Using GitLab Interactions To Predict Student Success When Working As Part Of A Team

Using GitLab Interactions To Predict Student Success When Working As Part Of A Team

Audrey Beatrice Ekuban[1[0000-0002-2261-7218]], Alexander Mikroyannidis[1[0000-0002-9518-1443]], Allan Third[1[0000-0002-0386-1936]] and John Domingue[1[0000-0001-8439-0293]]

[1] Knowledge Media Institute, Open University, Milton Keynes, MK7 6AA, UK
{audrey.ekuban, alexander.mikroyannidis, allan.third, john.domingue}@open.ac.uk

Abstract. This paper explores machine learning algorithms that can be used to predict student results in an assignment of a Software Engineering course, based on weekly cumulative average source code submissions to GitLab. GitLab is a source code version control system, commonly used in Software Engineering courses in Higher Education. The aim of this work is to create models that can be used to predict if a group of students in a team will pass or fail an assignment. In this paper, we present results from Decision Tree, Random Forest, Extra Trees, Ada Boost and Gradient Boosting machine learning models. These models were evaluated using cross-validation, with Ada Boost achieving the highest average score.

Keywords: Learning Analytics, Educational Data Mining, Machine Learning

1 Introduction

Two important areas of research in education are Learning Analytics (LA) and Educational Data Mining (EDM). Learning Analytics, together with Educational Data Mining, allow an institution to gain actionable insights from student data.

1.1 Overview

A regularly used definition of LA is attributed to the Society for Learning Analytics Research (SOLAR) [11]. This is stated as “Learning Analytics is the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs” [10]. In Higher Education, LA very often refers to the use of predictive models in order to optimise student success.

A definition of EDM, which predates LA by a few years, is attributed to the International Educational Data Mining Society (IEDMS) [7]. This is stated as “Educational Data Mining is an emerging discipline, concerned with developing methods for exploring the unique types of data that come from educational settings, and using those methods to better understand students, and the settings which they learn in” [1].

There is a tendency to use LA and EDM interchangeably. However, whereas there are some similarities between them, there are some differences that need to be considered. One of the main differences is that EDM focuses on the creation of machine learning models to extract knowledge from education datasets, whereas LA applies Business Intelligence tools and predictive models to analyse data, in order to optimise learning [2].

In this case study, we are investigating how the results of students' team assignments are impacted by the team's interaction with GitLab. GitLab [4] is a software repository system used by Higher Education establishments. Our aim is to build Machine Learning models that predict student team results for a practical assignment. The dataset referenced in this paper was provided by The University of Manchester [9] (Data Controller) to The Open University (Data Processor). The dataset contained: (i) students' pushes to University of Manchester's software repository, GitLab and (ii) a single Tutor Marked Assignment for each student.

1.2 Research Questions

In this study, machine learning techniques will be used to address two main research questions:

RQ1: How accurately can machine learning classification algorithms identify a team assignment result when using only cumulative average pushes to a software repository in a campus-based undergraduate course at a university?

RQ2: What are the important weeks when using only cumulative average pushes to a software repository in determining team assignment failure in a campus-based undergraduate course at a university?

This case study differs from previous known work [5], where cumulative weekly averages were not considered and Area under the Receiver Operating Characteristics (AUC - ROC) [6] was not one of the metrics used to evaluate model performance.

2 Data Summary

The dataset contained data for 2 cohorts of students on a Software Engineering course that was taught at the University of Manchester. As the cohorts were independent and the data was anonymised independently, it was not possible to tell if there were any students who repeated the course. This paper assumes complete independence, i.e., none of the students had an unfair advantage.

The dataset included one team assignment per cohort. The assignment was set in the first week of the course and marked in the last week of the course. Each student was allocated to a team. Marks were initially allocated to each student based on the team assignment, i.e., all students in a team were given the same mark. If it later transpired that a student did not contribute enough to the team, then that student was given a lesser mark, whilst the marks of the other students in the team are adjusted in the positive direction. The adjusted marks create the situation where the result of student A is dependent upon the result of student B, given that student A and student B are in the same team. This paper details team analysis using the unadjusted marks, i.e. a team is treated as a single entity.

Apart from results, the only other variables were the daily number of times a team submitted (pushed) code to GitLab. The number of students in each team could be calculated. But as there were, roughly, the same number of students in a team, this value was ignored. Hence, the only independent variable was the number of weekly pushes i.e. the weekly pushes were calculated from the daily pushes.

The Data Processor was not provided with the push data of students who withheld consent. As stipulated by the Data Controller, any teams in which at least one student opted-out was not included in the analysis. This is to remove the risk of re-identification [8] of these students. Re-identification is the process by which information about an individual that exists in the public domain can be used to gather some information about that individual from an anonymized dataset.

3 Data Exploration

An initial analysis of the data showed that:

- In Year 1, out of 37 teams
  - 5 teams failed (mark less than 40%)
  - 5 teams passed with a mark of less than 70%
  - 27 teams passed with a mark of 70% or greater
- In Year 2, out of 36 teams
  - 0 teams failed (mark less than 40%)
  - 7 teams passed with a mark of less than 70%
  - 29 teams passed with a mark of 70% or greater

There were a low number of failures in year 1 and 0 failures in year 2. There is insufficient data to undertake any further work that would give insights into teams which attained a mark of less than 40% in year 1. Also, with only 2 years of data, it cannot be ascertained which of the years were “normal”. In order to progress predictive analytics, a team with less than 70% of the marks was deemed to have failed. A team with greater than or equal to 70% was deemed to have passed. Hence overall, there were 17 teams with a fail grade and 56 teams with a pass grade (see Fig. 1).

A graph of the weekly cumulative average of the number of times that teams push to GitLab (see Fig. 2) indicate that the likelihood of a team receiving a result of 70% or more increases with the average number of pushes.

In the further work detailed below, the data for week 13 was removed as this is the week when the assignments are marked.

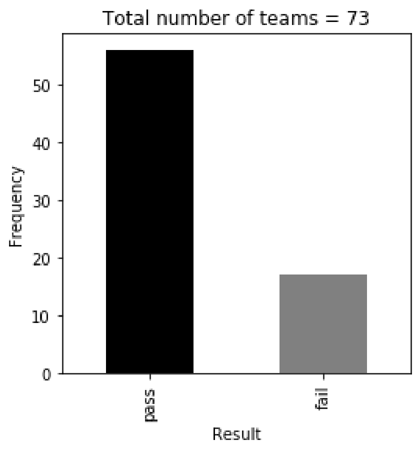

Fig. 1. Bar Plot of Team Results. Pass is attributed to teams with a result of 70% or over. Fail is attributed to teams with a result of less than 70%.

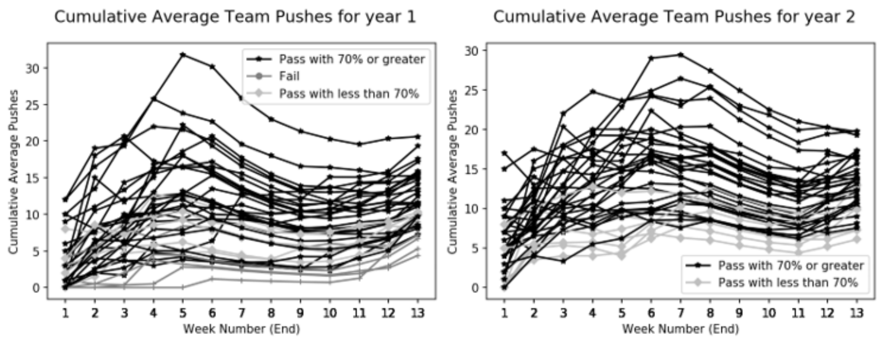

Fig. 2. Weekly Cumulative Average of number of times the teams submitted code to GitLab.

4 Methods

This case study uses a supervised machine learning approach. The goal was to train simple models that could be used to predict the result of a team assignment given the cumulative average number of GitLab pushes per week. The models are to be used to predict the teams that are likely to fail. The output from the classification models are 1 for fail, the positive class, and 0 for pass, the negative class.

The models were created using classifiers as implemented in the Python Scikit-learn package [3]. The following classifiers were investigated:

- Decision Tree
- Random Forest

- Extra Trees
- Ada Boost
- Gradient Boosting

More details are given in the following sections.

4.1 Model Training Considerations

In supervised machine learning, it is normal to split a dataset into training and test data. A model is created with training data, and evaluated with the test, or unseen, data. A model is said to overfit if it performs well on the training data but not on the test data i.e. the model cannot be generalized. In addition to training data (labelled data), many classifiers also make use of hyperparameters. These are parameters that are set prior to the training of the model and are used to tune a model. Model training is therefore an iterative process. However, after the first train/test cycle, the test data is no longer "unseen". Instead of splitting the data into 2, the data could be split into 3 - training, validation and test. In this case, the model is trained on the training data and evaluated using the validation data. In this case study, as the dataset is small, the whole dataset is used in training the model. Rather than manually splitting the data to create a validation set, 5-fold cross-validation is used to select the most suitable hyperparameters for each of the classifiers and to compare the performance of different machine learning models on the dataset. The dataset is divided into 5 parts (folds). In each training iteration, one fold is withheld as the validation set with training being done on the remaining 4 folds. A different fold is withheld for each cross-validation. The accuracy of the model is estimated by averaging the accuracies from all of the 5 cases of cross-validation. It is envisaged that a future year's data will be supplied in order to test the model.

4.2 Model Evaluation

The models were evaluated with a combination of accuracy score and AUROC (Area under the Receiver Operating Characteristics).

In this case study, predicting the fail class is of major importance:

- True Positive = Team fail result is correctly classified as fail
- False Positive = Team pass result is incorrectly classified as fail
- True Negative = Team pass result is correctly classified as pass
- False Negative = Team fail result is incorrectly classified as pass

Accuracy is the fraction of predictions that a classification model got correct i.e.

$$Accuracy = \Pr(correct\ outcome) = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

In binary classification, where there are only 2 labels:

$$Accuracy = \frac{TP + TN}{TP\ +\ FP\ +\ TN\ +\ FN}$$

where

- TP = Sum of True Positives
- TN = Sum of True Negatives
- FP = Sum of False Positives
- FN = Sum of False Negatives

A confusion matrix can be used to display these values.

A perfect model would have an accuracy score of 1 (or 100%). Accuracy can be misleading for imbalanced data sets. Consider a dataset with 95 students, who have a result of pass (positive class) and 5 students who have a result of fail (negative class). A model that classifies all results as pass will have an accuracy of 0.95 (or 95%). By relying on accuracy on its own, this model could end up being selected ahead of more useful models which have a lower accuracy.

AUC - ROC curve

In addition to directly predicting a class, some machine learning models can also predict the probability of a data point belonging to a classification class. The probabilities can be used to determine a classification threshold to interpret the results of a classifier. By default, equal probability is given to classes in binary classifications. In reality, there is usually a trade-off between Sensitivity (missed positives) and Specificity (negatives classed as positives). Sensitivity is also known as recall or True Positive Rate.

$$\begin{aligned} True\ Positive\ Rate\ &= \Pr(Predicted\ Fail|Fail\ Result) \\ &= \frac{TP}{TP + FN} \end{aligned}$$

$$\begin{aligned} \text{Specificity} &= \Pr(Predicted\ Fail|Fail\ Result) \\ &= \frac{TN}{TN + FP} \end{aligned}$$

$$\begin{aligned} False\ Positive\ Rate\ (FPR) &= 1 - Specificity \\ &= \frac{FP}{TP + FN} \end{aligned}$$

With AUC (Area Under The Curve) ROC (Receiver Operating Characteristics) curve, the performance of a classification problem can be visualized. The ROC curve is a probability curve which considers all possible classification thresholds. AUC indicates how much a model can distinguish between classification labels. The higher the AUC, the better the model is at predicting true positives and true negatives. Values range from

0 to 1. 0.5 imply that the model performs no better than random guessing. The ROC curve is plotted with True Positive Rate (TPR) against False Positive Rate (FPR).

5 Discussion

The hyperparameters for the classifiers under consideration were selected based on the best mean AUC-ROC for the cross-validation.

Table 1 displays those mean AUC-ROC scores and standard deviation, along with the mean accuracy and standard deviation. Table 2 displays the AUC-ROC score when the predictions from the model have the mean accuracy. The sections below explore these results in more detail. Table 3 shows that altering the cut-off point of the predicted probability, in this case, 0.4, leads to an increase in Recall for both Ada Boost and Gradient Boosting. However, with this probability, accuracy increases for Gradient Boosting, but decreases for Ada Boost.

Table 1. Mean Scores from 5-fold cross-validation.

Classifier	Mean Accuracy	Mean Recall	Mean AUC-ROC
Decision Tree	0.84 (+/-0.05)	0.67 (+/-0.42)	0.89 (+/-0.09)
Random Forest	0.85 (+/-0.08)	0.77 (+/-0.12)	0.91 (+/-0.09)
Extra Trees	0.85 (+/-0.03)	0.58 (+/-0.15)	0.92 (+/-0.06)
Ada Boost	0.89 (+/-0.09)	0.77 (+/-0.20)	0.93 (+/-0.05)
Gradient Boosting	0.79 (+/-0.05)	0.28 (+/-0.16)	0.90 (+/-0.06)

Table 2. Scores from Cross-validated estimates.

Classifier	Accuracy	Recall	AUC-ROC
Decision Tree	0.84	0.71	0.88
Random Forest	0.85	0.76	0.88
Extra Trees	0.85	0.59	0.89
Ada Boost	0.89	0.76	0.93
Gradient Boosting	0.79	0.29	0.87

Table 3. Scores from Cross-validated estimates with probability threshold of 0.4.

Classifier	**Accuracy**	**Recall**
Decision Tree	0.82	0.71
Random Forest	0.84	0.88
Extra Trees	0.78	0.71
Ada Boost	0.51	1.00
Gradient Boosting	0.84	0.65

5.1 Decision Trees

A decision tree is usually trained by recursively splitting the data. Each split is chosen according to an information criterion which is maximised (or minimised) by one of the splitters. A decision tree is very easy to visualise.

The cross-validation mean accuracy for the Decision Tree Classifier model was 0.84 with a standard deviation of 0.05. The cross-validation AUC-ROC for the cross-validation prediction was 0.93 with a standard deviation of 0.05. The accuracy score was 0.84 with 49 of the 56 team passes and 12 of the 17 team failures being correctly predicted. The most important features were the weekly cumulative averages at week 3, 4, 9 and 11, with week 9 being the most important. The AUC-ROC for the model which gives the mean reported predicted accuracy was 0.88. (See Fig. 3)

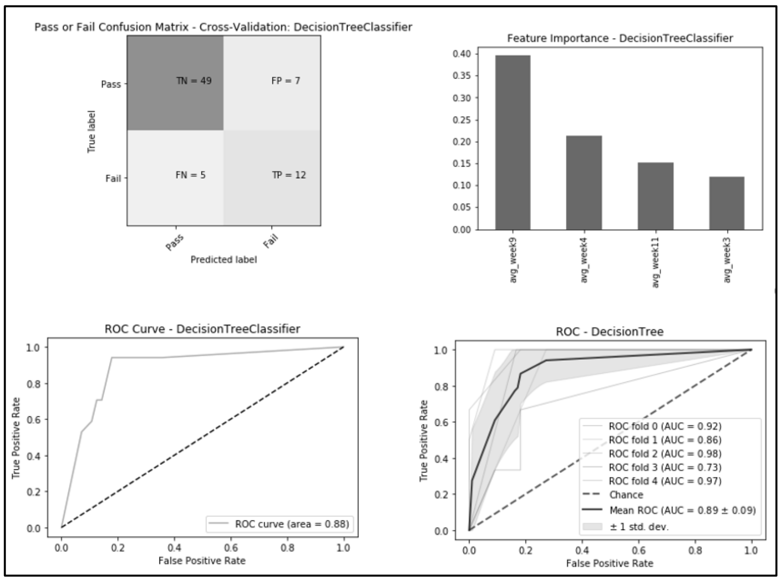

Fig. 3. Confusion Matrix, Important Features, AUC-ROC scores for Decision Tree Classifier.

This model performs well when considering both the individual cross-validation sets and the mean accuracy prediction. With a TPR of 0.71, there is no reason why this model should be rejected.

5.2 Random Forest

A random forest uses an ensemble approach to find the decision tree that best fits the training data. Many decision trees are created with predictions being averaged. Each decision tree is created using a random selection of features. A set of decision trees is a forest. Hence the term random forest.

In this case study, the cross-validation mean accuracy for the Random Forest Classifier model was 0.85 with a standard deviation of 0.08. The cross-validation AUC-ROC was 0.91 with a standard deviation of 0.09.

The accuracy score was 0.85 with 49 of the 56 team passes and 13 of the 17 team failures being correctly predicted. The most important features were the weekly cumulative averages at week 6, 7, 8, 9 and 11, with week 6 being the most important. The AUC-ROC for the model that gives the mean reported predicted accuracy was 0.89. (See Fig. 4).

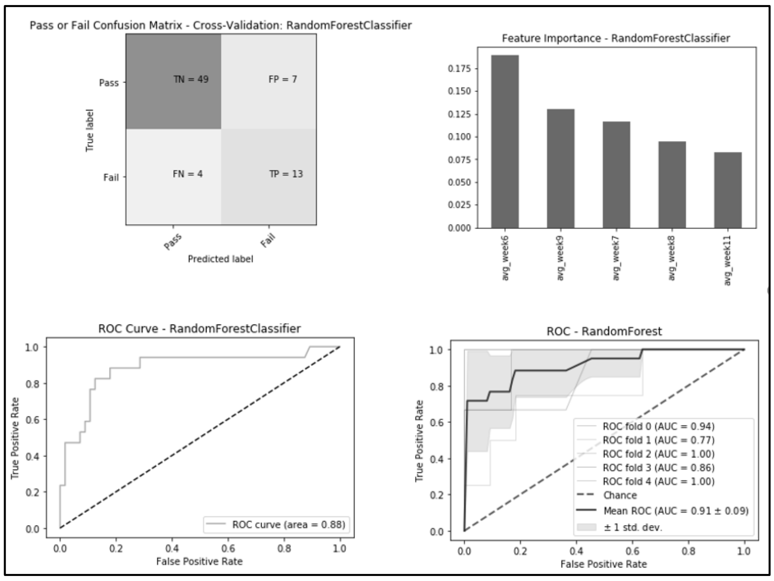

Fig. 4. Confusion Matrix, Important Features, AUC-ROC scores for Random Forest Classifier.

This model performs well when considering both the individual cross-validation sets and the mean accuracy prediction. When compared to the Decision Tree Classifier, this model has the same AUC-ROC score for a higher accuracy. In addition, this model has a higher AUC-ROC mean with the same standard deviation. The Random Forest Classifier appears to be a better model than the Decision Tree model. Therefore, with a TPR of 0.76, there is no reason why this model should be rejected.

5.3 Extra Trees

Extra Trees is short for extremely Randomised Trees. The main difference between random forests and extra trees is that in the random forest, a locally optimal feature/split is used, whereas in extra trees, for each feature under consideration, a random value is selected for the split.

This model gives similar cross-validated scores to the Random Forest Classifier, but for the cross-validated prediction, the Random Forest Classifier has a higher recall. This model is therefore rejected.

5.4 Ada Boost

Adaptive Boosting (Ada Boost1) is a popular boosting algorithm used for classification problems. A sequence of weak learners, e.g. small decision trees, are trained repeatedly on modified versions of the data. The predictions are then combined through a weighted majority vote to produce a final prediction.

In this case study, the cross-validation mean accuracy for the Ada Boost Classifier model was 0.89 with a standard deviation of 0.09. The cross-validation AUC-ROC was 0.93 with a standard deviation of 0.05.

The accuracy score was 0.89 with 52 of the 56 team passes and 13 of the 17 team failures being correctly predicted. The most important features were the weekly cumulative averages at week 1, 6, 3, 11 and 2, with week 1 being the most important. The AUC-ROC for the model that gives the mean reported predicted accuracy was 0.89. (See Fig. 5).

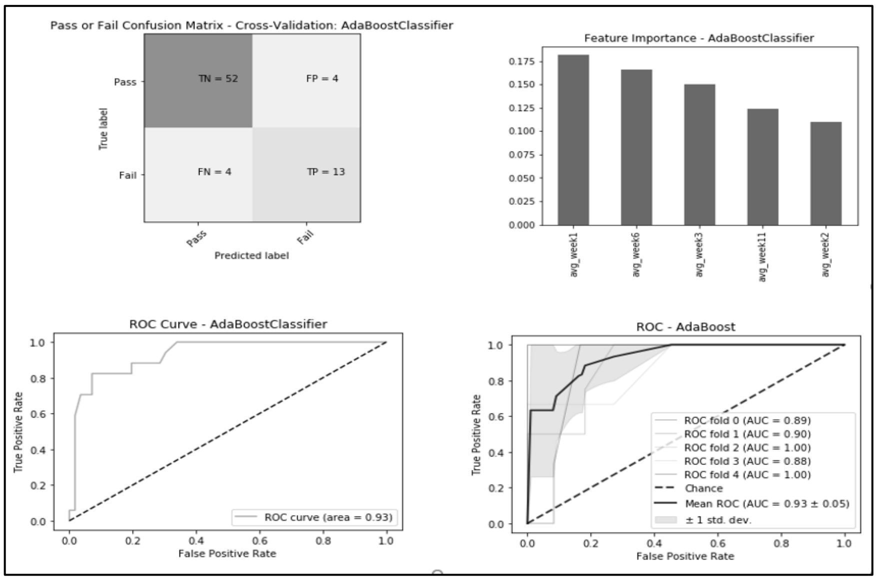

Fig. 5. Confusion Matrix, Important Features, AUC-ROC scores for Ada Boost Classifier.

This model performs extremely well when considering both the individual cross-validation sets and the mean accuracy prediction. With a TPR of 0.76, there is no reason why this model should be rejected.

5.5 Gradient Boosting

Gradient Boosting is an Additive Model, i.e., it uses an iterative and sequential approach of adding the weak learners. Each iteration should reduce the value of a loss function.

In this case study, the cross-validation mean accuracy for the Gradient Boosting Classifier model was 0.79 with a standard deviation of 0.05. The average cross-validation AUC-ROC was 0.9 with a standard deviation of 0.06. Considering the mean accuracy and recall scores, this model does not perform well. There is therefore reason to reject this model.

6 Conclusion

This work answered the following research questions:

- How accurately can machine learning classification algorithms identify a team assignment result when using only cumulative average pushes to a software repository in a campus-based undergraduate course at a university?
- What are the important weeks when using only cumulative average pushes to a software repository in determining team assignment failure in a campus-based undergraduate course at a university?

The process started with the collection of data from University of Manchester. This data was then pre-processed to remove the teams which contained students who had opted out. The weekly cumulative averages were then calculated for each of the remaining teams. Five machine learning algorithms were chosen to explore the data with. They were Decision Trees, Random Forests, Extra Trees, Ada Boost and Gradient Boosting. The chosen evaluation metrics were accuracy, recall and AUC-ROC. As the dataset was small, a 5-fold cross-validation approach was used.

The 3 best machine learning classifiers were Decision Trees, Random Forests and Ada Boost:

- Decision Tree
  - Accuracy = 0.84
  - Recall = 0.71
  - AUC-ROC = 0.88
- Random Forest
  - Accuracy = 0.85
  - Recall = 0.76
  - AUC-ROC = 0.88
- Ada Boost

- ○ Accuracy = 0.89
- ○ Recall = 0.76
- ○ AUC-ROC = 0.93

There is no real consensus between the 3 selected models on the important weeks. The 2 most important features are 1) weeks 4 and 9 for the Decision Tree model, 2) weeks 6 and 9 for the Random Forests model and 3) weeks 1 and 6 for the Ada Boost model.

These models will be further evaluated using data from a future year as a test set.

7 Acknowledgements

This work was supported by the Institute of Coding, an initiative funded by the UK Office for Students (OfS). The Dataset for this work was provided by the University of Manchester.

References

1. Baker, R.S., Yacef, K.: The State of Educational Data Mining in 2009: A Review and Future Visions. Journal of Educational Data Mining 1(1), 3–17 (2009)
2. Chen, G., Rolim, V., Mello, R.F., Gaˇsevi´c, D.: Let's shine together! a comparative study between learning analytics and educational data mining. In: Proceedings of the Tenth International Conference on Learning Analytics and Knowledge, LAK '20, p. 544–553. Association for Computing Machinery, New York, NY, USA (2020)
3. Fabian Pedregosa, Ga¨el Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and ´Edouard Duchesnay. Scikit-learn: Machine learning in python. J. Mach. Learn. Res., 12(null):2825–2830, November 2011.
4. GitLab Homepage: https://about.gitlab.com/. Last accessed 03-July-2020
5. Guerrero-Higueras, A., Matella´n-Olivera, V., Esteban-Costales, G., Fern´andezLlamas, C., Rodr´ıguez-Sedano, F., ´Angel, C.: Model for evaluating student performance through their interaction with version control systems. Learning Analytics Summer Institute (LASI), SNOLA (2018)
6. Hanley, J., McNeil, B.: A method of comparing the areas under receiver operating characteristic curves derived from the same cases. Radiology, 148(3), 839-843 (1983).
7. IEDMS Homepage: https://educationaldatamining.org/. Last accessed 03-July2020
8. Lubarsky, B.: Re-identification of "anonymized data". UCLA L. REV 1754(2010) (1701)
9. Rıos, J.C.C., Kopec-Harding, K., Eraslan, S., Page, C., Haines, R., Jay, C., Embury, S.M.: A methodology for using gitlab for software engineering learning analytics. CoRR abs/1903.06772 (2019)
10. Siemens, G.: Learning analytics: Envisioning a research discipline and a domain of practice. In: Proceedings of the 2nd International Conference on Learning Analytics and Knowledge, LAK '12, p. 4–8. Association for Computing Machinery, New York, NY, USA (2012)
11. SOLAR Homepage: https://www.solaresearch.org/. Last accessed 03-July-2020