Using natural language processing to predict student problem solving performance

Jeremy Munsell,
Department of Physics and Astronomy, Purdue University, 525 Northwestern Ave., West Lafayette, Indiana, USA, 47906

N. Sanjay Rebello
Department of Physics and Astronomy, Purdue University, 525 Northwestern Ave., West Lafayette, Indiana, USA, 47906
Department of Curriculum and Instruction, Purdue University,100 N. University St., West Lafayette, Indiana, USA, 47906

Carina M. Rebello
Department of Physics and Astronomy, Purdue University, 525 Northwestern Ave., West Lafayette, Indiana, USA, 47906

In this work we report on a pilot study where we used machine learning to predict whether students will correctly solve the classic "ballistic pendulum" problem based on an essay written by students elucidating their approach to solving the problem. Specifically, students were asked to describe the "principles, assumptions, and approximations" they used to solve the problem. Student essays were codified using the practices of natural language processing. Essays from two non-consecutive semesters were used for training/validation (N = 1441) and testing (N=1480). The final model used to make predictions was an ensemble classification scheme using random forest, eXtreme Gradient Boosting classifier (XGBoost), and logistic regression as estimators. Our accuracy in predicting students' correctness was around 80% with slightly higher accuracy in identifying students who incorrectly solved the problem and slightly lower in predicting student who correctly solved the problem.

2021 PERC Proceedings edited by Bennett, Frank, and Vieyra; Peer-reviewed, doi.org/10.1119/perc.2021.pr.Munsell

I. INTRODUCTION

Research has shown that facilitating students to attend to the underlying concepts and principles needed to solve a problem improve problem solving performance [1,2]. We implemented strategy writing [2] in a pilot study with students in a calculus-based physics course at a large public mid-western university. Students were asked to write an essay describing their strategy for solving a problem. Their essays were analyzed using Natural Language Processing (NLP) to determine whether they could predict the ground truth label i.e. the correctness of the student's answer to the problem.

NLP is a branch of artificial intelligence (AI) in which computers perform operations on human language. NLP has numerous applications such as determining the sentiment of tweets; chatbots/assistants which perform speech recognition/generation; and machine text translation. Classification in NLP is at the intersection of machine learning and NLP. Machine learning (ML) can be thought of as a collection of methods where a statistical model is developed that maps numerical data on to a target variable (label). A ML algorithm is trained when an objective function which quantifies the error made by incorrect predictions is minimized with respect to the model's parameters (e.g. weights and biases in the case of multiple linear regression). The trained model is then used to predict the class membership of unseen data known as a testing set. The fundamental rule of ML is testing data is not used for training or any manner of model parameter tuning.

In this work we report on the use of NLP to predict whether students in a first semester calculus-based course would correctly solve a problem (Fig. 1) during a quiz taken in lab. We asked students to write an essay describing their strategy for solving the problem, including underlying principles used, and objects in the system/surroundings. Data were labeled 0/1 based on whether students solved the problem incorrectly/correctly. This work was exploratory in nature to determine how well we could make accurate predictions. Our vision for the future of this work is a platform to provide in-situ feedback to improve student learning.

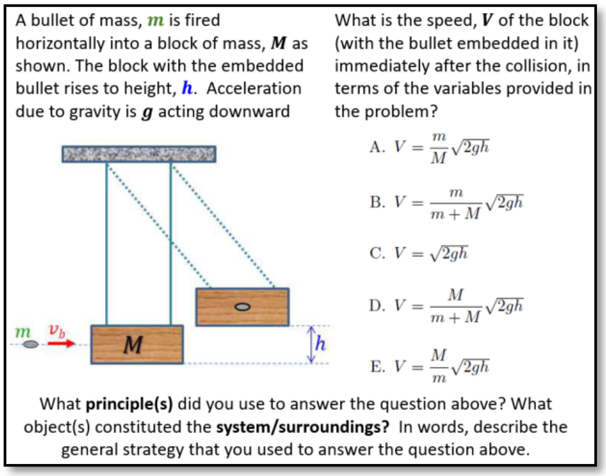
The text data from the essay were transformed using the term frequency-inverse document frequency (TFIDF) method. We constructed a ML model using the Scikitlearn [3] library in Python. The final prediction model was a hard voting scheme using Random Forest [4], Logistic Regression, and eXtreme Gradient Boosting classifier [5] as estimators. We used data from Spring 2020 for model training and general validation, and data from Spring 2021 for testing. More details are presented in the following sections.

We addressed the following **Research Question**: With what accuracy can we predict if a student will correctly solve the "ballistic pendulum" problem given an essay outlining the student's strategy?

II. METHODS

A. Task

Students completed the task shown in Fig. 1 on Quiz 3, which was administered in Week 7 of the semester. The quiz was administered in a sterile environment where notes and collaboration were not allowed. We chose this problem because it is a well-known problem in introductory physics that students have difficulties with.



A bullet of mass, m is fired horizontally into a block of mass, M as shown. The block with the embedded bullet rises to height, h. Acceleration due to gravity is g acting downward

What is the speed, V of the block (with the bullet embedded in it) immediately after the collision, in terms of the variables provided in the problem?

A. $V = \frac{m}{M}\sqrt{2gh}$

B. $V = \frac{m}{m+M}\sqrt{2gh}$

C. $V = \sqrt{2gh}$

D. $V = \frac{M}{m+M}\sqrt{2gh}$

E. $V = \frac{M}{m}\sqrt{2gh}$

What **principle(s)** did you use to answer the question above? What object(s) constituted the **system/surroundings?** In words, describe the general strategy that you used to answer the question above.

FIG 1: Problem solved by students in online Quiz 3 in Week 7

B. Data

The descriptive statistics for the word length of the essay data are shown in Table I. A thorough analysis of the differences between the words and phrases used by each group is beyond the scope of this paper. There is no significant difference in essay length between the correct and incorrect responses, or between the data sets.

TABLE I. Descriptive Statistics for Length of the Essays

Data Set	Mean ± S.D.	Median
Spring 2020	Correct (N_1=703): 57.8±31.2	51
(Training)	Incorrect (N_0=738): 56.5±29.9	51
Spring 2021	Correct (N_1=679): 60.2±32.0	55
(Testing)	Wrong (N_0=801): 59.8±37.3	52

C. Text Processing

I. Text Cleaning

The essays from both sets were cleaned using a function in Python, that removes unimportant commonly used words (stop words) [6] to reduce noise, as well as punctuation, numbers, and equations which some students (6.1% in training, 4.5% in testing set) included in the essay. Finally, the essays were spell checked using a context-unaware spell checker from the textblob [7] library.

II. TFIDF transformation

ML algorithms cannot perform computation on raw text. Most standard methods in NLP involve transforming text into a vector. The simplest approach is the bag-of-words model in which text is transformed into a vector of dimensionality equal to the number of unique words in the corpa and whose components are the word counts in a particular corpus. A higher level of sophistication is the TF-IDF transformation, which converts each essay (corpus) into a vector whose dimensionality is the number of unique words in all the essays (corpa). The components of each vector are a calculated score for each unique word in the corpa based on its frequency of appearance in that corpus and inverse frequency in the corpa:

$$W(t,d,D) = f_{t,d} \log\left(\frac{N}{n_t}\right)$$

The TFIDF score, W, for each word, t, is calculated corpus-wise for each document d in the corpa D. W is large for words with a high frequency (f) appearing in a small number of documents (n_t). W is low for words that have low frequency appearing in a large number of documents.

D. Prediction Model

The prediction model uses three independent estimators, Random Forest [4], eXtreme Gradient Boosting (XGBoost) [5], and Logistic Regression. The predictions emerging from these algorithms are combined to make a single final prediction, a scheme known as ensemble learning.

I. Random Forest Classifier

A decision tree is a flowchart like structure where a datum is classified after passing through a network of nodes representing features of the model. In some cases, the decision tree can be conceptualized as a series of yes/no questions that ultimately results in a classification [8]. Decision trees are robust to irrelevant features (noise) and are capable of learning complex patterns. However, they tend to learn the training set very well while struggling with unseen testing data (overfitting).

The random forest classifier is an ensemble (forest) of decision trees [4]. Each tree in the forest is built by randomly sampling the training data with replacement, a method known as bootstrap aggregation, and using a random subset of the features (variables) to make predictions. The final classification is the majority vote of all the trees. This has the effect of reducing overfitting relative to a single decision tree by producing a series of weak uncorrelated learners which averaged together make more accurate predictions [9].

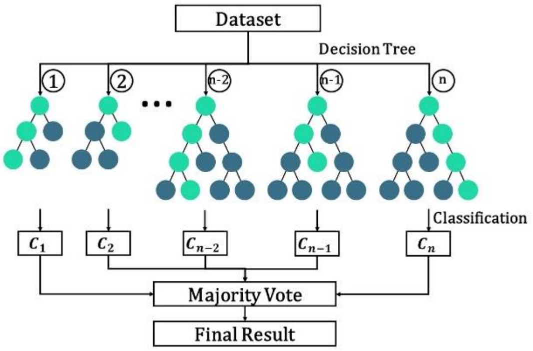


FIG 2: The figure shows a random forest classifier where n decision trees are generated from a random subset of the data, using a random subset of features. Each tree makes an independent classification and the final result is the majority vote.

II. eXtreme Gradient Boost (XGBoost)

Boosting is a technique whereby the classifier learns from its' mistakes (incorrect predictions) [5]. The version of XGBoost used in this work is based on the random forest classifier. XGBoost uses boosted tree learning to improve upon the consistently high performance of random forest. The goal of XGBoost is to learn a decision function (classifier) that encapsulates the structure and function of a random forest. Boosting happens in iterations called boosting rounds. The decision function is initialized to a constant value, obtained by solving an optimization problem. During each of the m subsequent boosting rounds the decision function is updated recursively to correct mistakes made in the previous round. This scheme results in a classification algorithm that is robust to overfitting but can be susceptible to outliers [10]. For labeled data $\{x_i, y_i\}$, the decision function F_m after the *m-th* boosting round, and the objective function (error function) *L:*

$$F_m(x) = F_{m-1}(x) - \gamma_m \sum_{i=1}^{n} \nabla F_{m-1} L\big(y_i, F_{m-1}(x_i)\big)$$

While the usual gradient descent algorithm that is at the heart of machine learning aims to minimize the objective function with respect to the parameters of the decision function, gradient boosting endeavors to minimize the objective function with respect to the decision function.

III. Logistic Regression

In logistic regression we predict samples using the sigmoid function:

$$h(x) = \frac{1}{1+e^{-\boldsymbol{\theta x}}}$$

Where $\boldsymbol{\theta}$ is a vector of weights and biases (high dimensional analog to slope and intercept) and $\mathbf{x}$ is a feature vector (data point). The vector $\boldsymbol{\theta}$ is obtained by minimizing the log-loss objective function with respect to $\boldsymbol{\theta}$.

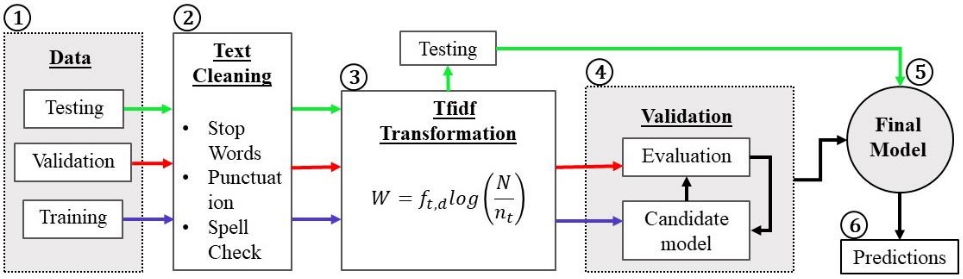


FIG. 3. A diagram showing the machine learning workflow. (1) Training data (blue), validation data (red), and testing data (green) are processed by (2) removing stop words, punctuation, and checking spelling. (3) A Tfidf transformer object is fitted to the training data and used to transform training, validation, testing sets. The testing set is put aside. The training set is used to train a candidate model, and the candidate model is evaluated on the validation set. (4) The model is tuned in a feedback loop to improve classification performance on the validation set. The process continues until performance is saturated and the final model (5) emerges. The training and validation sets are used to train the final model and (6) predictions are made on the testing set

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} \times \log\left(h_\theta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \times \log\left(h_\theta\left(x^{(i)}\right)\right)\right]$$

The sigmoid function is a continuous valued function bounded on (0,1). When making a binary classification, a thresholded decision function $h'(x)$ is used such that:

$$h'(x) = \begin{cases} 1 & if\ h(x) \geq 0.5 \\ 0 & if\ h(x) < 0.5 \end{cases}$$

E. Training, Validation, and Testing

Model training is the process (Fig. 3) of using the training data to select the optimal parameters for a given model. The optimal parameters are usually determined by minimizing an objective function with respect to the model parameters. This gives a candidate model. The success of a model is determined by its ability to correctly classify unseen data. The hypothetical scenario is that the testing set is not available to you when you create the model, and it will be used in production to classify new data in real time. Thus, it is necessary to validate the model before production on some data that was not used during training (validation set).

In k-fold validation, we split all the data into k equal sized partitions. k-1 sets are used for training and the remaining set is used for testing. This is repeated until all k sets have been used in training and testing. The accuracy is averaged across the k trials.

III. RESULTS

The classification accuracy is an important metric by which to judge the performance of the prediction model. However, accuracy should not be considered in isolation. Other important metrics to consider are precision, recall, and F-score.

We define a true positive (t_p) classification as a student who is labeled '1' and is predicted as '1', likewise a false positive (f_p) classification is a student is labeled as '0' but predicted as '1'. We define a true negative (t_n) as a student who is labeled as '0' and predicted as '0', likewise a false negative (f_n) is a student who is labeled '1' but predicted as '0'.

Precision is the fraction of correct classifications made by the classifier.

$$P_1 = \frac{t_p}{t_p + f_p} \qquad P_0 = \frac{t_n}{t_n + f_n}$$

Recall is the fraction of each population correctly identified by the classifier.

$$R_1 = \frac{t_p}{t_p + f_n} \qquad R_0 = \frac{t_n}{t_n + f_p}$$

The F-score is the harmonic mean of precision and recall. F-score is a balanced metric to determine the overall quality of the classifier.

$$f_1 = 2\frac{P_1\,R_1}{P_1 + R_1} \qquad f_0 = 2\frac{P_0\,R_0}{P_0 + R_0}$$

Cohen's kappa [11] is a measure of agreement between raters, controlling for agreement by chance.

$$\kappa = \frac{p_0 - p_e}{1 - p_e}$$

Where p_0 is the observed agreement between raters, and p_e is the probability of agreement by chance.

The results of the classification are in Tables II and III below.

TABLE II. Precision, Recall, f-score, and Average Accuracy

Class	N	Precision	Recall	F-Score	Accuracy
0	801	0.79	0.87	0.82	0.80
1	679	0.82	0.72	0.77	

TABLE III. Confusion matrix: Correct predictions are on the diagonal. Incorrect predictions are off diagonal.

	Predicted Negative	Actual Positive
Actual Negative	695	106
Actual Positive	190	489

Finally, Cohen's kappa was calculated to be $\kappa = 0.594$, and 5-fold validation accuracy is 0.78.

IV. DISCUSSION

A proposed instrument for essay scoring should only be deployed if it is shown to be valid, fair, and reliable. A method is considered valid if it measures what it claims to measure. A method is fair if it does not unfairly penalize correct responses, and it is reliable if the results are repeatable [12]. It is difficult to gauge the validity of this model without a direct comparison with other models which perform the same function. Many projects that attempt automatic essay scoring (AES) use comparison with human raters as a metric [13]. A competition among commercial AES vendors used eight student essay corpa from six member states of the Race-To-The-Top assessment consortium as a dataset [13]. Students wrote persuasive, expository, narrative, and source-based essays (where they formulated an argument based on a passage). This dataset used the state adjudicated score conferred by human scorers (resolved score) as the ground truth (label) and compared the performance between different proprietary scoring engines. A metric used in this study is percent agreement between computer scoring systems and the resolved score. Percent agreement (identical to accuracy) is the percentage of times the resolved score and the computer score were identical. The percent agreement of the scoring engines ranged from 0.29 to 0.76, and the Cohen's κ ranged from 0.04 to 0.84 across eight datasets. Thus, our results (accuracy = 0.80, and Cohen's $\kappa = 0.594$) are within the range of proprietary scoring engines used in [13]. A key difference with our study is that in [13] the essays themselves were scored by a multi-point rubric, while we did not score the essays per se, rather we used problem correctness (0/1) as a proxy for scoring of the essays themselves. It is also worth noting that the scoring engines in [13] had high performance on "adjacent agreement" when the computer score was within 2 points of the resolved score on a rubric of 8 points (maximum). There is no way to directly compare our results on this metric due to the differences in essay scoring.

Presently, there is not enough information to establish that our prediction model is valid for scoring student essays themselves. However, the goal of the present study was to use the strategy essay to predict if the student will correctly solve a problem. If we could substantially reduce the error rate, this model could be useful to provide feedback to students so they can correct errors before submission.

In regards to fairness, about 20% of students were incorrectly scored, out of which 13% were predicted incorrect despite solving the problem correctly. Finally, since we currently only have two sets of data to work with, we cannot establish the reliability of this model.

V. CONCLUSIONS, LIMITATIONS & IMPLICATIONS

Despite the shortcomings of our classification scheme these results are promising since the model is able to predict, based on the strategy essay written by a student, whether or not the student has answered the problem correctly with 80% accuracy. For the purposes of predicting incorrect answers, the prediction rate is 87%.

This study has the following limitations. First these results leave room for improvement in accuracy and fairness, which could be achieved with a larger training set and more powerful state-of-the-art machine learning methods, such as deep learning. Second, the study used only a single problem that required students to determine their answer in symbolic representation using a multiple-choice format. Therefore, the results are not generalizable to problems in other formats and representations, not to mention other topical areas in introductory physics. Finally, the study did not score the essays themselves, rather it predicted the scores of the problem that students wrote the strategy essay for, and they may have written this essay not necessarily before solving the problem. Future studies will require students to write the essays before they provide a solution. Further, we will use human raters to score the essays based on the validity of the outlined approach, and therefore the likelihood that the strategy will lead to a correct solution.

Despite these limitations, the study has several implications for research and education. This study provides proof-of-concept that it is possible to predict students' correctness of a problem with a high degree of accuracy, based on the essay they have written describing their strategy to solve the problem. Research has shown that asking students to describe their strategies for solving problems can be useful in helping them develop more expert-like problem solving strategies [2]. However, past studies did not provide feedback to students on their strategy writing. The time cost of providing such feedback, especially in large enrollment introductory classes can be prohibitive. The results of this study are promising because they provide proof of concept that it might be possible, using NLP methods to provide students feedback on their strategy writing in real time, thereby giving them the opportunity to reflect on, and if necessary, alter their problem-solving strategy before they apply it to solve the problem. Such a system would also allow us to investigate whether real time strategy feedback can improve students' metacognitive skills and make them more expert-like problem solvers in the future.

ACKNOWLEDGMENTS

This work is supported in part by U.S. National Science Foundation grant 1712201. We would also like to thank Dr. Tianlong Zu for his excellent suggestions about this study.

[1] R. Dufresne, W. Gerace, J. Mestre, and P. Hardiman, J. Learn. Sci. **2** (3), 307 (1992)

[2] W. Leonard, R. Dufresne, and J. Mestre, Am. J. Phys. **64** (12), 1495 (1996)

[3] Pedregosa, F, Varoquaux, G.,Gramfort, A., Michel, V., Thiron, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., Journal of Machine Learning Research, **12,** p. 2825-2830 (2011)

[4] Tin Kam Ho, *Proceedings of 3rd International Conference on Document Analysis and Recognition*, p. 278-282, **1** (1995)

[5] Chen, Tianqi and Guestrin, Carlos, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (2016)

[6] Bird, Steven, Edward Loper and Ewan Klein, *Natural Language Processing with Python*. (O'Reilly Media Inc., 2009)

[7] Loria, S. textblob Documentation. *Release 0.15*, *2*. (2018)

[8] Quinlan, J. R. International Journal of Man-Machine Studies **27** (3). p. 221-234 (1987)

[9] Jyiu, T., Towards Data Science. (June 12, 2019)

[10] J.H. Freidman, The Annals of Statistics, **29** (5), p. 1189 – 1232, (2001)

[11] McHugh M. L., Biochemia medica, **22** (3), p. 276–282 (2012)

[12] Chung, Gregory K.W.K., and Baker, E.L., Automated Essay Scoring: A Cross-Disciplinary Perspective. In *Issues in the Reliability and Validity of Automated Scoring of Constructed Responses* (Lawrence Erlbaum Associates, Mahwah, New Jersey, 2003)

[13] Shermis, M.D., Assessing Writing, **20**, p. 53–76. (2014)