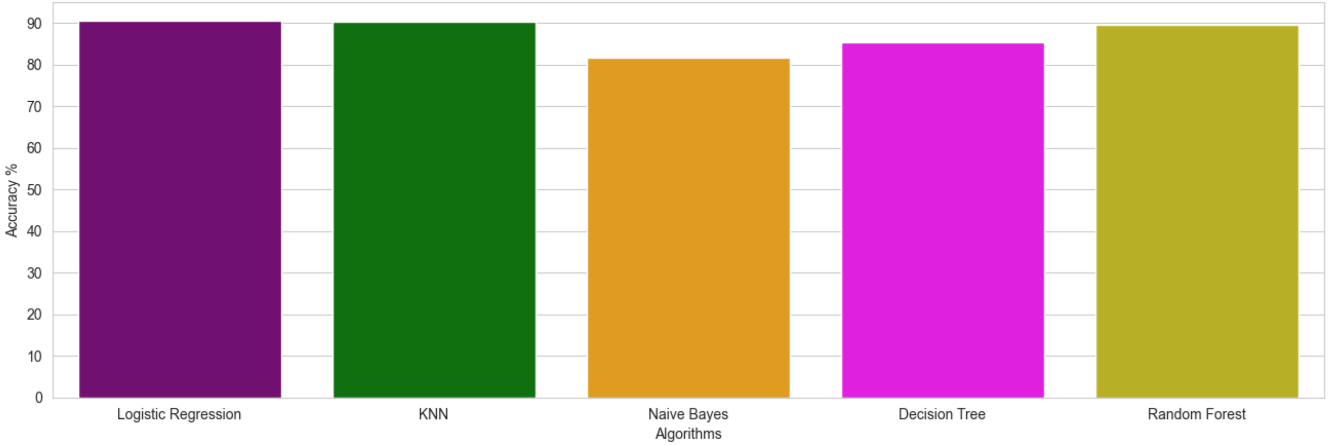
Additionally, we observed inconsistensis between the data and the description given. For instance, the attributes `PhysHlth` and `MentHlth` are described to be values between 1 to 5 but in reality these are actually in between 0 to 30 with no specific logic behind it. Furthermore, the attribute `Diabetes` is suposed to be binary but it is actually presents three diferent values (0, 1, and 2), therefore we splited this variables into two `diabetes_1` and `diabetes_2` to signify diabetes type 1 and type 2 respectively. Finally, we identified samples that had negative values in the `Age` attribute which we dropped.

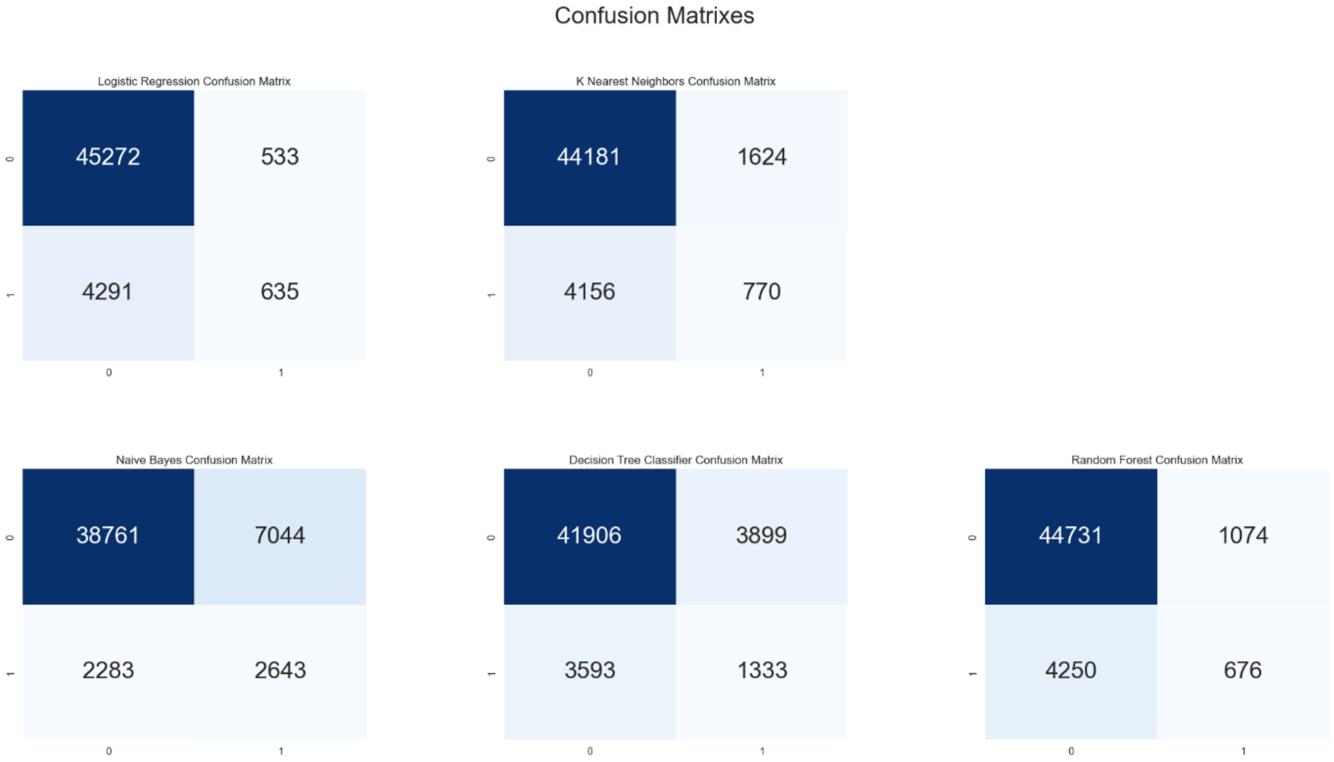
Latly, in terms of missing data, we decided to drop the rows that presented missing values as after inspections there were 27 rows presenting missing values wich represents only 0.01% of the dataset and therefore having little to no effect.

Prediction model

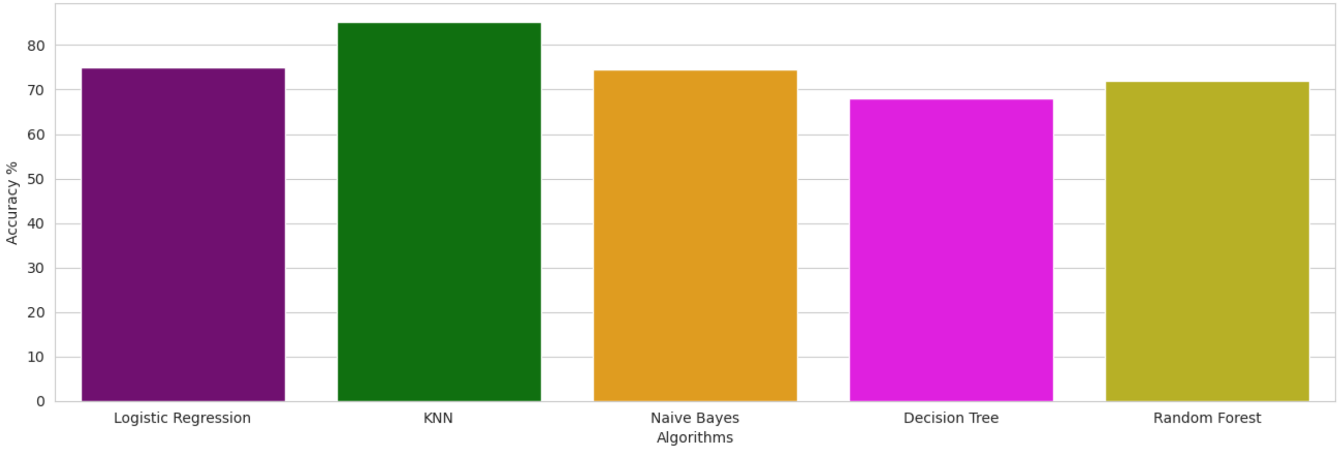
Now we the data properly preprared, we decided to quickly test different classification models to evaluate multiple models, in specific Logistic Regression, Naive Bayes, Decision Trees, Random Forest, and KNN. This would allow us to get a quick grasp of which model is responding better to the dataset and devlivering better results.

From the first run, despite getting good accuracy scores under further inspection when evaluating the confusion matrix of the different models we can see that the models are mostly inclining to make a negative prediction instead of properly learning. This results can be seen on Figure 2, Accuracy of the models and Figure 3 Confusion matrices of the models

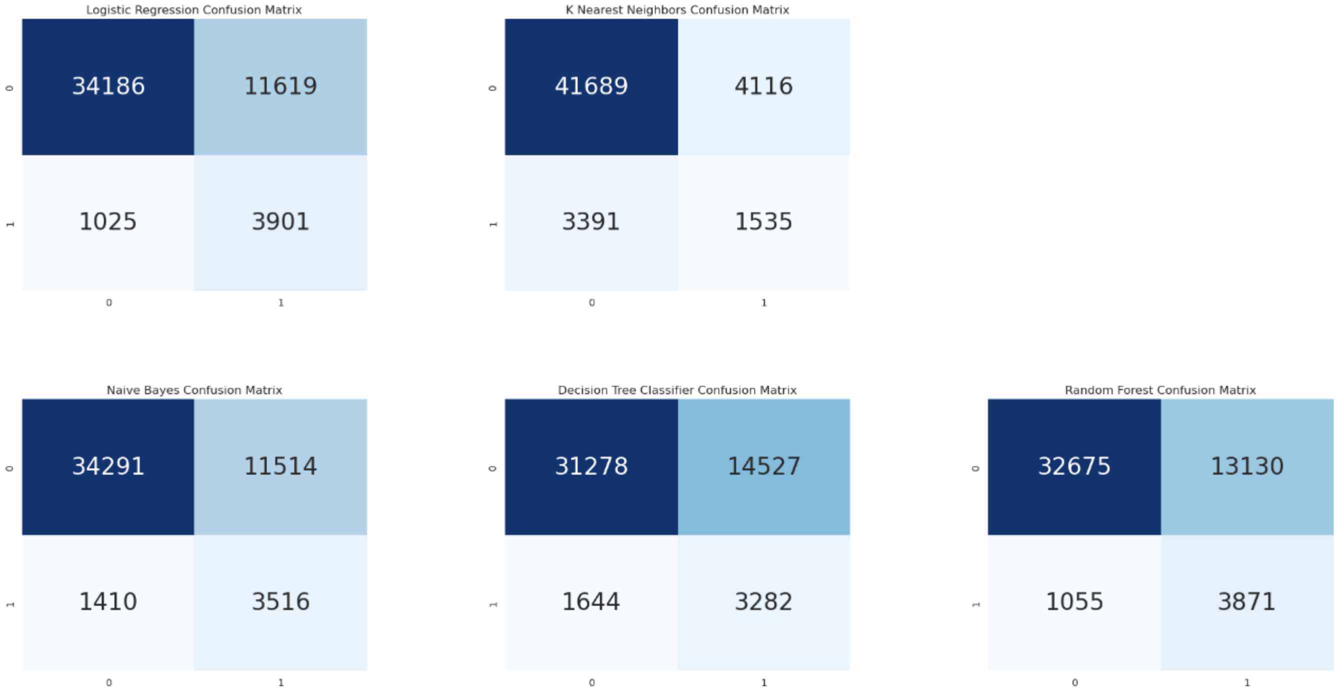


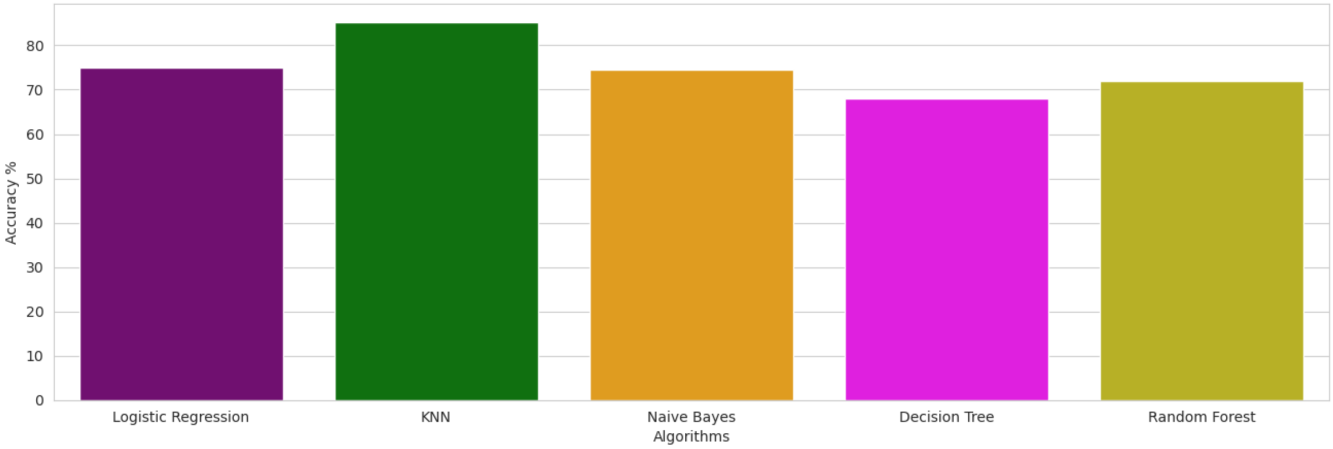


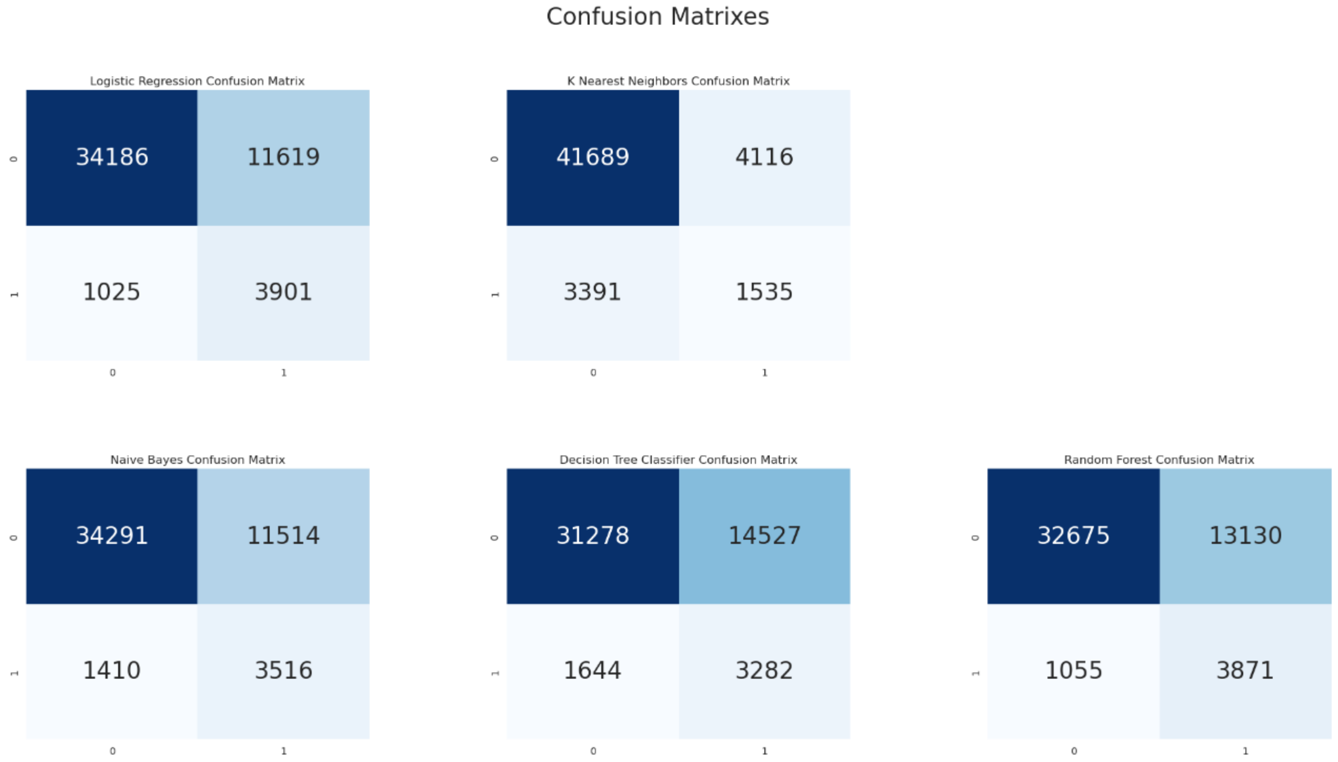
After this we decided to try to solve this skew in the results, therefore we restested the models with both oversampling and undersampling the dataset in order to have a more baanced distribution of the data and reduce the strong skew in the prediction. The results of the models with oversampled and undersampled data can be seen in Figure 4 and Figure 5 for oversampled and Figure 6 and Figure 7 for undersampled.



Confusion Matrixes







At the end, we decided to use the random forest classifier, taking into account that the model was the one performing better. In order to maximize performance, we decided to create a hyper parameter tuning pipeline with different parameters for this model.

In order to run this pipeline, we need to select a performance metric, which the algorithm will take into account in order to select the best model out of all of them. We tried using `f1_score_weighted` and `recall`. However, the most important type of person to classify correctly are the people who will have a heart attack, because if it goes undetected, the cost is really high. This makes really important to obtain the highest TPR (True positive rate) possible. On the `f1_score_weighted` score, even if it takes those into account, they doesn't have the importance that they do in our problem. Then, it felt straightforward to go for `recall`, the issue was that it was barely classifying correctly the negative cases in order to maximize the true positives, which at the end didn't perform as expected.

For that reason, we decided to build our own performance metric, the adherence probability to the plan, which we can see down below. We return -1000 in order for the function to be used correctly by `GridSearchCV` from the `sklearn`library.

```
def minimize_adherence(y_true, y_pred):
    cm = confusion_matrix(y_test,y_head_rf)
    heart_attacks = len(y_test[y_test==1])
    total_cases = len(y_true)
    fpp = cm[0][1]/total_cases
    fnp = cm[1][0]/total_cases
    tpp = cm[1][1]/total_cases

    adherence =  minimum_adheresplan_prob(total_cases, heart_attacks, fpp,
fnp, tpp)

    return adherence if (adherence > 0 and adherence < 1) else -1000
```

The best performing model was the one with the following parameters:

```
{
'class_weight': 'balanced',
'max_depth': 8,
'max_features': 3,
'n_estimators': 50
}
```

This way, we obtained the following performance metrics:

$$P(adheresplan) = 51.86\%$$

$$Accuracy = 72.74\%$$

$$Recall = 81.03\%$$

$$f1\_weighted = 78.1\%$$

$$Confusion\_matrix = \begin{bmatrix} 32911 & 12894 \\ 934 & 3992 \end{bmatrix}$$

This is a fairly good result, taking into account our previous results with different configurations and models. With this model, in order to obtain a 20% reduction in costs for the healthcare system, a 51.86% of adherence to the plan is needed, which is a reasonable goal.

Conclusion

We have achieved the goal of this project by creating a model that can reduce the costs associated with heart attacks in the national healthcare system by 20% having just 51.86% of adherence to the medical plan proposed. This was performed by doing a isolation of the adherence probability on the cost equation, followed by an extensive analysis of the dataset, the creation of different prediction models and the fine tuning of the best performing one.