MIRI - Machine Learning

Machine Learning Methods to Predict Student Performance

José Miguel Hernández Cabrera (jose.miguel.hernandez)

Jaume Pladevall Morros (jaume.pladevall)

June 17, 2021

Contents

1 Introduction

Educational Data Mining (EDM) is a disciplinary research area, concerned with developing methods for exploring the unique and increasingly large-scale data that can be obtained from massive online open courses (MOOCs), educational platforms, administrative data from schools and universities, etc [1].

EDM works towards the improvement of educational processes by introducing better, and effective learning practices for students [2]. Therefore, in this project, we used machine learning methods to study student's performance based on a series of variables: previous grades, relation with the family, support from the school, etc.

We worked with the data provided by Cortez and Silva [3] and retrieved from the UCI Machine Learning Repository [4]. This data was collected during the 2005-2006 school year from two public schools from the Alentejo region of Portugal. The data features can be classified into two categories: mark reports and questionnaires. This data was later integrated into two different data sets. One for the subject of Mathematics and the other one for the Portuguese language subject. Each data set has 33 features and a different number of students (649 for Portuguese and 395 for Mathematics).

Our objective is to use machine learning methods to predict student success according to the information retrieved about them and also find which are the most important features that affect the student performance. Given that we have two separate data sets available, one of the subject Portuguese and one of the subject Mathematics. We tried to see if with a model only trained to predict the success of students in the Portuguese subject could be generalized into predicting the success of students for the Mathematics subject.

In the next section, we dived more into how the data was obtained and what the authors of the data set did with it.

The Machine Learning methodology that we followed in this report started with a data exploration of the data sets, followed by pre-processing, which includes data cleaning, data transformation and feature selection. Next, we proceeded to the Modeling phase, where different classification models were built. Then, the results of these models were compared and conclusions were extracted. Lastly, a final section about future work was added.

2 Background and Methodology

In the original paper, Cortez & Silva [3] built the database from two sources: school reports, based on paper sheets and including few attributes (i.e. the three period grades and number of school absences); and questionnaires, used to complement the previous information. The latter was designed with closed questions (i.e. with predefined options) related to several demographic (e.g.

mother's education), social/emotional (e.g. alcohol consumption) and school related (e.g. number of past class failures) variables that were expected to affect student performance.

Afterwards, they used Decision Trees (DT), Random Forest (RF), Neural Networks (NN) and Support Vector Machines (SVM) to predict secondary student grades of two core classes: Mathematics and Portuguese. They concluded that past evaluations highly influence students achievement, along with number of absences, parent's job and education, and even alcohol consumption. They also called for further research in feature selection methods and hyper-tuning for non-linear function methods, such as NN and SVM, given that are sensitive to irrelevant data.

In our work, we used a different approach in terms that instead of using the same models twice in both data sets (i.e. creating a train data set per each data set), we opted for using the Portuguese data set as the only train data set and use it to predict the Mathematics data set. Also, focused our analysis only in predicting a binary classification, transforming the final grades into two classes: pass or fail, encoded as $y \in \{0, 1\}$. We also used a wider range of models, including Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis and Naive Bayes.

With respect for the metrics, we considered four metrics: accuracy, precision, recall and the F1-score. As a brief reminder, accuracy is the sum of all samples correctly predicted in a determined class (true positives) predictions by the total number of samples, but it has the disadvantage of being very sensitive to imbalanced data. The precision metric is the number of true positives divided by the sum of true positives and samples of another class incorrectly predicted as the current class (false positives), which is sensitive to false positive predictions. The recall is the true positives divided by the sum of true positives and the samples of the current class predicted incorrectly as another class (false negatives). In this sense, recall is also sensitive to false negative predictions. Lastly, the F1-score is the harmonic mean of precision and recall. Which provides a good balance between precision and recall. However, it is harder to interpret.

For our particular case, our data is unbalanced and models tended to allocate more observations to class pass. Therefore, accuracy, recall and precision would not be a good metric for us. In this sense, we focused on the F1-score metric since we are interested in having a balance between recall and precision. Further reasons are explained in section 4.

Another important decision for our work consisted in using both *G1* and *G2* grades along with the other features, except for the final grade (*G3*), to train our models. As opposed to what Cortez & Silva did which was use three setups: (1) all variables with *G1* and *G2* but no *G3*, (2) all variables but *G2* and *G3*, and (3) all variables but *G1* and *G3*. We did not consider doing this since considering both grades is important for the final grades, hence sticking to only the first configuration of the authors' binary classification.

3 Data Exploration Process

3.1 Exploration Data Analysis

Both data sets have 33 variables, the size of them are different: Mathematics has 395 observations, while the Portuguese has 649. Details regarding their description, type and domain are shown in Table 1. Out of these observations, we found no missing values.

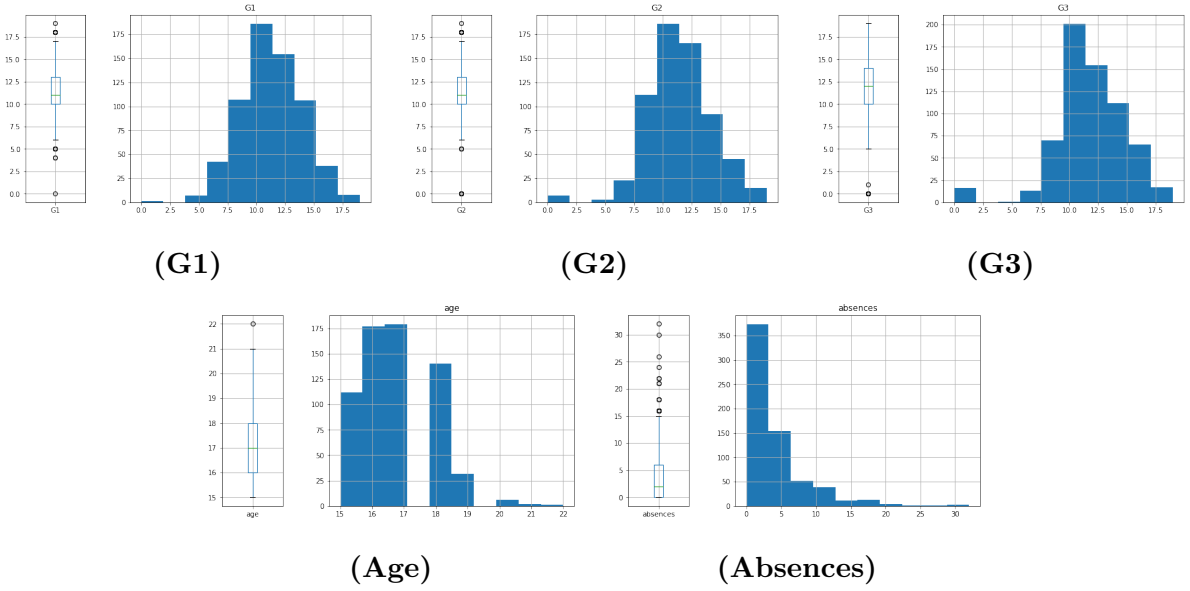
Table 1: Variable descriptions

	Attribute	Description	Type	Domain
1	G1	First period grade	Numeric	0 - 20
2	G2	Second period grade	Numeric	0 - 20
3	G3	Final grade	Numeric	0 - 20
4	age	Student's age	Numeric	15 - 22
5	absences	Number of school absences	Numeric	0 - 93
6	failures	Number of past class failures	Numeric	n if 1 <= n < 3, else 4
7	sex	Student's sex	Binary	female or male
8	school	Student's school	Binary	GP or MS
9	address	Student's home address type	Binary	Urban or rural
10	Pstatus	Parent's cohabitation status	Binary	Living together or apart
11	famsize	Family size	Binary	<= 3 or 3
12	schoolsup	Extra educational school support	Binary	Yes or no
13	famsup	Family educational support	Binary	Yes or no
14	activities	Extra-curricular activities	Binary	Yes or no
15	paidclass	Extra paid classes	Binary	Yes or no
16	internet	Internet access at home	Binary	Yes or no
17	nursery	Attended nursery school	Binary	Yes or no
18	higher	Wants to take higher education	Binary	Yes or no
19	romantic	With a romantic relationship	Binary	Yes or no
20	Mjob	Mother's job	Nominal	Five job descriptions[1]
21	Mjob	Father's job	Nominal	Five job descriptions[1]
22	guardian	Student's guardian	Nominal	mother, father or other
23	reason	Reason to choose this school	Nominal	Four reason categories[2]
24	Medu	Mother's education	Ordinal	Five education levels[3]
25	Fedu	Father's education	Ordinal	Five education levels[3]
26	famrel	Quality of family relationships	Ordinal	1 - very bad to 5 - excellent
27	traveltime	Home to school travel time	Ordinal	Four time intervals[4]
28	studytime	Weekly study time	Ordinal	Four hour intervals[5]
29	freetime	Free time after school	Ordinal	1 - very low to 5 - very high
30	goout	Going out with friends	Ordinal	1 - very low to 5 - very high
31	Walc	Weekend alcohol consumption	Ordinal	1 - very low to 5 - very high
32	Dalc	Workday alcohol consumption	Ordinal	1 - very low to 5 - very high
33	health	Current health status	Ordinal	1 - very low to 5 - very high

[1] Teacher, health care related, civil services, at home, other.
[2] Close to home, school reputation, course preference, other.
[3] 0 - none, 1 - 4th grade, 2 - 5th to 9th grade, 3 - secondary education, 4 - higher education.
[4] 1 - <15 min, 2 - 15 to 30 min, 3 - 30 min to 1 hour or 4 - >1 hour.
[5] 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours or 4 - >10 hours.

3.2 Outliers

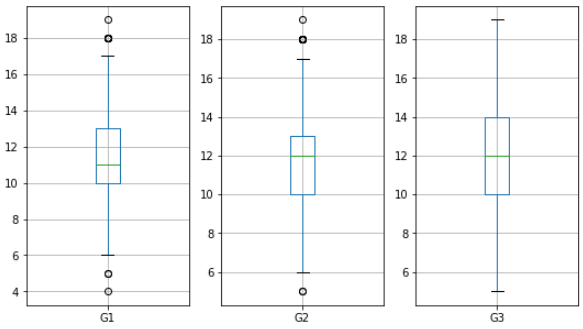
We found that some students could be outliers. However, it was in our interest to be able to classify this students, specially since they are students that failed the subject. Therefore, we trained our models with the grade outliers and later checked what would happen if we kept them.



(G1) **(G2)** **(G3)**

(Age) **(Absences)**

Figure 1: Histograms and boxplots for outlier detection of grades, age and absences.



(G1, G2 and G3)

Figure 2: Boxplots of grades after dropping outliers.

For the grades outliers, we dropped all the ones that were equal to 0 (Figure 2). However, for the outliers in *absences*, we imputed with the mean the extreme values (visual threshold from Figure 1 (Absences)), greater than 30) which felt outside the exponential distribution.

3.3 Transformations

One thing that we detected from Cortez & Silva [3] is that they relied on the usage of likert scales (coded as ordinal in Table 1) as they were numerical. In contrast, we decided to use the one hot

encoding technique to transform them, along with the binary and nominal variables.

We also normalized the numerical variables in order that the scale of the variables doesn't affect the hole model by having the same importance when training, specially, for the models that are not scale resistant.

3.3.1 Response variable

We transformed our response variable from numerical to binary. Following the preoccupation of the original paper. Because, we want to differentiate between students that pass from students that fail. By following the rule that if the *G3* grade was greater or equal than 10, the student had passed the subject, failed otherwise.

3.4 Feature importance

One advantage that RF and DT have is that they perform feature selection on training. We used RF to have a more profound understanding of which features were the most important for the RF model (Figure 3).

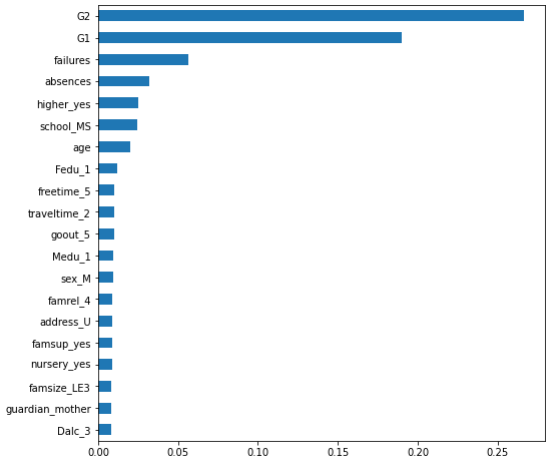


Figure 3: Important variables of the Portuguese data set for a default Random Forest model.

According to our RF model, *G2* and *G1* are the most important variables for predicting the *pass_fail* target. Followed by: *failures, absences, higher, school* and *age*. Making that a considerable amount of the variables have low or near to none importance value for the RF.

3.5 Feature selection

In the start of the EDA, we observed that some variables didn't show any sign of correlation between its values and the final grade. And from the previous section the RF model returned only a subset of the variables as their most important. Nevertheless, we decided to keep all of them while no model seemed to be affected negatively by them. We did this because we wanted to generalize also

for the math data set, so we needed to have as much information as possible. However, for the models that are known to be affected by irrelevant data, we trained a version of them with only the most important features to see if the classification improved.

4 Modeling

4.1 Data set partition

For the train/validation/test data sets, we chose stratified partitions; meaning that we kept the same proportion of *pass/fail* examples in each partition. Since we have an imbalanced data set this is important considering that a random partition may increase the imbalance.

4.2 Choosing metrics

We used the *F1-score (class)*, *F1-score (macro avg)*, and accuracy metrics. These metrics gave us a precise view of how our model was performing. In this particular problem the *fail* class is more important since we have an imbalanced data set and we observed in some of our models that they assigned every sample to the *pass* class. However, we considered that classifying a passing student as a failing student and a failing student as a passing student have the same impact. For this reason, we used the macro average of the F1-score instead of focusing on the metrics of one specific class. This metric tries to improve both *F1-scores*, one of each class. However, we kept the *F1-score* of the classes and accuracy to be able to have at the same time a good idea of how the classification of each class was going on and to compare the accuracies from the ones obtained in the data set's paper.

4.3 Methods

4.3.1 Logistic Regression

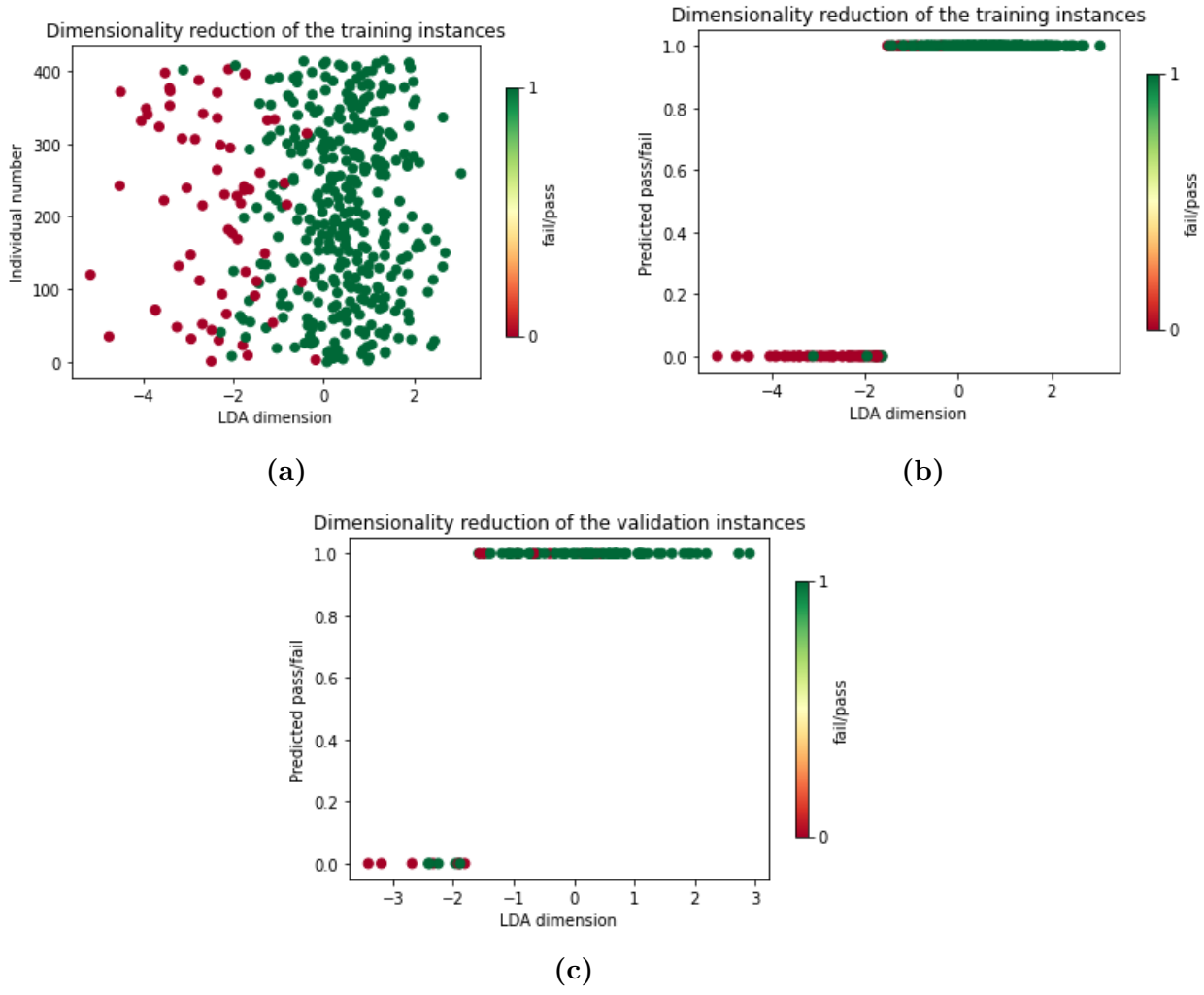
Considering that our approach is a binary classification, the first model we used is the Logistic Regression (LR) with L2(ridge) penalization and a regularization strength of 1. This model has a nice feature of not making the assumption on how data is distributed. We tested several regularization strength setups to obtain the best results. The smaller the value, stronger regularization. This also applies to SVM.

4.3.2 Linear Discriminant Analysis

We used Linear Discriminant Analysis (LDA) as a first approximation for the use of generative classifiers. The LDA implements Bayes Classifier under the assumption of class-conditional distributions $p(\mathbf{x}|y)$ are gaussian, and that all covariance matrices are the same. However, as we saw

in the data explorations, the distribution is not normal. Therefore, this model is not the most adequate to test our assumptions. Nonetheless, LDA can also perform dimensionality reduction, and as we can see in (a) of Figure 4, individuals passed an fail are fairly separated. For this work, we used the singular value decomposition solver, which does not compute the covariance matrix.

Another problem with this model is that since it works with different distances according to the behaviour of the covariance matrices, most of the data is indicative. The implication is that it cannot set a proper distance because a dummy variable only indicates the presence of the variable. Therefore, it will rely on the numerical variables.



(a) **(b)**

(c)

Figure 4: **(a)** Training real data **(b)** Predicted trained data **(c)** Validations of predicted data

4.3.3 Quadratic Discriminant Analysis

We used the Quadratic Discriminant Analysis (QDA) as a way to solve the non-gaussian distribution of our data. This model aims to choose the label with maximum probability a posteriori. In this sense, the classifier uses a different Mahalanobis distance from the matrices for each class center.

4.3.4 Naive Bayes

Another important member of the generative classifiers is the Naive Bayes. This model assumes that features are pair-wise independent in the class conditional probabilities. Furthermore, it is possible to model binary features as Bernoulli distributions. In the case of our data, we tested for three methods: the Gaussian Naive Bayes (GNB), were it is assumed that every feature is numerical (although we already know that this is not true); the Categorical Naive Bayes (CNB), were it only considered the one hot encoded variables, and; a combined version of GNB and CNB by adding the log-likelihoods together. However, in the case of the GNB we also face the same problem as we did in the LDA, therefore we did not expected a very good result.

4.3.5 Decision Trees

We advance now with a discriminative classifier which is Decision Trees (DT). We knew that training DT would be very fast and we could easily interpret the resultant model. However, we had to be careful to not overfit the model. Because of this and to choose the best values for the hyper-parameters, we performed a grid search checked with CV to obtain the parameters with best *F1-score (macro avg)*.

4.3.6 Random Forests

Following the DT, we tried with Random Forests (RF) to improve the DT generalization by generating many simpler trees thus reducing the variance obtained. RF has the *class_weight* parameter that helped us reduce the imbalance of classes at training. We performed grid search to choose the best value for it in combination with the other hyper-parameters.

To validate the model, we didn't use the OOB error that RF gives us since we wanted to compare it with other methods that do not have it. Therefore, to be fair we used the same validation set to compare them.

4.3.7 Support Vector Machines

We considered using Support Vector Machines (SVM) because they can be used to classify non-normal data, which was our case and normally gives good results out of the box. But knowing that SVM is one of the models that can be affected negatively with non-important variables. We decided also to train it with only the variables that were obtained from the feature selection we did in the Feature importance section. To see if we could obtain a better classification.

4.3.8 Multi Layered Perceptron

Finally, we used Multi Layered Perceptron. This neural network is a feed-forward network, which means that is a acyclic directed graph, whose neurons are arranged in layers, there is at least one hidden layer, every layer is fully-connected to the next one, and no connections are allowed within layers. We used back-propagation to train the models.

MLP depends strongly on hyperparameter configuration, this is why our main parameters are composed by 500 epochs (different from the 100 epochs used by Cortez & Silva), using the logistic sigmoid as activation function and weight optimization solver *lbfgs*. For everything else, we applied hyper parameter tuning for the size of the hidden layers and the L2 penalty parameter, which in this context is called α.

We achieved this using a grid search that used a cross validation strategy to evaluate each result of said parameters combination.

5 Results

5.1 Methods

5.1.1 Logistic Regression

We tested for $\lambda \in \{0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10\}$, and we obtained that the configuration with the strongest regularization ($C = 1/\lambda = 0.1$) has the best performance. With this parameters, we obtained a *F1-score* of 95.50% class pass, 73.33% for class fail, macro average of 84.44% and an accuracy of 92.30%.

5.1.2 Linear Discriminant Analysis

For the LDA, with svd solver, we obtained a *F1-score* of 93.25% class pass, 60.0% for class fail, macro average of 76.62% and an accuracy of 88.46%. So far, every result is worst than the LR, which is no surprise for reasons given in section 4.

5.1.3 Quadratic Discriminant Analysis

We tested for different regularizers of QDA, in which the one with value 1 obtained the best performance, getting a *F1-score* of 93.10% class pass, 64.70% for class fail, macro average of 78.90% and an accuracy of 88.46%. This was better than the LDA, but still worse than the LR.

5.1.4 Naive Bayes

Surprisingly enough, after testing the three kinds of Naive Bayes, the best model was the GNB, with a *F1-score* of 92.29% class pass, 68.42% for class fail, macro average of 80.68% and an accuracy of 88.46%. Which in comparison with the QDA, the GNB is better in almost every sense except for accuracy. However, it still stays far from the LR. The worst case was obtained by the CNB, having a class fail *F1-score* of only 37.5%, being the lowest score so far.

Regarding the Combined NB, we tried to tune it by setting threshold of 55.0%, which means that select class 1 is only selected when its combined probability is greater than the 55 percent, which indeed is a relaxed assumption. However, the *F1-score* for class fail managed to outperform the untuned Combined-NB, but not reaching the levels of the GNB.

5.1.5 Decision Trees

With the default configuration DT. With it, we obtained a high *F1-score (macro avg)* (80.19%). However the *F1-score* of class *fail* was very low compared to it (66.67%). Once trained with the best parameters obtained from the grid search. It wasn't able to improve the *F1-score (class fail)* but improved the *F1-score (macro avg)*. However, there were still some important misclassifications.

From observing the decision rules of the DT (in the Annex A, Listing 2), we could interpret what general conclusions the model extracted from the data. Given all the decision rules We could observe that they can be simplified to only four possible paths that offered a different classification, showed in Listing 1. Where we can see that the order of importance was first *G2*, second *G1* and finally *absences*. One interesting observation was that even though that the student could have failed both *G1* and *G2*, if the number of *absences* was lower than 5, the model would be confident that the student would pass the Portuguese subject.

Listing 1: Simplification of the possible paths for the decision rules.

```
if G2 <= 8.5: return 0
if G2 > 8.5: if G1 <= 8.5: if absences > 5: return 0
if G2 > 8.5: if G1 <= 8.5: if absences <= 5: return 1
if G2 > 8.5: if G1 > 8.5: return 1
```

5.1.6 Random Forests

The default RF (i.e., without specifying any hyperparameters) obtained a *F1-score (macro avg)* of 80.85%, of 95.02% for class pass, 66.66% for class fail, and an accuracy of 91.35%. Which was one of the best ones so far, but it wasn't able to improve the results of the LR.

After performing the grid search, it showed us that the best value for the parameter *class_weight* was *balanced_subsample*. Which consists in adjusting weights inversely proportional to class frequencies of the bootstrap sample for every tree grown. For the other hyper-parameters, we obtained that the best values are 20 for *max_depth*, 3 for *min_samples_leaf*, 2 for *min_samples_split* and with 500 *n_estimators*.

The resulting RF was the best model obtained so far with a *F1-score (macro avg)* of 85.94%, *F1-score* of 95.40% for class pass, 76.47% for class fail and 92.30% of accuracy.

By checking which were the most important variables for the best model (in Figure 5), we observed that it was similar to the one obtained with the default RF (Figure 3).

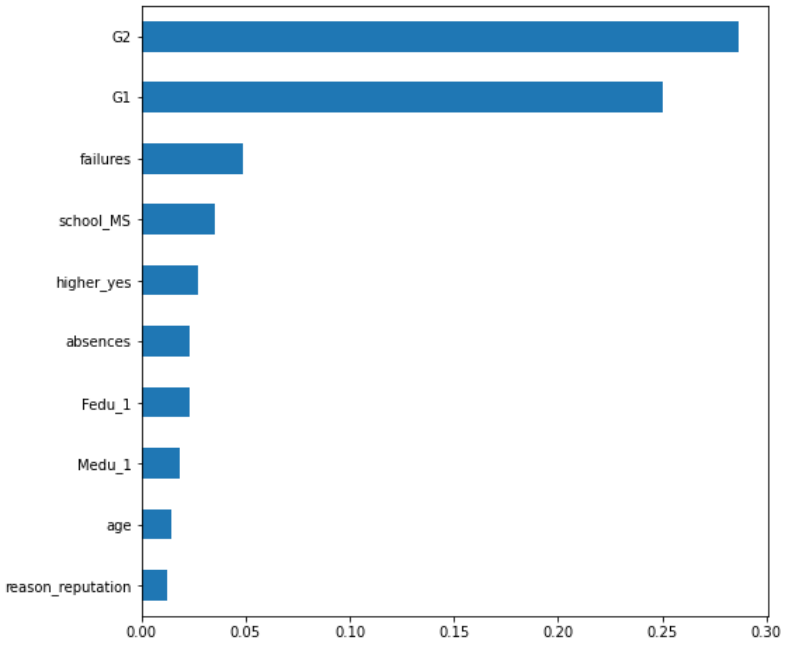


Figure 5: Important variables of the Portuguese data set for a the best Random Forest model obtained.

5.1.7 Support Vector Machines

We started running the SVM with the default parameters and it obtained a *F1-score (macro avg)* of 78%. After doing the grid search to obtain the best parameters we trained the model with a subset of all the variables, the most important ones considered by the RF. With that model, we were able to obtain a *F1-score (macro avg)* of 87.39% which was better. With also an *F1-score* of 96% for class pass, 78.79% for class fail, and 93.26%. Improving the previous results of the RF.

5.1.8 Multi Layered Perceptron

The F1-scores of the MLP are not as good as the SVM and RF. The best model is a two hidden-layers with $\alpha = 1$, with a *F1-score* of 93.02% class pass, 66.67% for class fail, macro average of

79.84% and an accuracy of 88.46%. Barely better than the GNB and slightly worse than the default DT configuration. However, both the version of this MLP, with the Feature Selection training data set, and the MLP with a single layer, obtained a *F1-score* for class fail of 0%, and in consequence, had the worse *F1-score (macro avg)*.

Table 2: Metrics of all the models when predicting the validation data of the Portuguese data set.

	F1-score (class pass)	**F1-score (class fail)**	**F1-score (macro avg)**	**Accuracy**
SVM-FS	0.96	0.787879	0.873939	0.932692
RF-best	0.954023	0.764706	0.859364	0.923077
RF-balance	0.961326	0.740741	0.851033	0.932692
LogReg-0.1	0.955056	0.733333	0.844195	0.923077
SVM-best	0.955056	0.733333	0.844195	0.923077
RF-balance-sub	0.955556	0.714286	0.834921	0.923077
SVM-with-clean	0.959641	0.709677	0.834659	0.929134
RF-default	0.950276	0.666667	0.808471	0.913462
Gaussian-NB-only-numerical	0.929412	0.684211	0.806811	0.884615
DT-best	0.94382	0.666667	0.805243	0.903846
DT-default	0.937143	0.666667	0.801905	0.894231
MLP[2,2]-alpha=1	0.930233	0.666667	0.79845	0.884615
QDA-1	0.931034	0.647059	0.789047	0.884615
SVM-default	0.951351	0.608696	0.780024	0.913462
LDA	0.932584	0.6	0.766292	0.884615
Combined-NB	0.920455	0.5625	0.741477	0.865385
Combined-NB-tuned	0.873418	0.6	0.736709	0.807692
Gaussian-NB	0.886228	0.536585	0.711406	0.817308
Gaussian-NB-only-categorical	0.886364	0.375	0.630682	0.807692
MLP[1]	0.916667	0.0	0.458333	0.846154
MLP[2,2]-alpha=1-FS	0.916667	0.0	0.458333	0.846154

5.2 Test results

Once we computed the validation tests for the trained data (Table 2) and saw that the Support Vector Machine with Feature Selection is the best model overall, we performed the prediction for the test data set of the Portuguese subject to verify a correct generalization of the model. In Tables 3 and 4, we present these results which resulted being better than the validation F1-scores showing a correct generalization.

Table 3: Metrics of the SVM when predicting the test data of the Portuguese data set.

	F1-score (class pass)	**F1-score (class fail)**	**F1-score (macro avg)**	**Accuracy**
SVM-FS	0.972727	0.85	0.911364	0.953846

Table 4: Confusion matrix of the SVM with Feature Selection when predicting the test data of the Portuguese data set.

		Predicted	
		0	**1**
Actual	**0**	17	3
	1	3	107

5.3 Without Outliers

As mentioned before in the Outliers section, we also wanted to check if the prediction would improve, if we removed the students previously considered as possible outliers. Therefore, we trained the best model, SVM-FS, with the data of the Portuguese subject without outliers and tested it with the test partition.

We obtained an SVM model with a *F1-score (macro avg)* of 83.47%, *F1-score* of 95.96 % for class pass, of 70.98 % for class fail, and an accuracy of 92.91 %. Having made no improvement in the classification, shows that the cases harder to correctly classify weren't the students we dropped.

5.4 Math generalization

The final step is to predict Mathematics using the best model obtained from the Portuguese training data set. The results in Table 5 are not better than the Portuguese test results, but better overall than the Portuguese validation. From Table 6, it can be observed that the number of correctly predicted values is balanced and the majority of classification fall in the diagonal. However, there appears to be a higher misclassification of failing students as passing students than the other way.

Table 5: Metrics of the SVM with Feature Selection when predicting the Mathematics data set.

	F1-score (class pass)	**F1-score (class fail)**	**F1-score (macro avg)**	**Accuracy**
SVM-FS	0.955046	0.831276	0.878161	0.896203

Table 6: Confusion matrix of the SVM when predicting the Mathematics data set.

		Predicted	
		0	**1**
Actual	**0**	101	29
	1	12	253

6 Conclusions

We were able to obtain good models for the Portuguese *pass_fail* prediction.

The good generalization for the prediction of the Math data set, lets us conclude that the two subjects do not share too much differences, even thought the distributions for the *G1, G2* and *G3* variables seemed to be different.

The variables that are more important to correctly predict the *pass_fail* for the Portuguese data set are: *G2, G1, failures, absences, higher, school* and *age*.

An important finding is that the most important features does not rely on personal attributes of factors that rarely change. Meaning that the student will be able to change its classification by taking the proactive decision of improving her/his grades.

To finalize, although we used Cortez & Silva work as the starting point of our research, we ultimately focused our results around the *F1-score* metric instead of the accuracy for being more adequate to seek a balanced correct prediction in both classes. Despite the fact that by optimizing the *F1-score* we have been able to indirectly obtain high accuracies.

The original authors used accuracy as the main metric and used the same hyper parameters for both data sets without trying to predict outcomes between one another. For this reason, in Table 7 we present our accuracy metrics for the Portuguese data against the three setup configurations used by Cortez & Silvia (mentioned in section 2). In general, compared with what the authors obtained, our test accuracy of the SVM-FS model obtained the highest accuracy results.

Table 7: Comparation of Cortez & Silva and our accuracy results.

	SVM-FS	**DT (A)**	**RF (B)**	**RF (C)**
Accuracy	95.38	$93.00 \pm_{0.3}$	$90.10 \pm_{0.2}$	$85.00 \pm_{0.2}$

7 Future work

In this work, we had the opportunity to work with survey data and numerical data regarding educational system, and the results with the used models were promising. Nonetheless, there is an open research on the use of these models over representative survey data.

Respecting the models, there is a wide opportunity to explore a higher dimension of hyper parameter tuning configurations for SVM and MLP, given that most of the parementers were left as constant or with few options.

Additionally, it would be interesting to try to perform classification with a clustering method like K Nearest Neighbours or even try to discover possible clusters inside the data and to see if they match with the labels we have.

Lastly, there is still a wide opportunity to explore the multiclass problem with multiclass models in the case of predicting the five level grades, or even predicting regression.

References

[1] I. E. D. M. Society, *Educational data mining*, https://educationaldatamining.org/, (Accessed June 2021). [Online]. Available: https://educationaldatamining.org/.

[2] A. Aleem and M. M. Gore, "Educational data mining methods: A survey," in *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*, 2020, pp. 182–188. DOI: 10.1109/CSNT48778.2020.9115734.

[3] P. Cortez and A. Silva, "Using data mining to predict secondary school student performance," in *Proceedings of 5th FUture BUsiness TEChnology Conference (FUBUTEC 2008)*, A. Brito and J. Teixeira, Eds., 2008, pp. 5–12.

[4] D. Dua and C. Graff, *UCI machine learning repository*, 2017. [Online]. Available: http://archive.ics.uci.edu/ml.

A Decision rules of DT

Listing 2: Decision rules from the best Decision Tree we obtained.

```
if G2 <= 8.5:
    if G1 <= 8.5:
      if freetime_3 <= 0.5:
        return 0
      else:  # if freetime_3 > 0.5
        return 0
    else:  # if G1 > 8.5
      return 0
  else:  # if G2 > 8.5
    if G1 <= 8.5:
      if absences <= 5.0:
        return 1
      else:  # if absences > 5.0
        return 0
    else:  # if G1 > 8.5
      if G2 <= 10.5:
        if health_5 <= 0.5:
          return 1
        else:  # if health_5 > 0.5
          return 1
      else:  # if G2 > 10.5
        if Medu_1 <= 0.5:
          return 1
        else:  # if Medu_1 > 0.5
          return 1
```