Machine Learning Group Project

Group 6 — Igor Ershov (20323105), Boris Flesch (20300025), Yasmine Guendouz (20325017)

1. Introduction

Initially, our aim was to evaluate how factors such as sleep, physical activities and sleeping time influenced the academic performance of students. This type of study would be of paramount importance because it can help target the habits leading to lower grades. Throughout this project, our data revealed no relevant findings linked to the student's grades. Our main goal still remains the same as we want to understand how important a student's habits are. The data that we use is taken from the National Sleep Foundation and is named "2006 Sleep in America® Poll – Teens and Sleep". This dataset contains the answers of a poll questionnaire carried out on 1600 students. The questions are either asked to the students or their parents.
Our strategy can be described as following: (i) Pre-processing of the data (ii) Analysis of the relevant features and target value (iii) Use of different models (iv) Selection of the most accurate model.
The features we decided to use as the input were a set of questions that polled the frequency of certain types of disruptive behaviours in the daily life of the teens. We used four different approaches in this multiclass problem to predict the depressive mood score (dpmscore) of the students: Logistic Regression, kNN, Decision Tree and MLP. As explained in the Summary of Findings, slide 44, the depressive mood score is used to evaluate the depressive mood of teenagers. The three possible outputs (i.e. target classes) of this depressive mood score are "not at all", "somewhat" and "much".

2. Dataset and Features

2.1. Description of the dataset

Our dataset is a CSV file that contains answers from participants of a questionnaire about teenagers' sleeping habits: "2006 Sleep in America Poll Screening Questionnaire". The dataset is composed of 218 columns that represent the questions of the poll (i.e. features) and the answers from 1600 participants. Most of the answers are integers that correspond to specific values or range of values defined by the questionnaire itself. Values 98 and 99 respectively mean "Refused" and "Don't Know". Here is a snippet of the data:

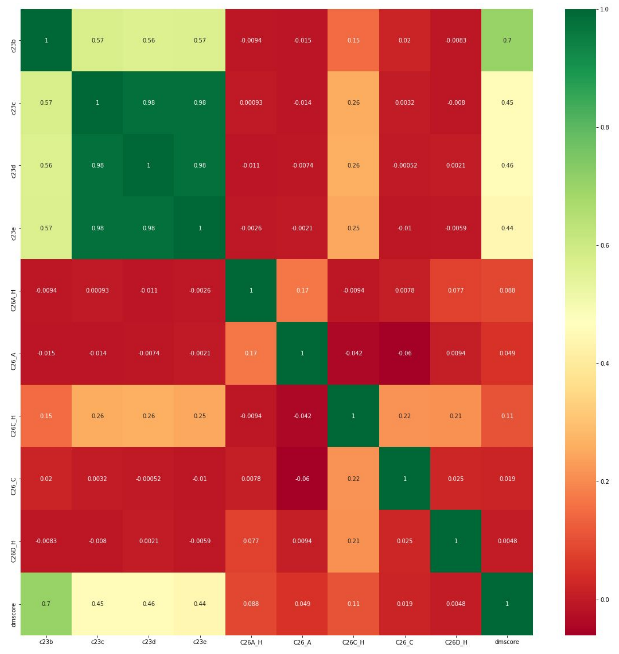
```
caseid,region,id,city,state,zip,fips,dma,census,rep,age,qs1,qs2,qs2a,qs3y_1,qs3m_
1,qs4_1,qs6_1,qs3y_2,qs3m_2
5255,3,NM6640,ALTAMONTESPG,14,32714,12117,534,5,2,Hispanic15-17,1,2,2,10,,2,6,15,
,2,10,,,,,,,,,,,,,,,,,
```

2.2. Preprocessing

The first thing we noticed when looking at the data was that most questions were optional. Furthermore, some of the questions contained civil registration data that was not of interest to us. The first preprocessing step was then to remove the civil registration data, we also removed all the questions which did not lead to numeric values as responses and replaced the missing responses from optional questions with the value 99 to avoid having any NaN.
Moreover, we searched for questions with a high amount (>1%) of invalid responses and removed them from the dataset. This allowed us to only focus on features with a relatively low amount of noise while still having a significant amount of features. We transformed the 218 question survey into a set of around 90 potential features doing these preprocessing steps.

2.3. Features discovery

In order to study the correlation between the features of our dataset, a heatmap is plotted. The heatmap is used for data visualization. The data values are illustrated with colours showing high (green) or low (red) correlation. A for loop is used to plot a heatmap between 10 features and a target value chosen. This method helps us evaluate which features would be relevant to train our model and what target value to choose.
The target value chosen in this case is the ‘dmscore’. In the following heatmap, the last column illustrates the relation between the ‘dmscore’ and the question asked to the students related to their mood and how they feel. This is thus logical to have high correlation as seen in Figure beside We then know that those features should not be taken into account when training our models.



Another approach we took to find correlations between the features was to randomly choose a combination of 2 features for the input, and another feature as the output. Once we had the inputs/outputs, we ran a LinearRegression model with 10-fold cross-validation.
As the questions were asked in sequential order, we found that questions near each other were strongly correlated, but this wasn't interesting as those were usually direct correlations. To avoid this, we added a check in the random selection of features that would only choose features that were not "near" each other in a range of 5 questions.
We ran this process around 100000 times to find any correlation. We saved to a file the features that lead a significant increase (>1000%) in accuracy against a baseline that predicts the 'most_common' class.

2.4. Features selection

As a result of both ways of finding the features, we found a potential correlation between the answers to the 'C17' range of questions and the depressive mood score. The 'C17' questions ask teenagers how frequently they experienced certain behaviours in the recent period, mapping a value from 5 for "Every night or almost every night" to 1 for "never" (2006 Sleep in America Poll Screening Questionnaire, page 15). The behaviours polled involve a mix of sleep-related ones and general ones such as : “Had nightmares or bad dreams”, “Had trouble concentrating or paying attention in school or while doing homework” , “Fallen asleep in school” or “Felt cranky or irritable during the day”. In total, there are 17 different questions that we used as input features. As each question had an exact value from 1 to 5, they were easily usable as input features.
For the output, we chose the "Depressive Mood Score" (dmscore), a scale used to assess depressive mood. This score was assigned to each teen based on an unrelated part of the questionnaire.
The scores can range from 10 to 30, with 10 being the lowest and 30 being the highest depressive mood. We divided the various scores into three separate classes as suggested by the questionnaire’s Summary of Findings, slide 44. With a dmscore of 10-14, 15-19 and 20-30, the class would respectively be “Not at all”, “Somewhat” and “Much”.

3. Methods

3.1. Logistic regression

A Logistic regression classifier uses a linear regression model to make predictions. It will use a sign function that will check if the θTx result goes over a “Decision Boundary”, if the output does so, it will be classified as positive, else it will be deemed negative. In a multiclass scenario, it will use a one-vs-rest approach. We are using the default “lbfgs” solver, which uses L2 regularization. We will find the C parameters, which will act as the penalty weight, using cross-validation. Furthermore, we will try to engineer features using Polynomial Features, the order of which will also be validated using cross-validation.

3.2. kNN Classifier

A kNN Classifier is an instance based model that makes decisions directly based on the training data. For each feature vector x it will calculate the distance between the vector and the training data, only selecting the k data points which are closest to it. From these data points it will then predict an output y based on the majority vote.
We will decide the number of neighbours using cross validation.

3.3. Decision Tree Classifier

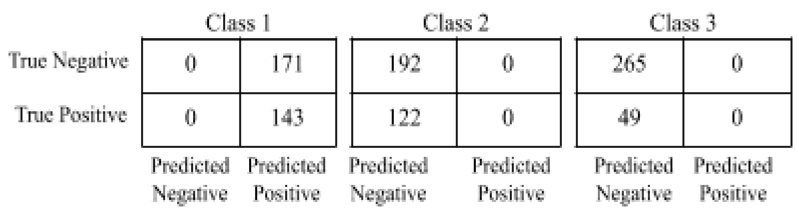
A Decision Tree Classifier model was used on our dataset. This type of machine learning method, as its name suggests, is built through branches and nodes. The higher the number of branches, the deeper the tree is. Those nodes and branches further split until arriving at the decision or output wanted. It acts following an if-else method. The tree depth is an important parameter when trying to avoid under or over-fitting. Moreover, different strategies can be used to refine a Decision Tree Classifier. In our case, the criterion that will be used is the 'entropy'. The latter measures the quality of the separation in the tree branches. The random state is set to '42' and the tree depth or 'max_depth' is determined using cross validation.

3.4. MLP

A Multi-Layer Perceptron (MLP) is a network composed of three layers: input layer, hidden layer and output layer. The number of hidden layers and nodes composing each hidden layer can be determined by cross-validation. In our case, we want to predict classification for depressive mood score range depending on some answers of the questionnaire (i.e. input features). Therefore, we are using *MLPClassification* from scikit-learn with the default *adam* solver, which is a stochastic gradient-based optimizer, and *ReLu* activation function which is a standard non-linear approach (scikit documentation).

4. Experiments, Results and Discussion

We followed the same approach for each of our trained models. Firstly, the data is split between the training/testing set (80%) and the validation set (20%) and that is respectively 1253 and 314 entries. To tune our hyperparameters, we performed k-fold cross-validation (k=5) using the training data (i.e. itself splitted into training and test data).
The metric used to evaluate our models during training was the F1 score as we have a large class imbalance in our data set. After having chosen the hyperparameters, we tested our model using validation data. The accuracy of our model has then been compared against a baseline model (*DummyClassifier*) that outputs the most frequent class. The confusion matrix of the baseline model is illustrated below (using the *multilabel_confusion_matrix* function); it provides an accuracy of 46%:



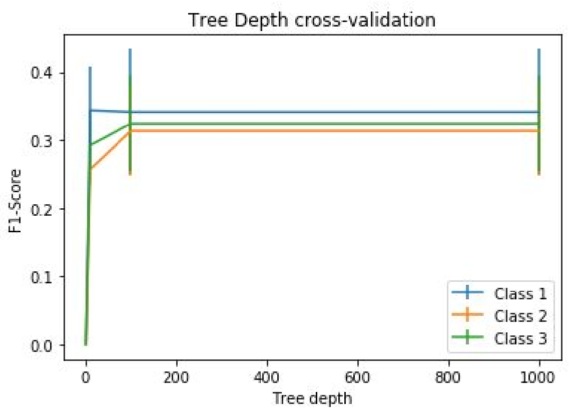
	Class 1		Class 2		Class 3	
	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive
True Negative	0	171	192	0	265	0
True Positive	0	143	122	0	49	0

Baseline Confusion matrices

Moreover, ROC curves have been plotted for each class alongside with the average ROC of our multilabel model classifier using the "micro" method. This method needs for the target column to be binarized with respect to each class (i.e. using *label_binarize*). Each ROC curve refers to the predictions of each class while the average ROC curve takes all the predictions as input. Finally, we evaluated the AUC of our models. To summarise, our metrics are:

- **Confusion matrices** : A quantitative way to visualize the performance of our models is the confusion matrix. It is a table with four combinations of the predictions and the true values being true positive, true negative, false positive, true negative.
- **F1-score** : This metric is a combination of both the precision and the recall as it estimates the influence of false positives and false negatives. This is the reason why this metric is more appropriate in the case of our imbalanced data set.
- **ROC/AUC**: The ROC curve corresponds to the plot of the True Positives rate against the False Positives rate. When 100% of predictions are correct the curve has a point in the top left-corner. This is a good way to visualize the efficiency of a model as we want the curve to be as close to the top-left corner. The Area Under the ROC Curve quantifies the classifier's performance. An ideal model has an AUC equal to 1.
- **Accuracy**: This measures the correct predictions over the whole predictions.

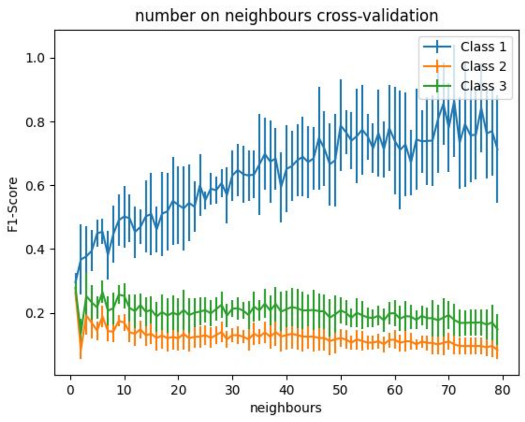
4.1. Decision Tree Classifier



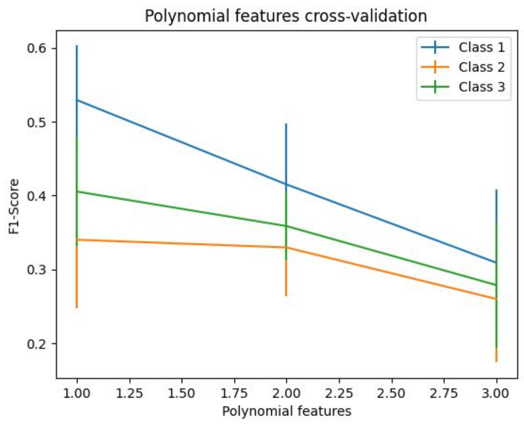
In order to obtain a suitable tree depth for our decision tree classifier, we perform cross-validation. The tree depth tested ranges from 1 to 1000, those values allow us to visualize how the model behaves throughout a wide spectrum of branches. The F1-score metric is then plotted against the tree depth for each class as seen in the figure. We want to maximize the F1-score while avoiding overfitting. We can observe that in the above figure, the F1-score increases until reaching a value of 100 branches then converges. Choosing a low tree depth would yield underfitting our data. On the other hand, increasing the tree depth to 1000 would not change the variance nor the F1-score. Thus, a three depth of 100 seems adequate.

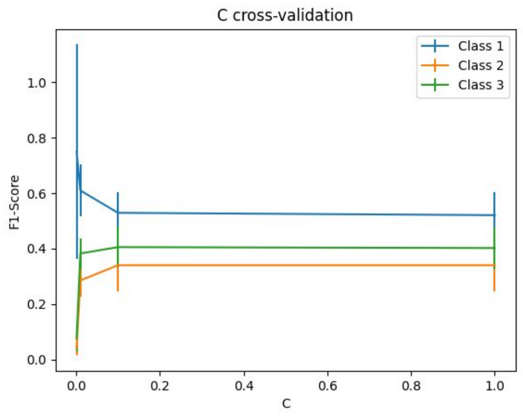
4.2. kNN

We will use cross-validation to find the optimal number of neighbours of the kNN classifier.
The number of neighbours will be chosen from a range from 0 to 80. As we can see from the graph on the left it seems that kNN is biased towards the first class. This is because the first class is also the most common one. At around k=35 we see an increase in all the classes' variance; because of this, we can choose this value as our number of neighbours.



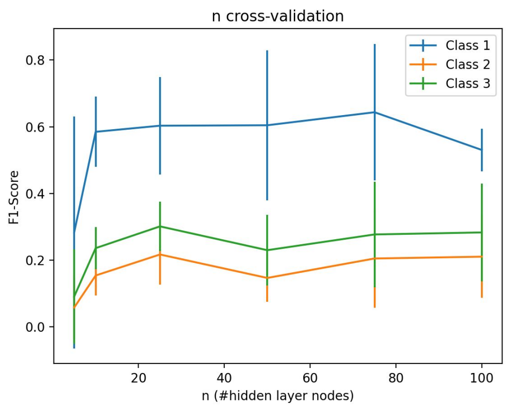
4.3. Logistic regression

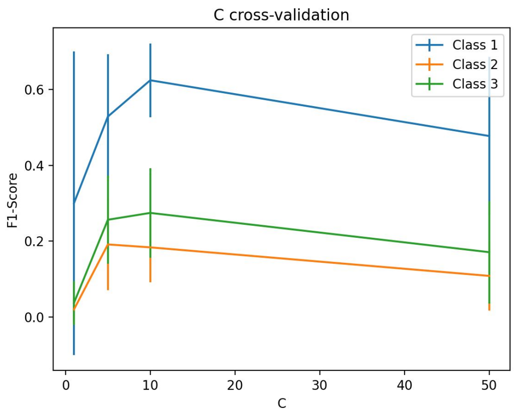




The order of polynomials will be chosen from a range from 1 to 3 with cross validation. We chose a small range to validate because we already have many features (17). As a starting C we will arbitrarily use 0.1.
As we can see from the graph above, adding polynomial features decreases the overall F1-score, while still keeping the same value of variance. Because of this, we will not use polynomial features.The C value will be chosen with cross validation from a range of [0.001, 0.01, 0.1, 1]. As we can see on the right graph, the variance and accuracy quickly converged at around 0.1, The model will be trained with no polynomial features and with a C=0.1

4.4. MLP





To determine the appropriate values for hyperparameters *n* and *C*, we used cross-validation. We can see from the results obtained by cross-validation(figures above) that the F1-Score is not much impacted by changing the number of hidden layer nodes. To prevent overfitting, we can keep a quite small value for *n*, such as *n=25* to minimise standard deviation (i.e. spread of the results). We can use a similar reasoning for *C*, where *C=10* seems to be an appropriate value. The *precision_recall_fscore_support* function has been used to calculate metrics of the model, in addition to its confusion matrices and ROC curves.
Further testing has been made using additional hidden layers and cross-validating the number of nodes of that new layer. Adding nodes and/or layers (i.e. adding weights in the network) could lead to fit more to the training data by capturing more patterns, hence potentially improving the overall performance of the model. However, we did not observe any improvement with these modifications and decided to keep only one hidden layer of *n=25* nodes.

4.5. Discussion of the results

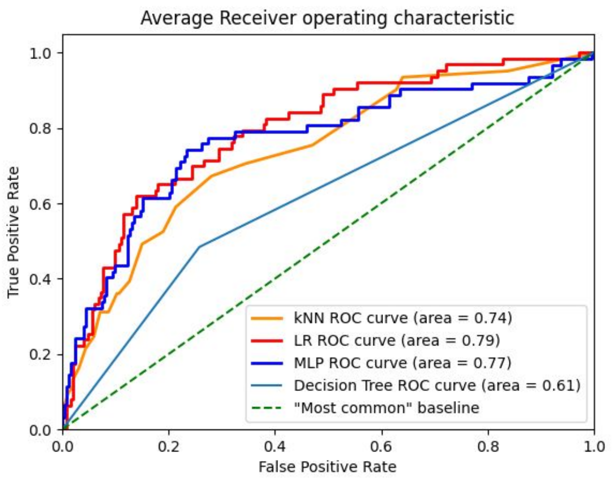
Logistic Regression results
With a Logistic Regression model with C=0.1 that the predictions for the first class are generally good, while the ones for the second and third class are worse. We can see on the ROC that we get a reasonably good result with an AUC of 0.80, a 76% increase over the baseline. This doesn't carry over to good accuracy, as we only get an accuracy of 57% which is a 23% improvement over the baseline, but not a good accuracy overall.

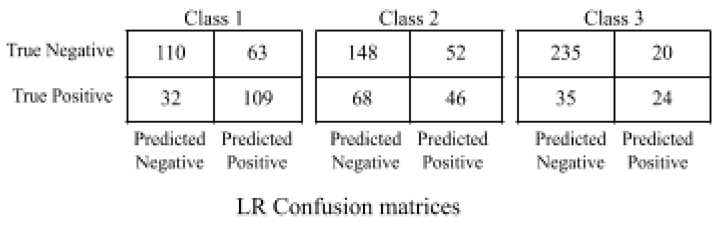
kNN results
With a kNN with k=35 the predictions for the first class are generally good, while the ones for the second and third class are worse, with the second class performing a bit better. We can see on the ROC that we get a fairly good result, with an AUC of 0.78, a 56% increase over the baseline. This doesn't carry over to a good accuracy, as we only get an accuracy of 55% which is 20% better than the baseline, but not a good accuracy overall.

Decision Tree Classifier results
As seen in the ROC curve, the Decision Tree Classifier (tree depth=100) is better performing than a baseline model. However, there is no striking difference as the area under the curve is equal to 0.57 while it is equal to 0.5. There is a difference of 12% which is relatively small. The accuracy is calculated between the predictions on the testing set and the testing set itself. This leads to a value of 49%. This shows that our tree classifier is more accurate than the baseline model by 3% which is not a striking improvement.

MLP Network results

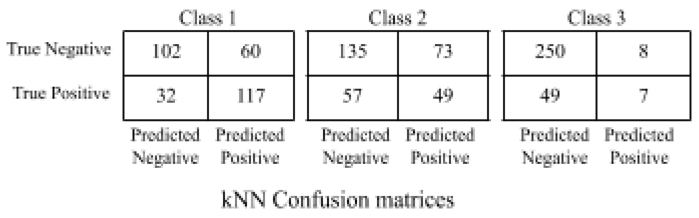
The results obtained do not show any sign of over-fitting: F1-Score is not peaking (avg 0.58), neither are precision (avg 0.6) or accuracy scores (avg 0.4). On the other hand, none of the metrics used reveal under-fitting, except for very small values of *n* and *C* when tuning hyperparameters (i.e. $n \leq 10$ and $C<5$). We can see from our ROC curves that the MLP network performs better than the baseline model. However, not any high value is reached (i.e. upper-left corner of the curves). Therefore, as under-fitting does not seem to be the cause of this issue, we can argue that either the correlation we try to determine is quite weak or the model (MLP Network) is not appropriate. It is also interesting to note that the MLP seems to perform largely better to predict Classes 1 and 3 than Class 2, as shown in individual ROC curves plot, F1 scores and confusion matrices.





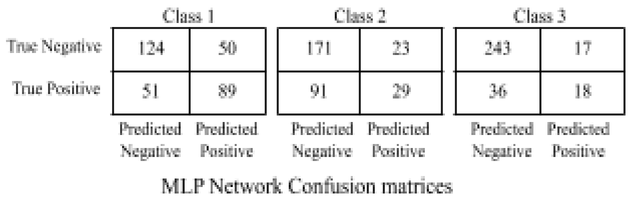
	Class 1		Class 2		Class 3	
True Negative	110	63	148	52	235	20
True Positive	32	109	68	46	35	24
	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive

LR Confusion matrices



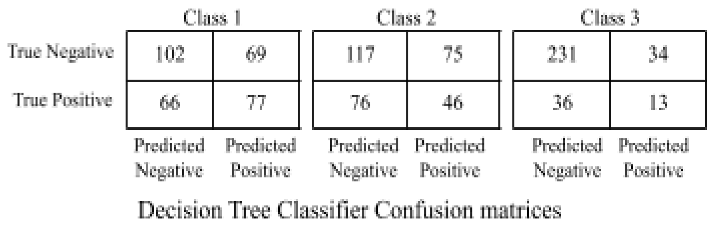
	Class 1		Class 2		Class 3	
True Negative	102	60	135	73	250	8
True Positive	32	117	57	49	49	7
	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive

kNN Confusion matrices



	Class 1		Class 2		Class 3	
True Negative	124	50	171	23	243	17
True Positive	51	89	91	29	36	18
	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive

MLP Network Confusion matrices



	Class 1		Class 2		Class 3	
True Negative	102	69	117	75	231	34
True Positive	66	77	76	46	36	13
	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive	Predicted Negative	Predicted Positive

Decision Tree Classifier Confusion matrices

Final results

After running and testing these various models, we can argue that the best model for this dataset is Logistic regression, as it had an AUC of around 0.8, while also having an accuracy of 0.57. While being better than the baseline, this is still not a worthwhile model to train as there does not seem to be a strong enough correlation between the data we have chosen.

5. Summary

Overall, we were able to test four different classifiers on our data set. K-fold cross-validation with k=5 was performed for each model in order to tune the hyper parameters. The best model being the Logistic Regression with an accuracy of 0.57 and an AUC of 0.8 and the worst one being the Decision Tree Classifier with an accuracy of 0.49 and an AUC of 0.61. One model that was interesting to refine was the MLP- having an AUC of 0.77- by adding hidden layers. It is also important to note that the MLP accuracy is much lower than the Logistic Regression model. The latter is thus the most accurate one, but its accuracy of 0.57 does not allow us to conclude that we would be able to predict the depressive mood score (*dmscore*) of a teenager given their answers to question C17 of the questionnaire (2006 Sleep in America Poll Screening Questionnaire, page 15).

6. Contributions

- Boris - B.F.
  - Dataset discovery and testing
  - Data exploration Code
  - Report Dataset introduction
  - MLP classifier Code & Report about it
  - Cross-validation Code
- Yasmine - Y.G.
  - Dataset discovery and testing
  - Report Introduction & Summary
  - Heatmap feature discovery Code & Report about it
  - Decision Tree classifier Code & Report about it
  - ROC Curve Code for each class
- Igor - I.E.
  - Dataset discovery and testing
  - Parsing and Preprocessing Code & Report about it
  - Random feature discovery Code & Report about it
  - Logistic regression Code & Report about it
  - kNN Code & Report about it

7. GitLab link

https://gitlab.scss.tcd.ie/fleschb/ml-group-project