mathematics

Article

Item Difficulty Prediction Using Item Text Features: Comparison of Predictive Performance across Machine-Learning Algorithms

Lubomír Štěpánek [1,2,*], Jana Dlouhá [1,3] and Patrícia Martinková [1,4]

[1] Institute of Computer Science of the Czech Academy of Sciences, 182 07 Prague, Czech Republic; dlouha@cs.cas.cz (J.D.); martinkova@cs.cas.cz (P.M.)
[2] First Faculty of Medicine, Charles University, 121 08 Prague, Czech Republic
[3] Faculty of Arts, Charles University, 116 38 Prague, Czech Republic
[4] Faculty of Education, Charles University, 110 00 Prague, Czech Republic
* Correspondence: lubomir.stepanek@cs.cas.cz

Abstract: This work presents a comparative analysis of various machine learning (ML) methods for predicting item difficulty in English reading comprehension tests using text features extracted from item wordings. A wide range of ML algorithms are employed within both the supervised regression and the classification tasks, including regularization methods, support vector machines, trees, random forests, back-propagation neural networks, and Naïve Bayes; moreover, the ML algorithms are compared to the performance of domain experts. Using f-fold cross-validation and considering the root mean square error (RMSE) as the performance metric, elastic net outperformed other approaches in a continuous item difficulty prediction. Within classifiers, random forests returned the highest extended predictive accuracy. We demonstrate that the ML algorithms implementing item text features can compete with predictions made by domain experts, and we suggest that they should be used to inform and improve these predictions, especially when item pre-testing is limited or unavailable. Future research is needed to study the performance of the ML algorithms using item text features on different item types and respondent populations.

Keywords: text-based item difficulty prediction; text features and item wording; machine learning; regularization methods; elastic net regression; support vector machines; regression and decision trees; random forests; neural networks; algorithm vs. domain expert's prediction performance

MSC: 62H12; 62H30; 68T50

check for updates

Citation: Štěpánek, L.; Dlouhá, J.; Martinková, P. Item Difficulty Prediction Using Item Text Features: Comparison of Predictive Performance across Machine-Learning Algorithms. *Mathematics* **2023**, *11*, 4104. https://doi.org/10.3390/math11194104

Academic Editors: Huawen Liu, Gaige Wang and Ioannis G. Tsoulos

Received: 1 July 2023
Revised: 8 September 2023
Accepted: 22 September 2023
Published: 28 September 2023

1. Introduction

In educational assessment, the analysis of test items is crucial for designing reliable, valid and fair tests. Item difficulty, the most important item characteristic, is commonly estimated using classical test theory (CTT) and item response theory (IRT) models based on test-taker responses [1]; however, item pre-testing is not always possible, or it may be limited, e.g., due to security or legal reasons. In such situations, automated estimation of item difficulty based on their wording can inform test construction.

Various properties of text wording of a given test item determine how difficult the item is for a test-taker. The item text features, such as length, word frequencies related to established corpora, characteristics of linguistic similarities, and readability indices, can be used to predict item difficulty using machine learning (ML) algorithms. ML and natural language processing (NLP) are already used in different areas of education for automated essay or item scoring [2–4], automated item generation [5–9], data-driven intelligent tutoring systems [10], online proctoring and cheating detection [11–13], and in other situations [14–17]. In addition to commonly used methods such as linear regression or decision trees [18], regularization approaches and neural networks are sometimes used to estimate the item difficulty from item wording based on item features [19]. A wide range of ML algorithms has been used in this context in the past [18,20]. However, their predictive performance is usually not compared; moreover, ML algorithms are rarely compared to the performance of domain experts, which is crucial to determine to what extent the ML algorithms are capable of improving the predictive accuracy of human raters. This is an area of focus in the study.

To address this gap, we introduce a framework for predicting item difficulty using textual features from item wording. We assess the predictive accuracy of multiple ML methods, and we compare them with the predictions made by domain experts. The tools of choice for the prediction we apply on the item features are supervised ML regression methods, namely regularization techniques—such as the least absolute shrinkage and selection operator, ridge regression and elastic net regression—support vector machines, regression trees, random forests, and artificial neural networks with back-propagation [9,21]. We predict the item difficulty as a continuous dependent variable, as it would be returned from student response data. Furthermore, switching the same algorithms into a classification fashion, we predict the membership of each item in one of the predefined difficulty intervals. We assume that classification into one of a few item difficulty intervals could be easier and more accurate for the algorithms than predicting a precise difficulty point value. We hypothesize that ML algorithms are able to compete with human domain experts in predicting (or classifying) item difficulty and that they may further inform and improve the experts' predictions.

The paper proceeds as follows. We start by describing the data preparation needed for the implementation of ML algorithms on cognitive test items, including the text preprocessing and extraction of item features. We then describe the ML algorithms used in this study in Section 2, *Materials and Methods*. We briefly describe applied software, model architecture, algorithms' pre-setting, and tuning parameter values in Section 3, *Implementation*. Next, in Section 4, *Results*, we describe the results, namely the comparison of the accuracy of item difficulty predictions returned by different artificial ML algorithms and those performed by domain experts. Finally, we discuss the key findings in Section 5, *Discussion*, and offer some deductions in Section 6, *Conclusions*.

2. Materials and Methods

A description of a dataset we used for item difficulty prediction and of the applied ML algorithms we built on follows.

2.1. Dataset and Item Text Processing

For this study, we use item wordings from the English as a foreign language test administered over eight years (2016–2023) as a part of the Czech matura exam. We use items from reading comprehension sections containing multiple-choice items with a single-paragraph passage and four response options, denoted as Section 5. We also utilize a dataset of test-takers' answers for the calculation of difficulty for each item as described in more detail in the next section. Finally, item difficulty evaluation by domain experts comes from another (internal) dataset.

Item text wordings are extracted from portable document format-based files (with suffixes .pdf) using optical character recognition (OCR). Then, we apply the scraping methods employing empirical approaches such as regular expressions' masking, by which we obtain an unstructured text for each item's wording, split into item passage, item question, key option (the correct answer), and distractors (incorrect answers). Next, the text is tokenized, i.e., sentences are split into atomic parts (*tokens*), in this case, words. In the next step, stopwords and special characters are removed, and the tokens are lemmatized, i.e., they are transformed into their corresponding lemmas [22], as schematically indicated in Figure 1.

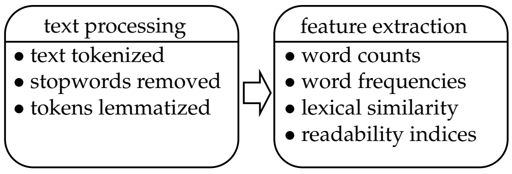


Figure 1. A scheme of text processing procedures and extraction of item text features.

Finally, item text features are derived [23]. We consider four types of item text features. Firstly, the *word counts* feature is easily calculated using lengths of vectors of item text tokens. Secondly, using The Corpus of Contemporary American English (COCA) [24,25], the *word frequencies* are assessed compared to usual frequencies of given words in ordinary language. Then, the *lexical similarity* is calculated using Euclidean and cosine metrics to describe how textually similar (or close) are vectors of tokens of item wording's different parts, e.g., how similar the item question and its key option, i.e., the correct answer, are, considering that their high lexical similarity may tend to make the item easier. Additionally, the lexical similarity between the key option and the distractors, i.e., incorrect answers, is calculated, considering that large dissimilarity can make the item easier. Lastly, we compute the *readability indices* depicting how easy-to-read and easy-to-understand the wording of the text is. In general, the readability indices usually follow formulae of the form

$$\text{readability index} = f\left(\boldsymbol{v}^T_{\text{word counts}}, \boldsymbol{v}^T_{\text{word frequencies}}, \boldsymbol{v}^T_{\text{word counts} \otimes \text{word frequencies}}\right),$$

where f is a function in an explicit form using a vector of absolute and relative counts of words and parts of speech of a given text $\boldsymbol{v}^T_{\text{word counts}}$, a vector of common or unique word's frequencies compared to everyday language, $\boldsymbol{v}^T_{\text{word frequencies}}$, and various combinations of previous two properties, $\boldsymbol{v}^T_{\text{word counts} \otimes \text{word frequencies}}$, as suggested by [26]. A more detailed explanation of individual item features derived using the above-described approaches is in Appendix A.

Eventually, using the above techniques, we derive more than 60 text features per item and list them into a structured dataset of size $n \times k$ so that each column represents one feature for each of n items, and each row contains a vector of all k features for a given item.

2.2. Item Difficulty Based on Student Responses

Having data from more than 50 thousand test-takers answering the items each year, we enrich the dataset of item text features constructed in the previous step by the item difficulty estimated using Rasch model [1] (p. 158), [27,28] from student responses. The Rasch model is relatively simple but can estimate item difficulty for each item; more complex models can describe other item parameters, such as item discrimination or item guessing, that are not of interest in this study. The Rasch models assumes that a test-taker with ability θ_p answers item i correctly with a probability

$$P(\text{test-taker with ability } \theta_p \text{ answers item } i \text{ correctly}) = \frac{e^{\theta_p - y_i}}{1 + e^{\theta_p - y_i}}, \tag{1}$$

where y_i is the difficulty Y of item i, that is of main interest in this study (thus the notation).

We use the conditional maximum likelihood method [1] (p. 165) to estimate difficulties for each item $i \in \{1, 2, \ldots, n\}$ based on the Rasch model (1). The conditional likelihood method accounts for the overall ability of the tested sample, which may differ each year; the item difficulty's estimate is proportional to a portion of incorrect answers to the item adjusted by a proportion of the total number of correct answers. As an output, we obtain a vector $(y_1, y_2, \ldots, y_n)^T$ of n values of item difficulty Y for each item. Note that estimates of item difficulty based on student responses are close to the true item difficulties when a representative and a sufficiently large sample of test-takers are available. This was the case in our study; however, such a respondent sample may not be available in all situations.

2.3. Machine Learning Algorithms

In this study, we compare the performance of several ML methods for predicting and classifying item difficulty. Let us define the regression and classification tasks more formally before describing the supervised regression and classification algorithms.

Assume we initially have $k \in \mathbb{N}$ item text features $X_1, X_2, \dots, X_k$ and $(k+1)$-th variable Y, a dependent one, i.e., item difficulty, derived as indicated in the previous section. As an output of the Rasch model from Formula (1), the item difficulty Y is estimated as a continuous variable. The *regression task* algorithms predict a value y_i of the item difficulty Y for item i using values $x_{i,1}, x_{i,2}, \dots, x_{i,k}$ of all item text features $X_1, X_2, \dots, X_k$ as predictors.

However, for test construction, predicting an exact point value from an item difficulty continuum is unnecessary; test developers often rely on the item difficulty category, thus classifying item difficulty into only a few, e.g., five categories, is sufficient. Thus, we also implement the *classification task*. As the first step, the item difficulty Y is categorized, obtaining Y_c, so that it is split into $m \in \mathbb{N}$ disjunctive intervals $\{c_1, c_2, \dots, c_m\}$ of the same size using appropriate quantiles. Thus, union $\bigcup_{\ell=1}^{m} c_\ell$ is a range of item difficulty variable Y, i.e.,

$$\bigcup_{\ell=1}^{m} c_\ell = \{\forall y \in \mathbb{R} : Y_{\min} \leq y \leq Y_{\max}\},$$

and intersection $\bigcap_{\ell=1}^{m} c_\ell$ is an empty set,

$$\bigcap_{\ell=1}^{m} c_\ell = \emptyset.$$

Then, within the classification task, item feature X_j, where $j \in \{1, 2, \dots, k\}$, is treated as an independent variable for a classification model, which predicts the most-likely interval $c_\ell^* \in Y_c$ of the categorized item difficulty Y_c.

A flowchart of the regression task is in Figure 2; similarly, a classification task scheme is in Figure 3. Regardless of the regression or classification task, the predicted item difficulty values are compared to the 'true' ones as estimated using the Rasch model. To increase reproducibility as much as possible, algorithms are learned on training subsets, while point estimates of the predictive metrics are estimated on testing subsets. This is repeated several times to obtain a more robust estimate of the predictive performance, averaging all point estimates collected from individual iterations.

Domain experts estimate the item difficulty Y using their empirical knowledge in the field, and their item difficulty estimates Y might also be categorized to create Y_c. Thus, domain experts can be treated as "another" regression and "another" classification algorithm and their performance can be compared to the predictive and classification performance of ML algorithms. Many ML algorithms have both the regression and classification version [29], as we describe in more detail in the next section.

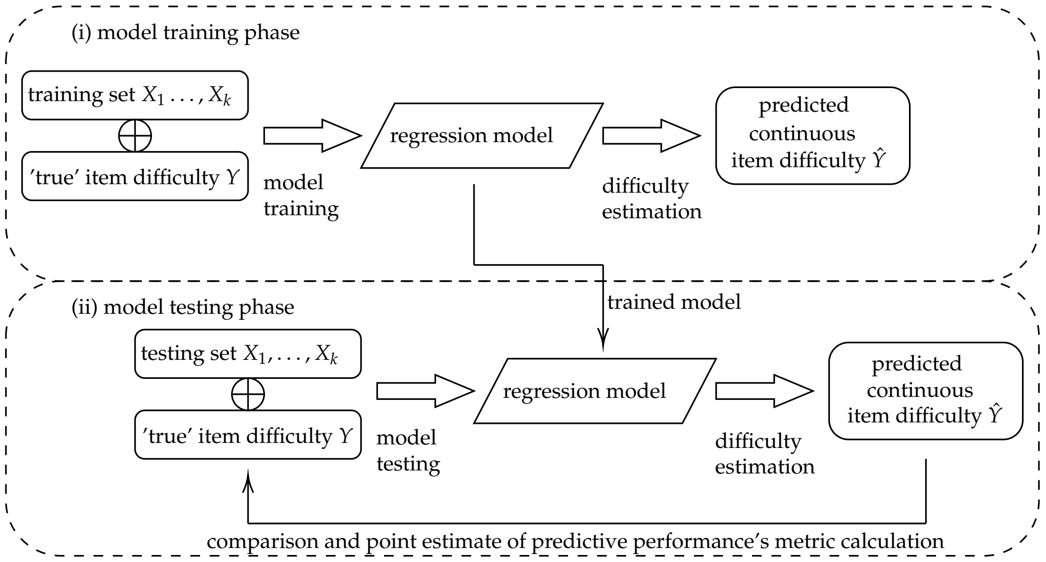
2.3.1. Regularization

Although regularization techniques could serve as regression algorithms, they also offer an option to select a subset of item features used for model building. Therefore, regularization methods enable *feature selection*, which helps reduce the problem's dimensionality with minimal loss of information.

LASSO (Least Absolute Shrinkage and Selection Operator) regression estimates a value y_i of item i's difficulty Y using least squares and L1 regularization-based coefficients $\beta_0, \beta_1, \dots, \beta_k$ minimizing the following term,

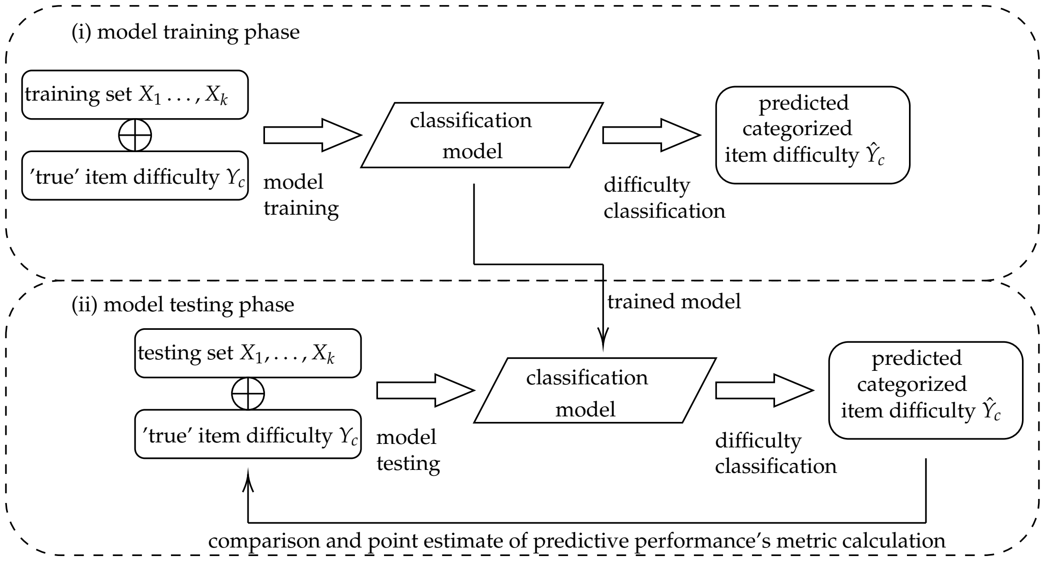
$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{i,j} \right)^2 + \lambda_{\text{LASSO}} \cdot \sum_{j=1}^{k} |\beta_j|, \tag{2}$$

where $x_{i,j}$ is a value of j-th feature of i-th item with $j \in \{1, 2, \ldots, k\}$ and $\lambda_{\text{LASSO}} > 0$ is a penalization term [30].



(i) model training phase
training set $X_1 \ldots, X_k$
'true' item difficulty Y
model training
regression model
difficulty estimation
predicted continuous item difficulty $\hat{Y}$
trained model
(ii) model testing phase
testing set $X_1, \ldots, X_k$
'true' item difficulty Y
model testing
regression model
difficulty estimation
predicted continuous item difficulty $\hat{Y}$
comparison and point estimate of predictive performance's metric calculation

Figure 2. A flowchart of the regression task. The model is built using a training set, while item difficulty Y is predicted using a testing set. The training and testing phase are repeated several times within the cross-validation to increase the robustness and reproducibility of the predictive performance metric estimate. The 'true' item difficulty Y is in quotes since it is estimated from student response data rather than simulated.



(i) model training phase
training set $X_1 \ldots, X_k$
'true' item difficulty Y_c
model training
classification model
difficulty classification
predicted categorized item difficulty $\hat{Y}_c$
trained model
(ii) model testing phase
testing set $X_1, \ldots, X_k$
'true' item difficulty Y_c
model testing
classification model
difficulty classification
predicted categorized item difficulty $\hat{Y}_c$
comparison and point estimate of predictive performance's metric calculation

Figure 3. A flowchart of the classification task. The model is built using a training set, while item difficulty Y_c is classified using a testing set. The training and testing phase are repeated several times within the cross-validation to increase the robustness and reproducibility of the predictive performance metric estimate. The 'true' item difficulty category Y_c is in quotes since it is estimated from student response data rather than simulated.

Similarly, *ridge regression* uses L2 penalization and penalization term $\lambda_{\text{ridge}} > 0$ to minimize

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{i,j} \right)^2 + \lambda_{\text{ridge}} \cdot \sum_{j=1}^{k} \beta_j^2, \tag{3}$$

while item difficulty Y's value y_i is estimated for item i using its item text features $x_{i,j}$ with $j \in \{1, 2, \ldots, k\}$ [31].

Finally, *elastic net* regression combines both L1 and L2 penalization and minimizes the following function,

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{i,j} \right)^2 + \lambda_{\mathrm{LASSO}} \cdot \sum_{j=1}^{k} |\beta_j| + \lambda_{\mathrm{ridge}} \cdot \sum_{j=1}^{k} \beta_j^2 \tag{4}$$

to estimate item i's difficulty Y using its text features $x_{i,j}$, where $j \in \{1, 2, \ldots, k\}$. Assuming both penalizations, i.e., the L1 and L2 terms in Formula (4) are convex [32], elastic net usually reaches values of the function in Formula (4) as minimal as LASSO or ridge regression individually does, and, thus, performs at least as good as the previous two regularization algorithms [33].

Since Formulae (2)–(4) are minimized while coefficients $\beta_0, \beta_1, \ldots, \beta_k$ are estimated, the terms $\lambda_{\mathrm{LASSO}} \sum_{j=1}^{k} |\beta_j|$, $\lambda_{\mathrm{ridge}} \sum_{j=0}^{k} \beta_j^2$ are also minimized. Thus, if $\lambda_{\mathrm{LASSO}} = 0$ or $\lambda_{\mathrm{ridge}} = 0$, penalization terms in Formulae (2)–(4) are removed, and the functions in the formulae become ordinary least squares usual for multivariate linear regression. Otherwise, whenever is $\lambda_{\mathrm{LASSO}} > 0$ and $\lambda_{\mathrm{ridge}} > 0$, then, for β_j close to zero, such a coefficient tends to be shrunk towards zero, and, consequently, j-th item feature is removed from the model while $\hat{\beta}_j = 0$. Thus, regularization techniques could also work as feature selectors. LASSO is considered a better feature selector than ridge regression [34]. Intuitively, assuming j-th item feature X_j is likely to be removed from the model, so it is $0 < |\beta_j| < 1$, then also $0 \cdot |\beta_j| < |\beta_j| \cdot |\beta_j| < 1 \cdot |\beta_j|$ and, consequently, $\beta_j^2 < |\beta_j|$ $(\dagger)$. Whenever is $\lambda_{\mathrm{ridge}} \cdot \beta_j$ or $\lambda_{\mathrm{ridge}} \cdot \beta_j^2$ term large enough so that removing the j-th item feature X_j from the model would reduce the penalization term significantly, the j-th item feature is removed. Thus, for constant values of $\lambda_{\mathrm{LASSO}} = \lambda_{\mathrm{ridge}}$, due to $(\dagger)$, term $\lambda_{\mathrm{ridge}} \cdot \beta_j^2$ in the ridge regression is not as large as term $\lambda_{\mathrm{LASSO}} \cdot |\beta_j|$ in the LASSO, and, consequently, it is less likely that j-th item feature X_j is removed from the ridge regression model than from LASSO model, keeping the penalization levels the same for the two models.

2.3.2. Naïve Bayes Classifier

Naïve Bayes classifier classifies i-th item into the most likely class c_ℓ^* of item difficulty Y. The Bayes theorem assumes that a relationship between conditional probabilities $P(Y_i = c_l \mid \forall x_{i,j})$ and $P(\forall x_{i,j} \mid Y_i = c_\ell)$, where $\forall x_{i,j}$ term stands for a joint proposition $\{X_{i,1} = x_{i,1} \land X_{i,2} = x_{i,2} \land \cdots \land X_{i,k} = x_{i,k}\}$, is

$$P(Y_i = c_\ell \mid \forall x_{i,j}) = \frac{P(\forall x_{i,j} \mid Y_i = c_\ell) P(Y_i = c_\ell)}{P(\forall x_{i,j})}. \tag{5}$$

The non-conditional probabilities $P(Y_i = c_\ell)$ and $P(\forall x_{i,j})$ are constant for a given dataset [35] and can be easily estimated as

$$\hat{P}(Y_i = c_\ell) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{I}(Y_i = c_\ell) \qquad \text{and} \qquad \hat{P}(\forall x_{i,j}) = \frac{1}{n \cdot k} \sum_{i=1}^{n} \sum_{j=1}^{k} \mathcal{I}(X_{i,j} = x_{i,j}), \tag{6}$$

where $\mathcal{I}(\mathcal{A})$ is an identifier function which is equal to 1 if and only if proposition A is true, otherwise it is equal to 0, i.e.,

$$\mathcal{I}(\mathcal{A}) = \begin{cases} 1, & \text{proposition } \mathcal{A} \text{ is true,} \\ 0, & \text{proposition } \mathcal{A} \text{ is false.} \end{cases} \tag{7}$$

Thus, proportion $\frac{P(Y_i = c_\ell)}{P(\forall x_{i,j})}$ is constant and Formula (5) can be rewritten as

$$P(Y_i = c_\ell \mid \forall x_{i,j}) \propto P(\forall x_{i,j}) \mid Y_i = c_\ell),$$

and as far as we assume classes $c_1, c_2, \dots c_m$ are independent, we may also write

$$\begin{aligned} P(Y_i = c_\ell \mid \forall x_{i,j}) \propto P(\forall x_{i,j} \mid Y_i = c_\ell) \propto \\ \propto P(X_{i,1} = x_{i,1} \wedge X_{i,2} = x_{i,2} \wedge \cdots \wedge X_{i,k} = x_{i,k} \mid Y_i = c_\ell) \propto \\ \propto \prod_{j=1}^{k} P(X_{i,j} = x_{i,j} \mid Y_i = c_\ell). \end{aligned}$$

With Naïve Bayes, item i is classified into interval c_ℓ^* so that

$$c_\ell^* = \underset{\ell \in \{1,2,\dots,m\}}{\operatorname{argmax}} \left\{ P(Y_i = c_\ell \mid \forall x_{i,j}) \right\} = \underset{\ell \in \{1,2,\dots,m\}}{\operatorname{argmax}} \left\{ \prod_{j=1}^{k} P(X_{i,j} = x_{i,j} \mid Y_i = c_\ell) \right\}.$$

For categorical item features X_j, probability $P(X_{i,j} = x_{i,j} \mid Y_i = c_\ell)$ is estimated similarly to Formula (6); for continuous variables X_j, it is estimated using cumulative version of normal distribution function, i.e., $\hat{P}(X_{i,j} = x_{i,j} \mid Y_i = c_\ell) = \Phi(x_{i,j} \pm \epsilon \mid Y_i = c_\ell)$ for small positive $\epsilon > 0$.

2.3.3. Support Vector Machines

Assuming the space of all item features $X_1 \times X_2 \times \cdots X_k$, *support vector machines* use a hyperplane to split the space into two disjunctive subspaces (of different classes). The splitting maximizes the margins, i.e., the distance between the two closest points, so that the first comes from one subspace (of the first class) while the latter comes from the second subspace (of the latter class). The hyperplane is orthogonal to the distance of the two closest points, assuming each subspace contains ideally observations of only one class; see Figure 4 for details. Assuming m classes, since one model of support vector machines can classify into only two classes, $\binom{m}{2}$ models in total are built [36].

Each model of support vector machines searches for a splitting hyperplane that follows a form of

$$\boldsymbol{w}^T \boldsymbol{x}_i - b = 0,$$

where $\boldsymbol{w}$ is a vector orthogonal to the splitting hyperplane, and b is maximally tolerated margin's width. Additionally, the two closest points from both subspaces are elements of mutually parallel hyperplanes (and also parallel to the splitting hyperplane), i.e., $\boldsymbol{w}^T \boldsymbol{x}_i - b > 0$ and $\boldsymbol{w}^T \boldsymbol{x}_i - b < 0$, respectively. Finally, the distance between the two closest points of different classes is $\left\{ \frac{2b}{\|\boldsymbol{w}\|} \right\}$, i.e., a width of both margins, and it should be maximized with respect to the existence of two distinguishable hyperplanes for two closest points of different classes, so that

$$\max \left\{ \frac{2b}{\|\boldsymbol{w}\|} \right\} \qquad \text{subject to} \qquad |\boldsymbol{w}^T \boldsymbol{x}_i - b| > 0,$$

where b as the tolerated margin width, i.e., a user's tuning parameter, is usually chosen as $b \geq 1$.

A kernel trick with various kernel functions is applied when the points that belong to different classes are not linearly separable. In principle, the universe of item features, $X_1 \times X_2 \times \cdots \times X_k$, is extended by new variables $U_1, U_2, \dots$, that increase the universe dimensionality [37] and, eventually, after that, it becomes linearly separable as indicated in Figure 5.

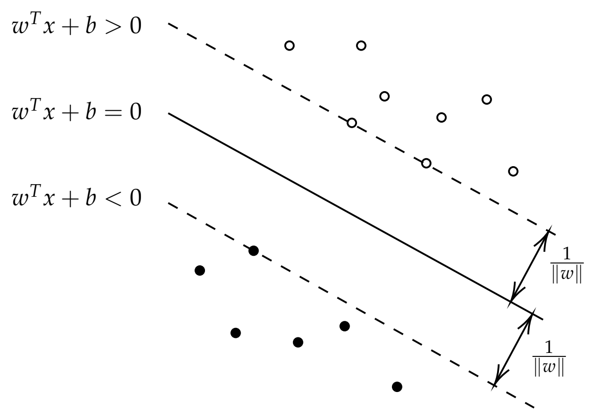


Figure 4. The margin between the hyperplane of the support vector machines (solid line) and the closest points of both subspaces (dashed lines) is maximized by the algorithm.

The classification of item i into difficulty Y's class c^*_ℓ is then performed using a voting scheme, i.e., the class c^*_ℓ is the one that the majority of all $\binom{m}{2}$ models votes for, i.e.,

$$c^*_\ell = \underset{\ell \in \{1,2,\dots,m\}}{\operatorname{argmax}} \left\{ \sum_{\mu=1}^{\binom{m}{2}} \mathcal{I}(\mu\text{-th model votes for class } \ell) \right\},$$

using the same mathematical notation and identifier function as defined in Formula (7).

When regression is applied, trivial (usually constant) models are built for each subspace of the space, divided by the splitting hyperplane. Therefore, averages of all coordinates of all observations belonging to a given subspace are calculated, representing the regression model of the given subspace.

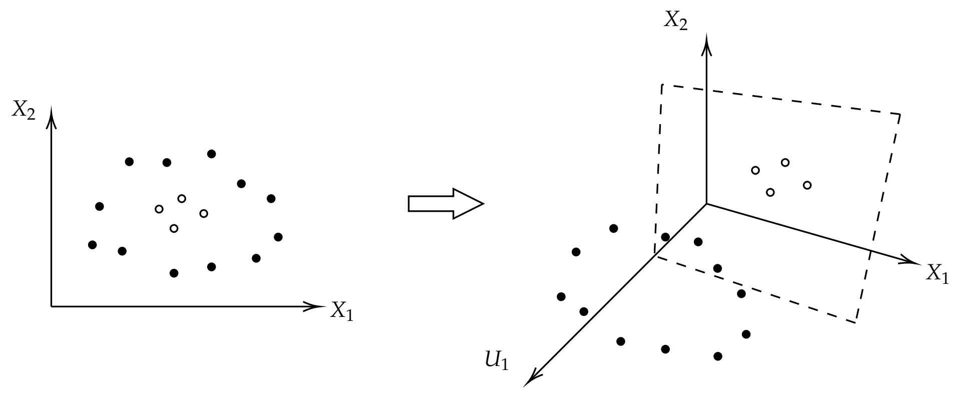


Figure 5. A visualization of the kernel trick's principle.

2.3.4. Regression and Classification Trees and Random Forests

Classification trees, also called *decision trees*, partition the dataset into subdatasets to contain ideally observations of only one class of item difficulty Y. The partitioning is performed successively from the original dataset by binary splitting; a given criterion is minimized within each dataset splitting. In other words, item features' universe $X_1 \times X_2 \times \cdots \times X_k$ is split into disjunctive orthogonal subspaces, including, if not all, then the vast majority of all points from one class of item difficulty Y. Each step of the dataset partitioning, i.e., splitting a parenting dataset into two new child subdatasets, enables the growth of a typical tree plot, dendrogram, by adding two new child branches; see Figure 6. The partitioning is applied multiple times until the dataset is split according to item difficulty Y classes' distribution [38].

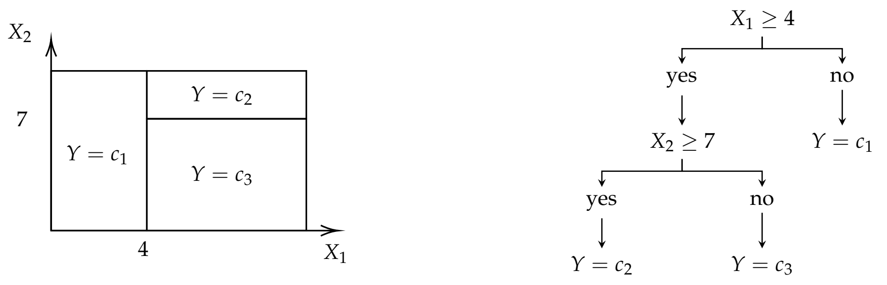


Figure 6. Linear splitting of the variables' space (on the **left**) and an appropriate tree representation (on the **right**).

Assuming $\rho_{\eta,\ell}$ is a proportion of observations of class c_ℓ in part of the dataset that is defined by all node rules from root to node η, then $\rho_{\eta,\ell}$ should be maximized as much as possible using an impurity criterion $Q(\eta)$. The most often used impurity measures are the misclassification error,

$$Q(\eta) = 1 - \rho_{\eta,\ell}, \tag{8}$$

the Gini index,

$$Q(\eta) = \sum_{\ell=1}^{m} \rho_{\eta,\ell}(1 - \rho_{\eta,\ell}), \tag{9}$$

and the deviance, also called cross-entropy,

$$Q(\eta) = -\sum_{\ell=1}^{m} \rho_{\eta,\ell} \cdot \log \rho_{\eta,\ell}. \tag{10}$$

Obviously, the impurity measure $Q(\eta)$ is minimized in each dataset's partitioning since the lower the impurity measure is, the larger the proportion $\rho_{\eta,\ell}$ is. Trees tend to overfit the distribution of classes in the dataset; it means the tree growth is stopped no sooner than all leave nodes have the impurity criterion as minimized as possible. To avoid overfitting, various stopping criteria or pruning are applied [39].

Once the tree is grown, it enables to classify item i into difficulty Y's class c^*_ℓ, so that

$$c^*_\ell = \underset{\ell \in \{1,2,\ldots,m\}}{\operatorname{argmax}} \left\{ \rho_{\text{leaf node determined by all node rules from root to the node, } \ell} \right\},$$

using the introduced notation and identifier function from Formula (7). Trivial (constant) models constructed for each subspace transform the classification trees into *regression trees* [40].

Multiple trees create a structure called *random forest*. Individual trees of a given random forest are mutually independent and different. This is ensured by applying only a subset of all item features pre-selected using a bootstrap for each new tree growing in a random forest. Finally, the classification or regression output of the random forest is determined by the voting scheme of individual trees [41], similarly as for the support vector machines: item i is classified into such a difficulty Y's class c^*_ℓ for which the majority of all trees in the random forest votes, i.e.,

$$c^*_\ell = \underset{\ell \in \{1,2,\ldots,m\}}{\operatorname{argmax}} \left\{ \sum_{\tau \in \{\text{trees of random forest}\}} \mathcal{I}(\text{tree } \tau \text{ votes for class } \ell) \right\},$$

using the same mathematical notation as above.

2.3.5. Neural Networks

Neural networks are universal algorithms suitable for regression and classification tasks. An architecture of a neural network consists of a layer of input and output neuron(s) and several hidden layers so that each hidden layer consists of multiple neurons [42].

An example of the neuron is in Figure 7. On input of the neuron, there is a vector of signals from neurons of a previous layer, i.e., $\boldsymbol{z}_{l-1} = (z_{l-1,1}, z_{l-1,2}, \ldots)^T$, multiplied by a vector of weights $\boldsymbol{w}_{l-1} = (w_{l-1,1}, w_{l-1,2}, \ldots)^T$. If $l = 1$, the neurons of the first layer accept weighted signals from a vector of item i's features, $\boldsymbol{x}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,k})^T$. Weighted signals from $(l-1)$-th layer are summed up together with bias term b_l within Σ function, i.e.,

$$\Sigma = \boldsymbol{w}_{l-1} \cdot \boldsymbol{z}_{l-1} + b_l,$$

and proceeded to the σ function, which is an activating function, usually of the sigmoid form,

$$\sigma(\zeta) = \frac{1}{1+e^{-\zeta}},$$

so that signal $y_{l,1}$ on output from the neuron of l-th layer is

$$y_{l,1} = \sigma(\Sigma + b_l) = \sigma(\boldsymbol{w}_{l-1} \cdot \boldsymbol{z}_{l-1} + b_l) = \frac{1}{1+e^{-(\boldsymbol{w}_{l-1} \cdot \boldsymbol{z}_{l-1} + b_l)}},$$

which is finally transcended to the next, $(l+1)$-th layer. Vectors of weights, $\boldsymbol{w}_l = (w_{l,1}, w_{l,2}, \ldots)^T$ are adjusted within each iteration of so-called *backpropagation* when the weights are increased or decreased by small gradients to minimize the loss function, often implemented as L1 or L2 penalization [43].

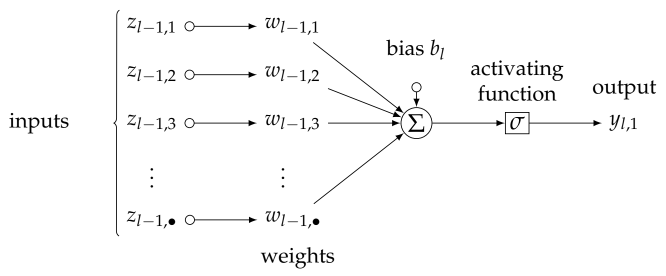


Figure 7. A scheme of one neuron in neural network.

In the regression framework, besides neurons in a hidden layer, we implement a single neuron in the output layer, returning continuous estimate $\hat{y}_i$ of item i's difficulty. In the classification framework, there are m output neurons representing classes $\{c_1, c_2, \ldots, c_m\}$ and we adopt voting for c_l^* in classifying network [44], as follows

$$c_\ell^* = \underset{\ell \in \{1,2,\ldots,m\}}{\operatorname{argmax}} \left\{ y_{\#\text{ of layers},\, \ell} \right\}.$$

2.3.6. Variable Importance Analysis

While the importance analysis is not a stand-alone algorithm for item difficulty (or its categorized variant) prediction, it enables us to evaluate how "important" a given variable is for a model, considering the predictive performance; in other words, how much poorer the model would predict if it lacked the given variable [45].

We apply two measures of variable importance; each variable, i.e., item feature, has its own value of the importance measure, considering a given dataset and model. Before we introduce the measures, we define the mean square error, MSE, as

$$\text{MSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2, \tag{11}$$

for vectors $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^T$ and $\hat{\boldsymbol{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)^T$ of observed and predicted difficulties of n items, respectively. The first importance measure is $\text{MSE}_{\text{increase}}(X_j)$, which is equal to an increase of mean square error of item difficulty prediction in such a model where values of the given item feature, X_j, are randomly permuted [45]. To be more specific, we firstly calculate mean square error $\text{MSE}_{\{-\emptyset\}}$ of a full model with all original item features, then we compute mean square error $\text{MSE}_{\{-j\}}$ of a model where item feature X_j has randomly shuffled values. Finally, $\text{MSE}_{\text{increase}}(X_j)$ is defined as

$$\text{MSE}_{\text{increase}}(X_j) = \frac{\text{MSE}_{\{-j\}} - \text{MSE}_{\{-\emptyset\}}}{\text{MSE}_{\{-\emptyset\}}}. \tag{12}$$

The more important item feature X_j for adequate and accurate prediction of item difficulty, the larger the prediction error, measured using mean square error MSE, when the item feature X_j is missing in the model. Thus, the greater the value of $\text{MSE}_{\text{increase}}(X_j)$, the more important the item feature X_j for item difficulty prediction.

The second importance measure, node purity increase, $\text{NodePurity}_{\text{increase}}(X_j)$ is defined similarly. Once impurity metric $Q(\eta)$ is chosen, i.e., either misclassification error (8), Gini index (9) or deviance (10), the node purity increase, $\text{NodePurity}_{\text{increase}}(X_j)$, for item feature X_j is simply an increase of "1 minus impurity metric" term averaged over all leaf nodes if the item feature X_j is newly introduced into a new model [45]. Thus, having the averaged "$1-$ node impurity" term, $\overline{(1-Q(\eta))}_{\{-j\}}$, of a tree model with all original item features except for item feature X_j, and averaged "$1-$ node impurity", $\overline{(1-Q(\eta))}_{\{-\emptyset\}}$, of a model where item feature X_j is already included, the $\text{NodePurity}_{\text{increase}}(X_j)$ is then

$$\text{NodePurity}_{\text{increase}}(X_j) = \frac{\overline{(1-Q(\eta))}_{\{-\emptyset\}} - \overline{(1-Q(\eta))}_{\{-j\}}}{\overline{(1-Q(\eta))}_{\{-j\}}}. \tag{13}$$

Again, the more important item feature X_j for the predictive model performance, the higher average "$1-$ node impurity" increase, i.e., the higher average purity increase we can expect once the item feature X_j is introduced into the model. Thus, the larger the value of $\text{NodePurity}_{\text{increase}}(X_j)$, the more important the item feature X_j for item difficulty prediction. According to some sources, e.g., [46], $\text{MSE}_{\text{increase}}(X_j)$ measure should be preferred to $\text{NodePurity}_{\text{increase}}(X_j)$, since the latter one is biased.

2.4. Evaluation of Algorithm Performance

Regression and classification tasks are evaluated using mutually different performance metrics. To obtain more robust estimates of the performance metrics, both regression and classification models are trained multiple times using various training sets, which enables us to average the metrics using all point estimates, collected one per each cross-validation iteration [47]; see Figures 2 and 3. We also compare an item difficulty prediction performance of the ML approaches with the performance of domain experts.

2.4.1. Evaluation of Regression Performance

The models within the regression task are evaluated and compared using root mean square error (RMSE), i.e.,

$$\text{RMSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{14}$$

for vectors $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^T$ and $\hat{\boldsymbol{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)^T$ of observed and predicted difficulties of n items, respectively. Obviously, inspecting Formulae (11) and (14), we obtain the following identity, $\text{MSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \text{RMSE}(\boldsymbol{y}, \hat{\boldsymbol{y}})^2$. Since RMSE indicates the significance of error between observed and predicted item difficulties, the lower RMSE indicates the better predictive performance of a given regression algorithm.

2.4.2. Evaluation of Classification Performance

Assuming there are m observed classes that are predicted using a classifier, we could calculate a number of cases $n_{u,v}$ when 'true' class c_u is predicted as class c_v, where $u \in \{1, 2, \dots, m\}$ and $v \in \{1, 2, \dots, m\}$. Listing these frequencies in a table, we obtain Table 1, called the *confusion matrix*.

Table 1. A confusion matrix for m observed, 'true' classes $Y_c \in \{c_1, c_2, \dots, c_m\}$ (in rows) and m predicted classes $\hat{Y}_c \in \{c_1, c_2, \dots, c_m\}$ (in columns).

		Predicted Class ($\hat{Y}_c$)			
		c_1	c_2	$\cdots$	c_m
'true' class (Y_c)	c_1	$n_{1,1}$	$n_{1,2}$	$\cdots$	$n_{1,m}$
	c_2	$n_{2,1}$	$n_{2,2}$	$\cdots$	$n_{2,m}$
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	c_m	$n_{m,1}$	$n_{m,2}$	$\cdots$	$n_{m,m}$

The better and more accurate the classification is, the higher frequencies $n_{i,i}$ are aligned across the confusion matrix's principal diagonal. Thus, marking the confusion matrix as $\boldsymbol{C}$ and assuming vectors $\boldsymbol{y}_c$ of observed item difficulty classes and $\hat{\boldsymbol{y}}_c$ of predicted difficulty classes, we define *predictive accuracy* as the ratio of correctly classified items,

$$\begin{aligned}\text{predictive accuracy}(\boldsymbol{y}_c, \hat{\boldsymbol{y}}_c) &= \frac{1}{n}\sum_{i=1}^{n}\{\mathcal{I}(\hat{y}_{c,i} = c_\ell \wedge y_{c,i} = c_\ell)\} = \\ &= \frac{\operatorname{tr}\boldsymbol{C}}{\sum \boldsymbol{C}} = \frac{\sum_{u=1}^{m} n_{u,u}}{\sum_{u=1}^{m}\sum_{v=1}^{m} n_{u,v}}. \end{aligned} \tag{15}$$

The higher the predictive accuracy, the better and more accurate the classification is [48]. Each of m classes of item difficulty Y_c are of equal size in the dataset (classes are split using quantiles) (†). Assuming a classifier would predict difficulties $\boldsymbol{y}_c$ as vector $\hat{\boldsymbol{y}}_{c,r}$ as a random guessing algorithm, then an expected value of its predictive accuracy is

$$\begin{aligned}\mathbb{E}(\text{predictive accuracy}(\boldsymbol{y}_c, \hat{\boldsymbol{y}}_{c,r})) &= \sum_{\ell=1}^{m} P(\hat{Y}_c = c_\ell \mid Y_c = c_\ell) \cdot P(Y_c = c_\ell) \overset{(\dagger)}{=} \\ &\overset{(\dagger)}{=} \sum_{\ell=1}^{m}\frac{1}{m}\cdot\frac{1}{m} = \sum_{\ell=1}^{m}\frac{1}{m^2} = \frac{m}{m^2} = \frac{1}{m}.\end{aligned}$$

Values of predictive accuracy greater than $\frac{1}{m}$ indicate that a classifier performs better than a random guessing algorithm.

In practice, the very accurate prediction of a correct difficulty class is unnecessary. A prediction *close* enough to the correct difficulty class, i.e., the correct one or one class below or above the correct one, is still useful. Thus, we also measure the classifiers' performance using an *extended predictive accuracy*. The item i is evaluated as correctly classified if it is classified in the correct difficulty class $\hat{y}_{c,i} = y_{c,i} = c_\ell$, or one class higher if such a class exists, $\hat{y}_{c,i} = c_{\ell+1}$, or one class lower if such a class exists, $\hat{y}_{c,i} = c_{\ell-1}$, compared to the difficulty class estimated from student response data, thus

$$\text{extended predictive accuracy}(\boldsymbol{y}_c, \hat{\boldsymbol{y}}_c) = \frac{1}{n}\sum_{i=1}^{n}\{\mathcal{I}(\hat{y}_{c,i} \in \{c_{\ell-1}, c_\ell, c_{\ell+1}\} \wedge y_{c,i} = c_\ell)\}, \tag{16}$$

where $c_{\ell-1}$ is one class below c_ℓ, and $c_{\ell+1}$ is one class above c_ℓ, respectively, if it exists, and an empty set otherwise. Thus, a probability that a classifier in this sense correctly classifies category with subscript $\ell \in \{2, 3, \dots, m-1\}$ is equal to $\frac{|\{\ell-1,\ell,\ell+1\}|}{m} = \frac{3}{m}$, while a probability that a classifier correctly classifies the first or last category with subscript

$\ell = 1$ or $\ell = m$ is equal to $\frac{|\{\ell,\ell+1\}|}{m} = \frac{2}{m}$ or $\frac{|\{\ell-1,\ell\}|}{m} = \frac{2}{m}$, respectively. Again, assuming a classifier would predict difficulties $\boldsymbol{y}_c$ as vector $\hat{\boldsymbol{y}}_{c,r}$ as a random guessing algorithm, then an expected value of its extended predictive accuracy is

$$\begin{aligned}\mathbb{E}(\text{extended predictive accuracy}(\boldsymbol{y}_c, \hat{\boldsymbol{y}}_{c,r})) = \\ &= \sum_{\ell=1}^{m} P(\hat{Y}_c \in \{c_{\ell-1}, c_\ell, c_{\ell+1}\} \mid Y_c = c_\ell) \cdot P(Y_c = c_\ell) \overset{(\dagger)}{=} \\ &\overset{(\dagger)}{=} \left(\frac{2}{m} + \underbrace{\frac{3}{m} + \cdots + \frac{3}{m}}_{(m-2)\text{-times}} + \frac{2}{m} \right) \cdot \frac{1}{m} = \frac{3m-2}{m^2}.\end{aligned}$$

Thus, any values of extended predictive accuracy that are greater than $\frac{3m-2}{m^2}$ show that a classifier predicts better than a random guessing procedure.

2.4.3. Cross-Validation

To obtain more robust estimates, the performance metrics are re-estimated multiple times within f-fold cross-validation, where $f \in \mathbb{N}$ and $f \geq 2$, using dataset splitting into training and testing subset of sizes of $\frac{f-1}{f}$ % and $\frac{1}{f}$ %, respectively, and then averaged [49], see Figure 8.

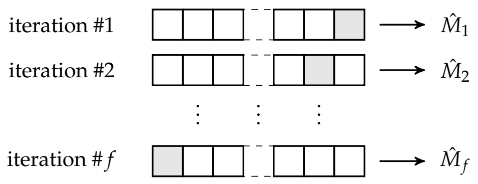


Figure 8. Within p-th iteration of f-fold cross-validation, where $p \in \{1, 2, \ldots, f\}$, $f > 1$ and $f \in \mathbb{N}$, a model is trained using the training set (colored in white) and tested using the test set (colored in grey), i.e., the $(f - p + 1)$-th of f equal-size parts, which the entire dataset was originally split into.

In particular, for even better comparison and integer-like sizes of both the training and testing subsets, it might be optimal to choose f as a divisor of sample size n; then the portions $\frac{f-1}{f}$ % and $\frac{1}{f}$ % for training and testing subsets, respectively, are of integer number sizes.

Assuming that p-th iteration of the f-fold cross-validation outputs point estimates of root mean square error, predictive or extended predictive accuracy $\hat{M}_p$, finally, we could average the estimates as

$$\bar{M} = \frac{1}{f} \sum_{p=1}^{f} \hat{M}_p,$$

to obtain a robust and unbiased estimate of $\hat{\mathbb{E}}(M) = \bar{M}$, i.e., the root mean square error, predictive or extended predictive accuracy [50], respectively.

2.4.4. Relationship between Model's Predictive Performance and a Number of Item Features in a Model

A value of the root mean square error, RMSE, following Formula (14) is *not* closely related to a number of item features considered within a model. Thus, model enrichment by any new extracted text item features could not necessarily improve predictive model performance. There are more details, formal derivation, and mathematical rationale of the relationship between the model predictive performance and the number of item features on model input in Online Supplement listed in Data Availability Statement at the end of the article.

3. Implementation

Text preprocessing and the entire analysis were implemented in statistical language and environment R [51]. For evaluation of the classification task, the continuous difficulty Y, estimated from student response data, of an original range $\langle -2.48, +1.63)$ was split into $m = 5$ disjunctive intervals, denoted $Y_c \in \{c_1, c_2, c_2, c_4, c_5\}$, of the same size using quintiles, specifically $\langle -2.48, -0.80)$, $\langle -0.80, -0.44)$, $\langle -0.44, +0.03)$, $\langle +0.03, +0.52)$, $\langle +0.52, +1.63)$, and labeled as $\{$*very easy*, *easy*, *moderate*, *difficult*, *very difficult*$\}$. Thus, regarding item difficulty, the dataset of item text wordings is well-balanced. While the final number of item features derived from their text wording is $k = 69$, the number of items is $n = 40$. Regarding the f-fold cross-validation, due to a straightforward advantage of whenever n is divisible by $f \geq 2$, we choose for $f = 20$. Thus, since $\frac{n}{f} = \frac{40}{20} = 2$, we applied a *leave-two-out cross-validation*.

Domain experts' evaluation of item difficulty originally uses an arbitrary scale of $\langle 1.0, 2.5 \rangle$. To make the experts' evaluation comparable with the outputs of classifiers, we split the experts' scale in the original logic the scale was designed, i.e., we consider $m = 5$ equidistant intervals over the range of $\langle 1.0, 2.5 \rangle$. Thus, we create $m = 5$ intervals of length 0.3 and name them also as $\{$*very easy*, *easy*, *moderate*, *difficult*, *very difficult*$\}$. Given the assumed Rasch model (1), the obtained 'true' item difficulty is on a logistic scale where very low and very high values are less common, yet possible. For this reason, we split the Rasch-based item difficulty using quantiles. The domain experts, on the other hand, naturally designed the difficulty evaluation scale in a linear fashion, which is our rationale for equidistant scale splitting.

The difficulty of items was estimated from student responses data using the Rasch model by the function `RM()` of `eRm` package [52]. Text preprocessing was performed using R package `quanteda` [53]. Regularization was implemented with a function `glmnet()` of `glmnet` package [54]. Naïve Bayes classifier and support vector machines were built using `naiveBayes()` and `svm()` functions of `e1071` package [55]. The radial kernel function was chosen for the kernel trick if applied. Classification and regression trees were enumerated by the function `rpart()` of `rpart` package [56]. Random forests' models were learned using function `randomForest()` from `randomForest` package [57], each time using 500 trees in a model, similarly as neural networks were modeled using `neuralnet()` function and `neuralnet` package [58]. The neural networks contain one hidden layer with the same number of neurons as item features on input.

4. Results

To assess the possibility and performance of item difficulty prediction from their textual wordings using ML methods, we applied the above-described methodology to the dataset of our interest. Firstly, we built supervised models of the regression task to estimate item difficulty as a continuous variable. There are outcomes of this approach more in detail in Table 2 presented using the root mean square errors (RMSE) for the $n = 40$ single-paragraph items, averaged over all $f = 20$ iterations of the f-fold cross-validation, across seven different regression algorithms and domain experts' estimates, too. The lower value of RMSE an algorithm outputs, the better accuracy and reliability its item difficulty estimate reaches.

A comparison of the algorithms highlights the varying performance levels between the models. Among the evaluated models, the regularization algorithms, i.e., LASSO regression, ridge regression, and elastic net, demonstrated superior performance by yielding the lowest RMSE value, indicating the highest accuracy and reliability. In particular, the elastic net returned the lowest RMSE of 0.666 among the regularization approaches (and, thus, among all models, too). Additionally, considering the data and model settings, the elastic net model outperformed domain experts in the continuous item difficulty prediction since domain experts reached an RMSE of 1.004. On the other hand, the regression trees and neural networks algorithm produced the highest RMSE value of about 0.978 and 0.971, respectively, suggesting less accuracy and reliability than the other models. The remaining algorithms displayed moderate performance levels. Meanwhile, regression trees and domain experts had higher but comparable RMSE values, further emphasizing the superior performance of the elastic net algorithm in this analysis. Since the domain experts evaluate item difficulty mostly using numbers such as $1.0, 1.5, 2.0, 2.5$ as described in Section 3, *Implementation*, they are a priori handicapped to estimate an exact point value of the item difficulty. Applying Sheppard's correction [59], their RMSE as a measure following the logic of the second moment is overestimated by a term of $\frac{\text{width of the interval between valid values}^2}{12} = \frac{0.5^2}{12} \approx 0.02$. However, in case all domain experts would systematically over- or under-estimate the true item difficulty, their RMSE could be, in theory, overestimated by the width of the interval between valid values, thus, by 0.5.

Table 2. Values of root mean square error (RMSE) for seven regression algorithms and domain experts, respectively, estimating item difficulty as a continuous variable, calculated over $f = 20$ iterations of the f-fold cross-validation.

Regression Algorithm	Root Mean Square Error (RMSE)
LASSO regression	0.694
Ridge regression	0.719
Elastic net regression	0.666
Support vector machines	0.716
Regression trees	0.978
Random forests	0.719
Neural networks	0.971
Domain experts	1.004

Additionally, Table 3 presents the predictive and extended predictive accuracies of different classification algorithms, including Naïve Bayes classifier, support vector machines, classification trees, random forests, neural networks, and domain experts.

Table 3. Values of averaged predictive and extended predictive accuracies for five classification algorithms and domain experts, respectively, estimating item difficulty as a categorized variable, calculated over $f = 20$ iterations of the f-fold cross-validation.

Classification Algorithm	Predictive Accuracy	Extended Predictive Accuracy
Naïve Bayes classifier	0.175	0.425
Support vector machines	0.000	0.575
Classification trees	0.150	0.525
Random forests	0.325	0.650
Neural networks	0.225	0.550
Domain experts	0.225	0.650

Assuming that only an approximate match of a true and predicted category of item difficulty is sufficient for applications, we focus on extended predictive accuracy. From the ML algorithms, random forests output the highest extended predictive accuracy with a score of 0.650, while Naïve Bayes classifier showed the lowest extended predictive accuracy, achieving a score of only 0.425. Domain experts achieved a superior accuracy of 0.650, indicating their important role in the classification of item difficulty.

For a better understanding of individual classifiers' predictive capacity, we plot the confusion matrices for each algorithm (see Figure 9), where each row represents numbers of items in each of the observed classes, while each column represents numbers of items in each of the difficulty classes predicted by the algorithm. The numbers in cells of the confusion matrices are summations over all iterations of the f-fold cross-validation. Overall, the results suggest that the ML algorithms could benefit from further improvement to accurately classify items in all classes of difficulty, especially in the middle classes, i.e., from *easy* to *difficult*. The domain experts did not use the highest category, *very difficult* much for these items; this may be caused by the fact that the test is in general easy and especially this type of item may appear simple compared to exercises from high school textbooks.

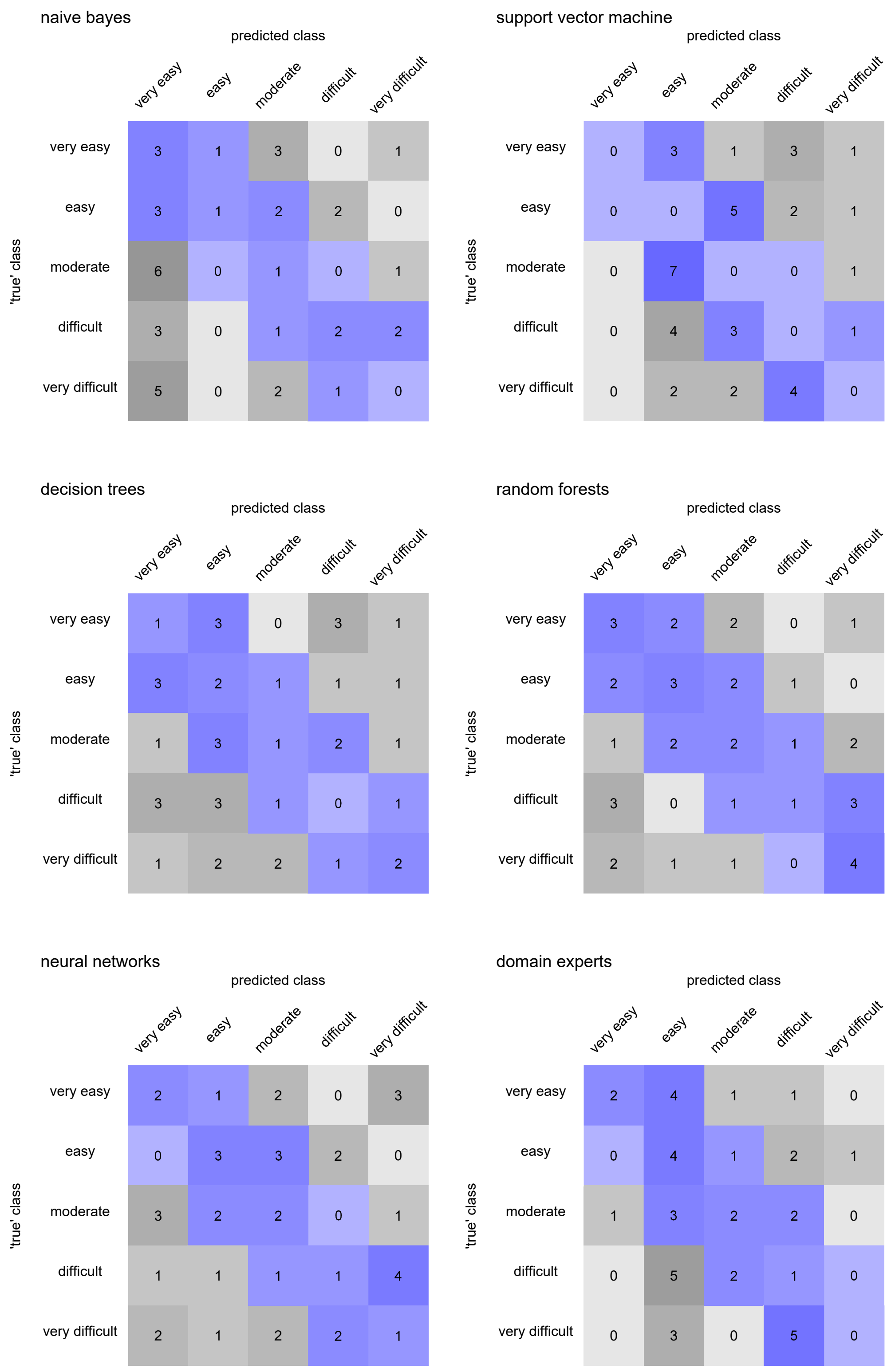
Tables 4 and 5 present the variable importance analysis of different item text features applied in our model for item difficulty prediction and classification. While Table 4 uses $\text{MSE}_{\text{increase}}$ metric, Table 5 utilizes $\text{NodePurity}_{\text{increase}}$ metric of variable importance. Both measures are reported in Tables 4 and 5 as an average $\pm$ standard deviation based on $f = 20$ point estimates from all iterations of f-fold cross-validation. The $\text{MSE}_{\text{increase}}$ as a metric of an item feature's importance operates with mean square error (MSE), which is a squared value of RMSE; it is more suitable for regression models and prediction of item difficulty as a continuous variable. Whereas $\text{NodePurity}_{\text{increase}}$ as a metric of item feature's importance calculates impurity of leaf nodes when classifying into a category of item difficulty; thus, it performs better in the classification of item difficulty. Both measures can provide valuable insights into feature importance; however, they may result in different rankings as they capture distinct aspects of model prediction performance. By considering both metrics, we can comprehensively understand item feature importance and make informed decisions for analysis and interpretation.

According to Table 4, the number of all characters in item wording seems to be the most crucial feature for item difficulty, with $\text{MSE}_{\text{increase}}$ of $5.912 \pm 0.0.673$, followed by the word length's standard deviation (in characters) with $\text{MSE}_{\text{increase}}$ about 4.845 ± 0.799. Various features such as readability indices, indices of similarity or portion of shared words between item passage, distractors, item question or key option, as well as longest and average word length in item wording, follow, with $\text{MSE}_{\text{increase}}$ between about 0.900 and 3.500.

In Table 5, the same two features seem to determine the classification of item difficulty the most—the word length's standard deviation (in characters) with $\text{NodePurity}_{\text{increase}}$ about 1.644 ± 0.121, and the number of all characters in item wording with $\text{NodePurity}_{\text{increase}}$ of 1.455 ± 0.137. Additionally, some of the readability indices, numbers of monosyllabic and rare words, or similarity between different parts of item wording are important for correct item difficulty prediction, returning $\text{NodePurity}_{\text{increase}}$ in an interval of 0.030–0.080.

A detailed explanation of individual item features listed in Tables 4 and 5 is in Appendix A. Note that although we sorted the item features in decreasing order according to the importance measures in Tables 4 and 5, the intervals for importance measures' mean values, indicated by $\pm$ standard deviation terms, overlap between various item features. Thus, the importance analysis is only illustrative.

Table 6 provides a summary of elastic net regression's model following Formula (4) that minimized the root mean square error, RMSE, with $\hat{\lambda}_{\text{LASSO}} \approx 1$ and $\hat{\lambda}_{\text{ridge}} \approx 0$. While most item features were removed by shrinking their coefficients towards zero, the item features listed in Table 6 are those that remained in the model. Compared to the item features' importance analysis, the elastic net model could tell us not only which item features are essential for the final model but also what is the approximate direction of a relationship between the features and item difficulty. The elastic net model suggests that a larger total number of characters in item text wording increases item difficulty ($\hat{\beta} = 0.002 > 0$), and that greater Dalle-Chall and FOG readability indices also make the item more difficult ($\hat{\beta} = 0.004 > 0$ and $\hat{\beta} = 0.026 > 0$, respectively). In addition to this, an increased standard deviation of word lengths within item wording ($\hat{\beta} = 0.809 \gg 0$) and an average sentence length (words) in distractors ($\hat{\beta} = 0.002 \gg 0$) increase item difficulty as well as does the greater proportion of common words in the passage and distractors ($\hat{\beta} = 0.630 \gg 0$) (the passage and distractors–common words_1 is a proportion of a number of common words in the item passage also found in the wording of distractors, to a number of all words in the item passage).



naive bayes

'true' class \ predicted class	very easy	easy	moderate	difficult	very difficult
very easy	3	1	3	0	1
easy	3	1	2	2	0
moderate	6	0	1	0	1
difficult	3	0	1	2	2
very difficult	5	0	2	1	0

support vector machine

'true' class \ predicted class	very easy	easy	moderate	difficult	very difficult
very easy	0	3	1	3	1
easy	0	0	5	2	1
moderate	0	7	0	0	1
difficult	0	4	3	0	1
very difficult	0	2	2	4	0

decision trees

'true' class \ predicted class	very easy	easy	moderate	difficult	very difficult
very easy	1	3	0	3	1
easy	3	2	1	1	1
moderate	1	3	1	2	1
difficult	3	3	1	0	1
very difficult	1	2	2	1	2

random forests

'true' class \ predicted class	very easy	easy	moderate	difficult	very difficult
very easy	3	2	2	0	1
easy	2	3	2	1	0
moderate	1	2	2	1	2
difficult	3	0	1	1	3
very difficult	2	1	1	0	4

neural networks

'true' class \ predicted class	very easy	easy	moderate	difficult	very difficult
very easy	2	1	2	0	3
easy	0	3	3	2	0
moderate	3	2	2	0	1
difficult	1	1	1	1	4
very difficult	2	1	2	2	1

domain experts

'true' class \ predicted class	very easy	easy	moderate	difficult	very difficult
very easy	2	4	1	1	0
easy	0	4	1	2	1
moderate	1	3	2	2	0
difficult	0	5	2	1	0
very difficult	0	3	0	5	0

Figure 9. Summative confusion matrices for five classification algorithms and domain experts, respectively. For each algorithm, within each iteration of the f-fold cross-validation, a partial confusion matrix was calculated from training $\frac{1}{f}$ fraction of the dataset, and the resulting f confusion matrices were combined into one final summative confusion matrix, which is displayed. The blue color indicates cells considered for calculating the extended predictive accuracy.

Table 4. Top twenty item features with the highest value of importance for item difficulty prediction, measured using $\text{MSE}_{\text{increase}}$. The $\text{MSE}_{\text{increase}}$ measure is reported as an average ± standard deviation based on $f = 20$ point estimates from all iterations of f-fold cross-validation. A detailed explanation of individual item features listed in the table is in Appendix A. The abbreviation COCA stands for The Corpus of Contemporary American English, DF matrix for document-feature matrix.

Item Feature	$\textbf{MSE}_{\textbf{increase}}$
Number of characters	5.912 ± 0.673
Word length's standard deviation (characters)	4.845 ± 0.799
Passage and distractors–word2vec similarity	3.521 ± 0.823
Text readability–Traenkle-Bailer index	3.385 ± 0.767
Question and key item–word2vec similarity	2.447 ± 0.956
Distractors–average sentence length (words)	2.385 ± 0.838
Key option and distractors–number of features from a DF matrix	2.225 ± 0.697
Distractors–average word length (characters)	1.689 ± 0.827
Distractors–average word length (characters)	1.689 ± 0.827
Text readability–SMOG index	1.680 ± 0.790
Question and key option–number of features from a DF matrix	1.655 ± 0.807
Item passage and distractors–common words_1	1.570 ± 0.832
Passage and key option–word2vec similarity	1.409 ± 0.979
Text readability–FOG index	1.355 ± 1.143
Question and passage–euclidean distance	1.341 ± 0.784
Average word length (characters)	1.322 ± 0.972
Passage and key option–euclidean distance	1.266 ± 0.997
Passage and distractors–euclidean distance	1.072 ± 1.218
Item passage and distractors–cosine similarity	1.018 ± 0.933
Question and distractors–euclidean distance	0.937 ± 0.927

Table 5. Top twenty item features with the highest value of importance for item difficulty prediction, measured using $\text{NodePurity}_{\text{increase}}$. The $\text{NodePurity}_{\text{increase}}$ measure is reported as an average ± standard deviation based on $f = 20$ point estimates from all iterations of f-fold cross-validation. A detailed explanation of individual item features listed in the table is in Appendix A. The abbreviation CEFR stands for The Common European Framework of Reference for Languages, DF matrix for document-feature matrix.

Item Feature	$\textbf{NodePurity}_{\textbf{increase}}$
Word length's standard deviation (characters)	1.644 ± 0.121
Number of characters	1.455 ± 0.137
Text readability–Traenkle-Bailer index	1.214 ± 0.118
Question and key item–word2vec similarity	0.820 ± 0.103
Passage and distractors–word2vec similarity	0.819 ± 0.097
Passage and distractors–euclidean distance	0.806 ± 0.128
Item passage and distractors–common words_1	0.707 ± 0.099
Distractors–average word length (characters)	0.684 ± 0.153
Distractors–average word length (characters)	0.684 ± 0.153
Question and passage–number of features from a DF matrix	0.674 ± 0.063
Text readability–FOG index	0.631 ± 0.101
Text readability–Dale-Chall index	0.620 ± 0.095
Text readability–SMOG index	0.537 ± 0.067
Distractors–average sentence length (words)	0.514 ± 0.089
Key option and distractors–number of features from a DF matrix	0.508 ± 0.099
Item passage and distractors–cosine similarity	0.499 ± 0.087
Average word length (characters)	0.478 ± 0.068
Question and passage–euclidean distance	0.463 ± 0.083
Average sentence length (words)	0.458 ± 0.062
Passage and key option–euclidean distance	0.431 ± 0.047

Table 6. Coefficients of elastic net regression's model that minimizes RMSE with $\hat{\lambda}_{\text{LASSO}} \approx 1$ and $\hat{\lambda}_{\text{ridge}} \approx 0$.

Item Feature	**Coefficient**
(intercept)	-3.808
Number of characters	0.002
Word length's standard deviation (characters)	0.809
Distractors–average sentence length (words)	0.002
Dale-Chall index	0.004
FOG index	0.026
Passage and distractors–common words$_1$	0.630
Key option and distractors–word2vec similarity	0.023

Finally, considering Table 6, increased word2vec similarity between key option and distractors is associated with a higher item difficulty ($\hat{\beta} = 0.023 > 0$) (the key option and distractors–word2vec similarity is a similarity of the key option and distractors of the item wording based on word2vec algorithms, where vectors of tokens for both parts are generated and the similarity between them is captured from the context). These features were also detected as important by the importance analysis.

An example of a decision tree, estimating categorized item difficulty as an interval, is in Figure 10. The tree in the figure uses various item features such as the word length's standard deviation (in characters), frequency of uncommon words–according to COCA corpus, item passage and key option–common words$_1$, key option and distractors–number of features from a document-feature matrix, distractors–average sentence length (in words), passage and distractors–word2vec similarity, and number of all characters in item wording. An interpretation is possible and relatively straightforward–in general, if the item's words vary significantly in their lengths, the frequency of uncommon words is high, the proportion of words common for key option and distractors is low enough, item passage and distractors are dissimilar enough, or the item wording is long enough, then the item's difficulty is relatively high.

More specifically, if the word length's standard deviation (in characters) is not lower than 2.3, then the item's difficulty is *difficult* (in $\langle +0.03, +0.52)$) or *very difficult* (in $\langle +0.52, +1.63)$). Otherwise, when the frequency of uncommon words–according to COCA corpus is lower than 0.26, the item's difficulty could be *easy* (in $\langle -0.80, -0.44)$) or *difficult* (in $\langle +0.03, +0.52)$), according to the common words$_1$ among the item passage and key option. Conditional on the previous rules, whenever the number of features from a document-feature matrix of key option and distractors, i.e., a number of common words both in item key option and distractors, is less than 15, the item's difficulty is *very easy* (in $\langle -2.48, -0.80)$) *easy* (in $\langle -0.80, -0.44)$), or could be *difficult* (in in $\langle +0.03, +0.52)$) for the number of characters at least 1062. If the word2vec similarity between passage and distractors is lower than 0.79, then the item difficulty is *moderate* (in $\langle -0.44, +0.03)$). Otherwise, the item difficulty depends on the number of characters in the item wording–usually, if the difficulty could be one of two different difficulty classes, a lower character number in item wording tends to classify the item into the easier class, as we can see in the last-but-one nodes in the tree in Figure 10.

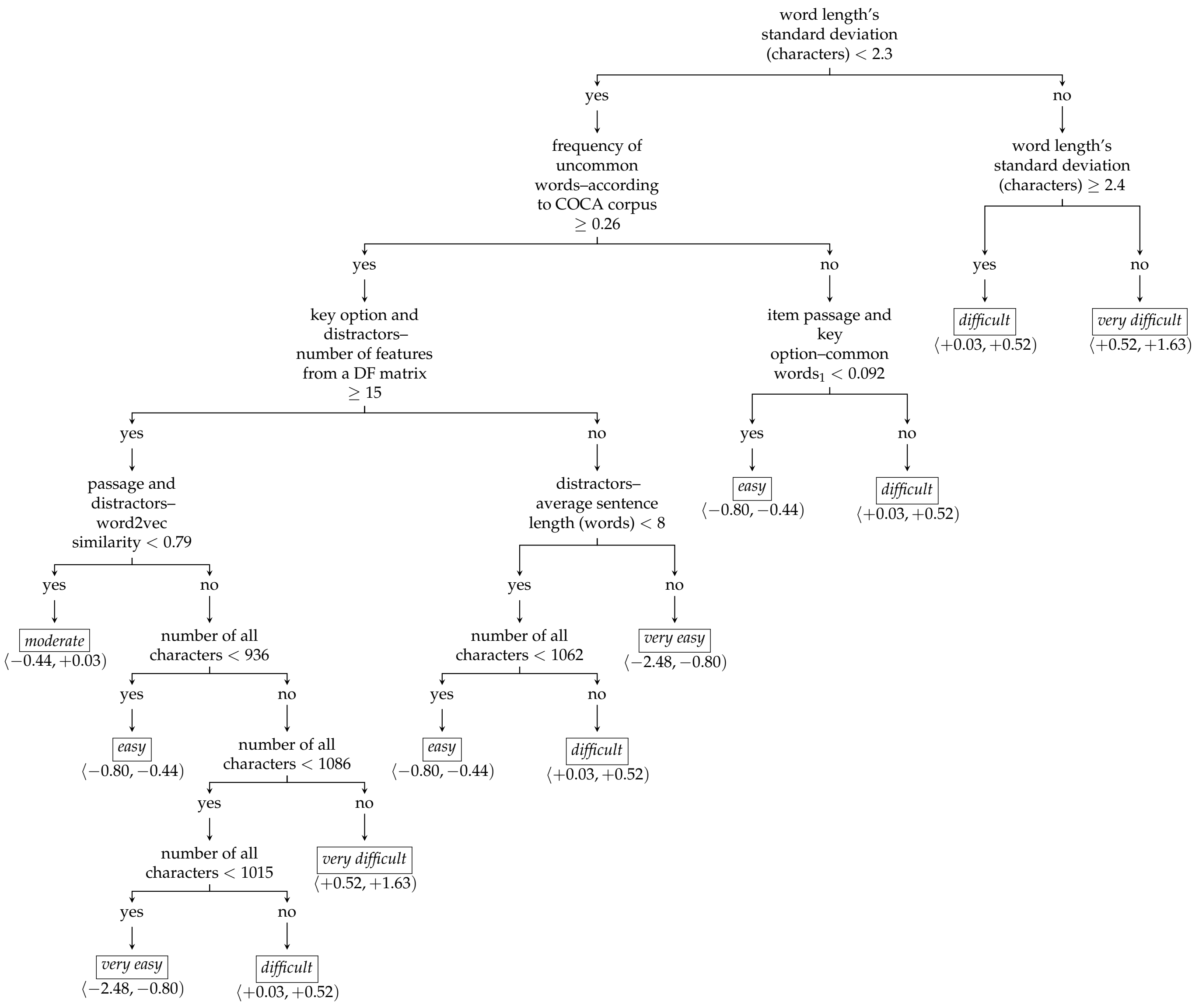

word length's standard deviation (characters) < 2.3
yes
no
frequency of uncommon words–according to COCA corpus ≥ 0.26
word length's standard deviation (characters) ≥ 2.4
yes
no
yes
no
key option and distractors–number of features from a DF matrix ≥ 15
item passage and key option–common words1 < 0.092
difficult ⟨+0.03, +0.52)
very difficult ⟨+0.52, +1.63)
yes
no
yes
no
passage and distractors–word2vec similarity < 0.79
distractors–average sentence length (words) < 8
easy ⟨−0.80, −0.44)
difficult ⟨+0.03, +0.52)
yes
no
yes
no
moderate ⟨−0.44, +0.03)
number of all characters < 936
number of all characters < 1062
very easy ⟨−2.48, −0.80)
yes
no
yes
no
easy ⟨−0.80, −0.44)
number of all characters < 1086
easy ⟨−0.80, −0.44)
difficult ⟨+0.03, +0.52)
yes
no
number of all characters < 1015
very difficult ⟨+0.52, +1.63)
yes
no
very easy ⟨−2.48, −0.80)
difficult ⟨+0.03, +0.52)

Figure 10. An example of a decision tree classifying the categorized item difficulty into a difficulty class (and an appropriate interval).

5. Discussion

In this work, we provided a framework for predicting the difficulty of cognitive test items from their wording. We extracted various text features from English reading comprehension items and employed a number of ML algorithms. Our work is unique in that it compares a wide range of ML algorithms, both for regression and classification tasks, as well as in relating the predictions to those of domain experts. We also provide reproducible R code, which can be used and built on in future studies. The prediction of item difficulty using item text features may save time and resources needed for pre-testing and may help especially in situations when pre-testing is limited or not feasible. ML prediction of item difficulty presented in this work has the potential to be more precise than domain experts, and if not fully replacing domain experts, it may be used to guide and improve their predictions, as well as any imprecise estimates coming from pre-testing based on small or less representative samples.

Among all regression task algorithms, regularization approaches seemed to overcome others, similar to [60,61]. This is expectable given that the amount of data included in the training subset was relatively low. All ML algorithms outperformed domain experts in this task, although the domain experts are handicapped by not using a continuous scale, as mentioned in Section 4. To govern the accuracy-precision trade-off towards higher accuracy [62], we also considered the task of classifying the item difficulty into only a few categories. Domain experts slightly outperformed ML algorithms in the accuracy of difficulty classification when the task was to classify the item difficulty into five categories. From the ML algorithms, the random forests predicted with the highest extended predictive accuracy and performed almost as well as domain experts. We suppose that random forests could return the best predictive performance since this algorithm is a priori ensembled, embedding multiple decision trees.

It is hard to compare our results to those of other studies, given that different studies train ML algorithms on data which may differ in the topic, the number of available items, variability of item content and difficulty, as well as used difficulty scale or difficulty distribution among various parts of the scale. Benedetto et al. in [63] applied ML techniques on multiple true-false questions from CloudAcademy to predict the question difficulty and received RMSE about 0.700–0.900 for random forests, decision trees, support vector machines, and linear regression. In another paper, Benedetto et al. [64] introduced an R2DE model for newly generated items and automatically predicted their difficulty, originating from interval $\langle -5, +5\rangle$ with RMSE of 0.823, which is approximately comparable to our results, i.e., RMSE of 0.668 (elastic net) on item difficulty coming from an interval $\langle -2.48, +1.63\rangle$. Using word embedding and support vector machine with the radial kernel, Ehara in [65] reported RMSE about 3.632 for item difficulty prediction on English vocabulary tests with a pre-estimated difficulty range in $\langle -2, +4\rangle$; since our dataset if of similar difficulty range, we received better performance for item difficulty prediction in case of support vector machines—an RMSE of 0.716. Lee et al. in [66] predicted item difficulty for C-tests, i.e., tests where the second part of every second word is missing and should be fulfilled by a test-taker, and reached an RMSE of 0.240 using advanced architectures of support vector machines and neural networks. Regarding the adaptive scenarios, Pandarova et al. in [67] predicted the difficulty of cued gap-filling items using common item features and several ridge regression models and obtained an RMSE of 0.770. Qiu et al. in [68] trained a document-enhanced attention-based neural network on data from medical online education websites in China to predict the correct-answer ratio (in the range of 0 to 1) and output RMSE of 0.131. They also compared the approach with support vector machines-based prediction, yielding an RMSE of about 0.172, which is, considering their difficulty range $\langle 0, 1\rangle$, comparable with our results. Ha et al. in [69], and Xue et al. in [70] published, besides response times, prediction of item difficulty using medical datasets based on correct-answer ratios (i.e., difficulty in a range of 0 to 1) and employing various ML methods and transfer learning, resulting in an RMSE in the range of 0.200–0.300. Similar approaches and results as Ha et al. in [69] are also reported by Yaneva et al. in [71]. Yin et al. in [72] proposed a new text-embedded and hierarchical pre-trained model QuesNet for item representation, that is able to predict item difficulty, ranged in the interval 0–1, with an RMSE of 0.253. Several studies went deeper into item difficulty classification rather than continuous prediction. Hsu et al. in [73] predicted item difficulty (of five levels, i.e., *very easy*, *easy*, *moderate*, *difficult*, *very difficult*) in social studies tests using semantic spaces and word embedding techniques, by which they reached accuracy about 0.350 and extended accuracy about 0.780. Similar to our study, they also found that semantic similarity between an item stem and the options strongly impacts item difficulty. One year later, Lin et al. in [74] remade the analysis by Hsu and applied long short-term memory on the same problem and datasets; they received an accuracy of 0.370 and extended accuracy of 0.840. Compared with the above-mentioned studies, our analysis is limited by the number of items available for training the ML algorithms, as well as by the relatively low and homogeneous item difficulty related to the level of the exam, which was set to B1 according to the Common European Framework of Reference for Languages (CEFR) standard.

This study opens several paths for further research. One possible path to improving the algorithms presented here is to extend or improve the extracted item text features while keeping in mind that simply boosting a number of item features would not necessarily improve model predictive performance; see Section 2.4.4. We focused on text content rather than context within the item difficulty prediction using their text wording. In our case, various readability indices and indices of similarity between individual parts of item text wording seemed to be important for the difficulty prediction, similarly to [73]. Additionally, considering the elastic net summary, the standard deviation of item words' length (in characters) was of significant importance. The contentual features are easier to extract, while they may reduce information encoded in the textual wording significantly [75]. Further research may consider also incorporating contextual analysis, which, however, also requires extensive samples of textual data [76]. Other future paths include tuning the settings of the involved ML algorithms or even including further ML methods.

Involving a wider range of training datasets is another possible path to follow. Our work focused on predicting item difficulty in the reading comprehension section of the English language test; however, the possible usage of the methods presented here is much wider. Similar methods may find their use in the prediction of item difficulty in other knowledge tests [69,70,77], or to provide a better understanding of the rating of the quality of grant proposals [78,79] when a text complementing numerical ratings is available. Text analysis and ML methods may provide a deeper insight into item-level differences in responding and explain so-called differential item functioning (DIF) [80–82] or item-level between-group differences in change after treatment (differential item functioning in change, DIF-C) [83]. Given the increasing computational power, we expect more research implementing textual data analysis will complement the analysis of rating data in the future.

6. Conclusions

To conclude, the text analysis of item wording may be useful for the prediction of item difficulty, especially when item pre-testing is limited or not available. Machine learning algorithms, particularly regularization or random forests, may be able to inform and improve item difficulty estimates of the domain experts. Future studies should consider more complex and deeper text analysis, including context analysis, as well as other ML methods, and method tuning to even further improve the performance of the item difficulty prediction.

Author Contributions: Conceptualization, L.Š., J.D. and P.M.; Data curation, L.Š., J.D. and P.M.; Formal analysis, L.Š., J.D. and P.M.; Funding acquisition, P.M.; Investigation, L.Š., J.D. and P.M.; Methodology, L.Š., J.D. and P.M.; Project administration, P.M.; Resources, L.Š., J.D. and P.M.; Software, L.Š., J.D. and P.M.; Supervision, P.M.; Validation, L.Š., J.D. and P.M.; Visualization, L.Š.; Writing—original draft, L.Š., J.D. and P.M.; Writing—review & editing, L.Š., J.D. and P.M. All authors have read and agreed to the published version of the manuscript.

Funding: The study was supported by the Czech Science Foundation Grant Number 21-03658S, by the institutional support RVO 67985807, and by the Charles University programme Progres Q15 "Life course, lifestyle and quality of life from the perspective of individual adaptation and the relationship of the actors and institutions".

Data Availability Statement: Data, source code for the ML analysis, and further supplementary material are available at OSF platform at https://osf.io/nzfgk/ (accessed on 27 September 2023). Original data with item wordings are available at https://data.cermat.cz/ (in Czech) (accessed on 30 March 2023).

Acknowledgments: The authors thank the Centre for Evaluation of Educational Achievement for sharing insights on item difficulty evaluation and for data of preliminary difficulty predictions by domain experts. We also thank anonymous reviewers and Eva Potužníková for suggestions to previous versions of the manuscript and Filip Martinek for assistance with software computations.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Appendix A

In this part of the appendix, we describe selected item features and their definitions in more detail, particularly those listed in Tables 4 and 5. The wording of an item usually consists of the following parts: an item passage, a question, a key option, and distractors. The item passage is an introductory text of varying length that mentions important terms or definitions asked in the following item question or describes the item's context. The item question is followed by a permutation of a key option, i.e., a correct answer, and several distractors, i.e., incorrect answers. In summary below, we mark any of the item wording part as $\{\mathcal{A}\}$,

$$\{\mathcal{A}\} \in \{\text{item passage, question, key option, distractors}\},$$

and any pair of the item wording parts as $\{\mathcal{A} \text{ and } \mathcal{B}\}$,

$$\begin{aligned}\{\mathcal{A} \text{ and } \mathcal{B}\} \in \{&\text{key option and distractors,}\\ &\text{item passage and distractors,}\\ &\text{item passage and key option,}\\ &\text{question and distractors,}\\ &\text{question and key option,}\\ &\text{item passage and question}\}.\end{aligned}$$

Each item feature is either a characteristic of an entire item text wording (i.e., there is one numerical value of the item feature for the item) or of each item wording part (i.e., there is one numerical value for each wording part), or a pair of item wording parts. In case the item feature is a numerical characteristic of part $\mathcal{A}$ of the item wording, or pair of parts $\{\mathcal{A} \text{ and } \mathcal{B}\}$ of the item wording, it is indicated below using $\{\mathcal{A}\}$: "item feature label", or $\{\mathcal{A} \text{ and } \mathcal{B}\}$: "item feature label" notation, respectively.

Item Feature	**Description or Definition of the Item Feature**
number of characters	Total number of characters in a text of the item wording.
$\{\mathcal{A}\}$–number of characters	Total number of characters in a text of part $\mathcal{A}$ of the item wording.
number of tokens	Total number of unique tokens, i.e., words in a text of the item wording.
$\{\mathcal{A}\}$–number of tokens	Total number of unique tokens, i.e., words in a text of part $\mathcal{A}$ of the item wording.
number of monosyllabic words	Number of monosyllabic words, i.e., words with only one syllable in a text of the item wording.
$\{\mathcal{A}\}$–number of monosyllabic words	Number of monosyllabic words, i.e., words with only one syllable in a text of part $\mathcal{A}$ of the item wording.
number of multi-syllable words	Number of multi-syllable words, i.e., words with more than three syllables in a text of the item wording.
$\{\mathcal{A}\}$–number of multi-syllable words	Number of multi-syllable words, i.e., words with more than three syllables in a text of part $\mathcal{A}$ of the item wording.
average word length (characters)	Average number of characters in words in a text of the item wording.
$\{\mathcal{A}\}$–average word length (characters)	Average number of characters in words in a text of part $\mathcal{A}$ of the item wording.
longest word length (characters)	Number of characters contained by the longest word in a text of the item wording.

$\{\mathcal{A}\}$–longest word length (characters)	Number of characters contained by the longest word in a text of part $\mathcal{A}$ of the item wording.
average sentence length (words)	Average number of words in sentences in a text of the item wording.
$\{\mathcal{A}\}$–average sentence length (words)	Average number of words in sentences in a text of part $\mathcal{A}$ of the item wording.
word length's standard deviation (characters)	Standard deviation of a number of characters in words in a text of the item wording.
$\{\mathcal{A}\}$–word length's standard deviation (characters)	Standard deviation of a number of characters in words in a text of part $\mathcal{A}$ of the item wording.
number of uncommon words, according to COCA corpus	Number of words in a text of the item wording that appear *uncommonly* as defined in COCA (Corpus of Contemporary American English) corpus.
number of rare words, according to COCA corpus	Number of words in a text of the item wording that appear *rarely* as defined in COCA (Corpus of Contemporary American English) corpus.
frequency of the A1 words (CEFR)	Frequency of words in a text of the item wording at A1 level in CEFR (Common European Framework of Reference for Languages) scale.
frequency of the B2–C2 words (CEFR)	Frequency of words in a text of the item wording at B2–C2 levels in CEFR (Common European Framework of Reference for Languages) scale.
number of footnotes (hints) in the item	Total number of footnotes or hints in a text of the item wording.
Dale-Chall index	The readability score of a text of the item wording based on Dale-Chall readability formula. Dale-Chall readability formula follows, $$\text{Dale-Chall index} = \left(95 \cdot \frac{n_{\text{difficult}}}{n_w}\right) - (0.69 \cdot \overline{w}),$$ where $n_{\text{difficult}}$ is a number of words not included in Dale-Chall list of 3000 familiar words, n_w is a total number of words in a text of the item wording, and $\overline{w}$ is a value computed as a number of words divided by a number of sentences, i.e., it is an average number of words per a sentence [84]. The greater is a value of Dale-Chall index for a given text, the more difficult is to read the text.
FOG index	The readability score of a text of the item wording based on Gunning's Fog Index. The formula is $$\text{FOG index} = 0.4 \cdot \left(\overline{w} + 100 \cdot \frac{n_{\text{words with} \geq 3 \text{ syllables}}}{n_w}\right),$$ where, again, $\overline{w}$ is a value computed as a number of words divided by a number of sentences, i.e., it is an average number of words per a sentence, n_w is a total number of words in a text of the item wording, and $n_{\text{words with} \geq 3 \text{ syllables}}$ is a number of words with three or more syllables in a text of the item wording [85]. If the average length of a sentence or the number of words with three or more syllables in a text increases, the FOG index increases, too.

SMOG index	The readability score of a text of the item wording based on Simple Measure of Gobbledygook (SMOG) index, so $$\text{SMOG index} = 1.043 \cdot \sqrt{n_{\text{words with} \geq 3 \text{ syllables}}} \cdot \left(\frac{30}{n_s}\right) + 3.129,$$ where $n_{\text{words with} \geq 3 \text{ syllables}}$ is a number of words with three or more syllables in a text of the item wording and n_s is a number of sentences in a text of the item wording [86]. Whenever the term $\frac{\sqrt{n_{\text{words with} \geq 3 \text{ syllables}}}}{n_s}$ increases, i.e., the square root of a number of words with three or more syllables per a sentence, readability increases in difficulty and the SMOG index increases.
Traenkle-Bailer index	The readability score of a text of the item wording based on Traenkle-Bailer index (mostly used in German-speaking countries) is calculated as $$\text{T-B index} = 224.68 - (79.83 \cdot \bar{c}) - (12.24 \cdot \overline{w}) - \left(129.29 \cdot \frac{n_{\text{prep}}}{n_w}\right),$$ where $\bar{c}$ is an average number of characters per a word, $\overline{w}$ is an average number of words per a sentence, n_{prep} is a number of prepositions and n_w is a total number of words in a text of the item wording [87]. Traenkle-Bailer index decreases, if the average number of characters per a word, average number of words per a sentence, or average number of prepositions per a word increases.
$\{\mathcal{A}$ and $\mathcal{B}\}$–euclidean distance	Let us assume two textual parts of item wording, $\mathcal{A}$ and $\mathcal{B}$, so that a union of their tokens has a length $l \in \mathbb{N}$. Additionally, let us assume two vectors of the same length l, i.e., $\boldsymbol{t}_{\mathcal{A}} = (t_{\mathcal{A},1}, t_{\mathcal{A},2}, \ldots, t_{\mathcal{A},l})^T$ and $\boldsymbol{t}_{\mathcal{B}} = (t_{\mathcal{B},1}, t_{\mathcal{B},2}, \ldots, t_{\mathcal{B},l})^T$, where $t_{\mathcal{A},i} = 1$ (or $t_{\mathcal{B},i} = 1$) if and only if text $\mathcal{A}$ (text $\mathcal{B}$) contains token i, otherwise is $t_{\mathcal{A},i} = 0$ (or $t_{\mathcal{B},i} = 0$), for $\forall i \in \{1, 2, \ldots, l\}$. The euclidean distance between the parts $\mathcal{A}$ and $\mathcal{B}$ is $$d(\mathcal{A}, \mathcal{B}) = \sqrt{\sum_{i=1}^{l} \left(t_{\mathcal{A},i} - t_{\mathcal{B},i}\right)^2}.$$ The more similar the parts $\mathcal{A}$ and $\mathcal{B}$ of the item wording are, the lower the value of euclidean distance $d(\mathcal{A}, \mathcal{B})$ is.
$\{\mathcal{A}$ and $\mathcal{B}\}$–cosine similarity	Again, let us assume two textual parts of item wording, $\mathcal{A}$ and $\mathcal{B}$, so that a union of their tokens has a length $l \in \mathbb{N}$. Additionally, let us assume two vectors of the same length l, i.e., $\boldsymbol{t}_{\mathcal{A}} = (t_{\mathcal{A},1}, t_{\mathcal{A},2}, \ldots, t_{\mathcal{A},l})^T$ and $\boldsymbol{t}_{\mathcal{B}} = (t_{\mathcal{B},1}, t_{\mathcal{B},2}, \ldots, t_{\mathcal{B},l})^T$, where $t_{\mathcal{A},i} = 1$ (or $t_{\mathcal{B},i} = 1$) if and only if text $\mathcal{A}$ (text $\mathcal{B}$) contains token i, otherwise is $t_{\mathcal{A},i} = 0$ (or $t_{\mathcal{B},i} = 0$), for $\forall i \in \{1, 2, \ldots, l\}$. The cosine similarity between the parts $\mathcal{A}$ and $\mathcal{B}$ is $$\cos(\mathcal{A}, \mathcal{B}) = \frac{\boldsymbol{t}_{\mathcal{A}} \cdot \boldsymbol{t}_{\mathcal{B}}}{\lVert \boldsymbol{t}_{\mathcal{A}} \rVert \, \lVert \boldsymbol{t}_{\mathcal{B}} \rVert} = \frac{\sum_{i=1}^{l} t_{\mathcal{A},i} \cdot t_{\mathcal{B},i}}{\sqrt{\sum_{i=1}^{l} t_{\mathcal{A},i}^2} \cdot \sqrt{\sum_{i=1}^{l} t_{\mathcal{B},i}^2}}$$ The more similar the parts $\mathcal{A}$ and $\mathcal{B}$ of the item wording are, the higher the value of cosine similarity $\cos(\mathcal{A}, \mathcal{B})$ is.

$\{\mathcal{A}$ and $\mathcal{B}\}$–word2vec similarity	Similarity of parts $\mathcal{A}$ and $\mathcal{B}$ of the item wording based on word2vec algorithms. Vectors of tokens for each part $\mathcal{A}$ and $\mathcal{B}$ are generated, and the similarity between them is captured from the context [88]. Thus, text parts with similar context end up with similar vectors and high word2vec similarity.
$\{\mathcal{A}$ and $\mathcal{B}\}$–common words$_1$	Proportion of common words from a text of part $\mathcal{A}$ found in a text of part $\mathcal{B}$ of the item wording. Let us assume two textual parts of item wording, $\mathcal{A}$ and $\mathcal{B}$, so that a union of their tokens, called also *document-feature matrix* has a length $l \in \mathbb{N}$. Additionally, let us assume two vectors of the same length l, i.e., $\boldsymbol{t}_{\mathcal{A}} = (t_{\mathcal{A},1}, t_{\mathcal{A},2}, \dots, t_{\mathcal{A},l})^T$ and $\boldsymbol{t}_{\mathcal{B}} = (t_{\mathcal{B},1}, t_{\mathcal{B},2}, \dots, t_{\mathcal{B},l})^T$, where $t_{\mathcal{A},i} = 1$ (or $t_{\mathcal{B},i} = 1$) if and only if text $\mathcal{A}$ (text $\mathcal{B}$) contains token i, otherwise is $t_{\mathcal{A},i} = 0$ (or $t_{\mathcal{B},i} = 0$), for $\forall i \in \{1, 2, \dots, l\}$. Then the $\{\mathcal{A}$ and $\mathcal{B}\}$–common words$_1$ is $$\{\mathcal{A} \text{ and } \mathcal{B}\}\text{–common words}_1 = \frac{\boldsymbol{t}_{\mathcal{A}} \cdot \boldsymbol{t}_{\mathcal{B}}}{\|\boldsymbol{t}_{\mathcal{A}}\|^2} = \frac{\sum_{i=1}^{l} t_{\mathcal{A},i} \cdot t_{\mathcal{B},i}}{\sum_{i=1}^{l} t_{\mathcal{A},i}^2} = = \frac{\sum_{i=1}^{l} t_{\mathcal{A},i} \cdot t_{\mathcal{B},i}}{\sum_{i=1}^{l} t_{\mathcal{A},i}}.$$
$\{\mathcal{A}$ and $\mathcal{B}\}$–common words$_2$	Proportion of common words from a text of part $\mathcal{B}$ found in a text of part $\mathcal{A}$ of the item wording. Let us assume two textual parts of item wording, $\mathcal{A}$ and $\mathcal{B}$, so that a union of their tokens, called also *document-feature matrix* has a length $l \in \mathbb{N}$. Additionally, let us assume two vectors of the same length l, i.e., $\boldsymbol{t}_{\mathcal{A}} = (t_{\mathcal{A},1}, t_{\mathcal{A},2}, \dots, t_{\mathcal{A},l})^T$ and $\boldsymbol{t}_{\mathcal{B}} = (t_{\mathcal{B},1}, t_{\mathcal{B},2}, \dots, t_{\mathcal{B},l})^T$, where $t_{\mathcal{A},i} = 1$ (or $t_{\mathcal{B},i} = 1$) if and only if text $\mathcal{A}$ (text $\mathcal{B}$) contains token i, otherwise is $t_{\mathcal{A},i} = 0$ (or $t_{\mathcal{B},i} = 0$), for $\forall i \in \{1, 2, \dots, l\}$. Then the $\{\mathcal{A}$ and $\mathcal{B}\}$–common words$_2$ is $$\{\mathcal{A} \text{ and } \mathcal{B}\}\text{–common words}_2 = \frac{\boldsymbol{t}_{\mathcal{A}} \cdot \boldsymbol{t}_{\mathcal{B}}}{\|\boldsymbol{t}_{\mathcal{B}}\|^2} = \frac{\sum_{i=1}^{l} t_{\mathcal{A},i} \cdot t_{\mathcal{B},i}}{\sum_{i=1}^{l} t_{\mathcal{B},i}^2} = = \frac{\sum_{i=1}^{l} t_{\mathcal{A},i} \cdot t_{\mathcal{B},i}}{\sum_{i=1}^{l} t_{\mathcal{B},i}}.$$
$\{\mathcal{A}$ and $\mathcal{B}\}$–number of features from a document-feature (DF) matrix	Let us assume two textual parts of item wording, $\mathcal{A}$ and $\mathcal{B}$, so that a union of their tokens, called also *document-feature matrix* (abbreviated as DF matrix) has a length $l \in \mathbb{N}$. Additionally, let us assume two vectors of the same length l, i.e., $\boldsymbol{t}_{\mathcal{A}} = (t_{\mathcal{A},1}, t_{\mathcal{A},2}, \dots, t_{\mathcal{A},l})^T$ and $\boldsymbol{t}_{\mathcal{B}} = (t_{\mathcal{B},1}, t_{\mathcal{B},2}, \dots, t_{\mathcal{B},l})^T$, where $t_{\mathcal{A},i} = 1$ (or $t_{\mathcal{B},i} = 1$) if and only if text $\mathcal{A}$ (text $\mathcal{B}$) contains token i, otherwise is $t_{\mathcal{A},i} = 0$ (or $t_{\mathcal{B},i} = 0$), for $\forall i \in \{1, 2, \dots, l\}$. The number of features from a document-feature matrix for the parts $\mathcal{A}$ and $\mathcal{B}$ is equal to $$\|\boldsymbol{t}_{\mathcal{A}}\|^2 + \|\boldsymbol{t}_{\mathcal{B}}\|^2 = \sum_{i=1}^{l} t_{\mathcal{A},i}^2 + \sum_{i=1}^{l} t_{\mathcal{B},i}^2 = \sum_{i=1}^{l} t_{\mathcal{A},i} + \sum_{i=1}^{l} t_{\mathcal{B},i}.$$ Obviously, since $\forall i \in \{1, 2, \dots, l\}$ is either $t_{\mathcal{A},i} = 1$, or $t_{\mathcal{B},i} = 1$, or $t_{\mathcal{A},i} = t_{\mathcal{B},i} = 1$, it is also $$l \leq \|\boldsymbol{t}_{\mathcal{A}}\|^2 + \|\boldsymbol{t}_{\mathcal{B}}\|^2 = \sum_{i=1}^{l} t_{\mathcal{A},i}^2 + \sum_{i=1}^{l} t_{\mathcal{B},i}^2 = \sum_{i=1}^{l} t_{\mathcal{A},i} + \sum_{i=1}^{l} t_{\mathcal{B},i} \leq 2l.$$

References

1. Martinková, P.; Hladká, A. *Computational Aspects of Psychometric Methods: With R*; CRC Press: Boca Raton, FL, USA, 2023.
2. Kumar, V.; Boulanger, D. Explainable Automated Essay Scoring: Deep Learning Really Has Pedagogical Value. *Front. Educ.* **2020**, *5*, 572367.
3. Amorim, E.; Cançado, M.; Veloso, A. Automated Essay Scoring in the Presence of Biased Ratings. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; Association for Computational Linguistics: New Orleans, LA, USA, 2018; 1 (Long Papers), pp. 229–237.
4. Tashu, T.M.; Maurya, C.K.; Horvath, T. Deep Learning Architecture for Automatic Essay Scoring. *arXiv* **2022**, arXiv:2206.08232. [CrossRef]
5. Flor, M.; Hao, J. *Text Mining and Automated Scoring*; Springer International Publishing: Cham, Switzerland, 2021; pp. 245–262. [CrossRef]
6. Attali, Y.; Runge, A.; LaFlair, G.T.; Yancey, K.; Goodwin, S.; Park, Y.; Davier, A.A.v. The interactive reading task: Transformer-based automatic item generation. *Front. Artif. Intell.* **2022**, *5*, 903077. [CrossRef]
7. Gierl, M.J.; Lai, H.; Turner, S.R. Using automatic item generation to create multiple-choice test items. *Med. Educ.* **2012**, *46*, 757–765. [CrossRef]
8. Du, X.; Shao, J.; Cardie, C. Learning to Ask: Neural Question Generation for Reading Comprehension. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; Association for Computational Linguistics: Vancouver, Canada, 2017; pp. 1342–1352. [CrossRef]
9. Settles, B.; T LaFlair, G.; Hagiwara, M. Machine learning–driven language assessment. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 247–263. [CrossRef]
10. Kochmar, E.; Vu, D.D.; Belfer, R.; Gupta, V.; Serban, I.V.; Pineau, J. Automated Data-Driven Generation of Personalized Pedagogical Interventions in Intelligent Tutoring Systems. *Int. J. Artif. Intell. Educ.* **2022**, *32*, 323–349. [CrossRef]
11. Gopalakrishnan, K.; Dhiyaneshwaran, N.; Yugesh, P. Online proctoring system using image processing and machine learning. *Int. J. Health Sci.* **2022**, *6*, 891–899. [CrossRef]
12. Kaddoura, S.; Popescu, D.E.; Hemanth, J.D. A systematic review on machine learning models for online learning and examination systems. *PeerJ Comput. Sci.* **2022**, *8*, e986. [CrossRef]
13. Kamalov, F.; Sulieman, H.; Santandreu Calonge, D. Machine learning based approach to exam cheating detection. *PLoS ONE* **2021**, *16*, e0254340. [CrossRef]
14. von Davier, M.; Tyack, L.; Khorramdel, L. Scoring Graphical Responses in TIMSS 2019 Using Artificial Neural Networks. *Educ. Psychol. Meas.* **2023**, *83*, 556–585.
15. von Davier, M.; Tyack, L.; Khorramdel, L. Automated Scoring of Graphical Open-Ended Responses Using Artificial Neural Networks. *arXiv* **2022**, arXiv:2201.01783. [CrossRef]
16. von Davier, A.A.; Mislevy, R.J.; Hao, J. (Eds.) *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment: With Examples in R and Python*; Methodology of Educational Measurement and Assessment; Springer International Publishing: Cham, Switzerland, 2021. [CrossRef]
17. Hvitfeldt, E.; Silge, J. *Supervised Machine Learning for Text Analysis in R*; Chapman and Hall/CRC: Boca Raton, FL, USA, 2021.
18. Ferrara, S.; Steedle, J.T.; Frantz, R.S. Response demands of reading comprehension test items: A review of item difficulty modeling studies. *Appl. Meas. Educ.* **2022**, *35*, 237–253. [CrossRef]
19. Belov, D.I. Predicting Item Characteristic Curve (ICC) Using a Softmax Classifier. In *Proceedings of the Annual Meeting of the Psychometric Society*; Springer: Cham, Switzerland, 2022; pp. 171–184. [CrossRef]
20. AlKhuzaey, S.; Grasso, F.; Payne, T.R.; Tamma, V. A systematic review of data-driven approaches to item difficulty prediction. In *Lecture Notes in Computer Science*; Lecture notes in computer science; Springer International Publishing: Cham, Switzerland, 2021; pp. 29–41.
21. James, G.; Witten, D.; Hastie, T.; Tibshirani, R. *An Introduction to Statistical Learning*; Springer: New York, NY, USA, 2021.
22. Jurafsky, D. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*; Pearson Prentice Hall: Upper Saddle River, NJ, USA, 2009.
23. Chomsky, N. Three models for the description of language. *IEEE Trans. Inf. Theory* **1956**, *2*, 113–124. [CrossRef]
24. Davies, M. The Corpus of Contemporary American English (COCA). 2008. Available online: http://corpus.byu.edu/coca/ (accessed on 29 June 2023).
25. Davies, M. Most Frequent 100,000 Word Forms in English (Based on Data from the COCA Corpus). 2011. Available online: https://www.wordfrequency.info/ (accessed on 29 June 2023).
26. Tonelli, S.; Tran Manh, K.; Pianta, E. Making Readability Indices Readable. In *Proceedings of the First Workshop on Predicting and Improving Text Readability for Target Reader Populations*; Association for Computational Linguistics: Montréal, QC, Canada, 2012; pp. 40–48.
27. Rasch, G. *Probabilistic Models for Some Intelligence and Attainment Tests*; The University of Chicago Press: Chicago, IL, USA, 1993.
28. Debelak, R.; Strobl, C.; Zeigenfuse, M.D. *An introduction to the Rasch Model with Examples in R*; CRC Press: Boca Raton, FL, USA, 2022.
29. Alpaydin, E. *Introduction to Machine Learning*; MIT Press: Cambridge, MA, USA, 2010.

30. Tibshirani, R. Regression Shrinkage and Selection Via the Lasso. *J. R. Stat. Soc. Ser. (Methodol.)* **1996**, *58*, 267–288. [CrossRef]
31. Hoerl, A.E.; Kennard, R.W. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics* **1970**, *12*, 55–67. [CrossRef]
32. Tuia, D.; Flamary, R.; Barlaud, M. To be or not to be convex? A study on regularization in hyperspectral image classification. In Proceedings of the 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, Italy, 26–31 July 2015; IEEE: Piscataway, NJ, USA, 2015. [CrossRef]
33. Zou, H.; Hastie, T. Regularization and Variable Selection Via the Elastic Net. *J. R. Stat. Soc. Ser. Stat. Methodol.* **2005**, *67*, 301–320. [CrossRef]
34. Fan, J.; Li, R. Comment: Feature Screening and Variable Selection via Iterative Ridge Regression. *Technometrics* **2020**, *62*, 434–437. [CrossRef]
35. Friedman, N.; Geiger, D.; Goldszmidt, M. Bayesian Network Classifiers. *Mach. Learn.* **1997**, *29*, 131–163. [CrossRef]
36. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
37. Schölkopf, B. The Kernel Trick for Distances. In Proceedings of the 13th International Conference on Neural Information Processing Systems (NIPS'00), Hong Kong, China, 3–6 October 2006; MIT Press: Cambridge, MA, USA, 2000; pp. 283–289.
38. Gray, N.A.B. Capturing knowledge through top-down induction of decision trees. *IEEE Expert* **1990**, *5*, 41–50. [CrossRef]
39. Breslow, L.A.; Aha, D.W. Simplifying decision trees: A survey. *Knowl. Eng. Rev.* **1997**, *12*, 1–40. [CrossRef]
40. Rutkowski, L.; Jaworski, M.; Pietruczuk, L.; Duda, P. The CART Decision Tree for Mining Data Streams. *Inf. Sci.* **2014**, *266*, 1–15. [CrossRef]
41. Breiman, L. *Classification and Regression Trees*; Chapman & Hall: New York, NY, USA, 1993.
42. McCulloch, W.S.; Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.* **1943**, *5*, 115–133. [CrossRef]
43. Rojas, R. The Backpropagation Algorithm. In *Neural Networks*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 149–182. [CrossRef]
44. Mishra, M.; Srivastava, M. A view of Artificial Neural Network. In Proceedings of the 2014 International Conference on Advances in Engineering & Technology Research (ICAETR-2014), Unnao, Kanpur, India, 1–2 August 2014; IEEE: Piscataway, NJ, USA, 2014. [CrossRef]
45. Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning*; Springer: New York, NY, USA, 2009. [CrossRef]
46. Altmann, A.; Toloşi, L.; Sander, O.; Lengauer, T. Permutation importance: A corrected feature importance measure. *Bioinformatics* **2010**, *26*, 1340–1347. [CrossRef] [PubMed]
47. Powers, D.M.W. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv* **2020**, arXiv:2010.16061.
48. Provost, F.J.; Fawcett, T.; Kohavi, R. The Case against Accuracy Estimation for Comparing Induction Algorithms. In Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98), Madison, WI, USA, 24–27 July 1998; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1998; pp. 445–453.
49. Moore, A.W.; Lee, M.S. Efficient algorithms for minimizing cross validation error. In Proceedings of the 11th International Conference on Machine Learning, New Brunswick, NJ, USA, 10–13 July 1994; Morgan Kaufmann: Burlington, MA, USA, 1994; pp. 190–198.
50. Kohavi, R. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In Proceedings of the 14th International Joint Conference on Artificial Intelligence–Volume 2 (IJCAI'95), Montréal, QC, Canada, 20–25 August 1995; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1995; pp. 1137–1143.
51. R Core Team. *R: A Language and Environment for Statistical Computing*; R Core Team: Vienna, Austria, 2021.
52. Mair, P.; Hatzinger, R.; Maier, M.J.; Rusch, T.; Debelak, R. eRm: Extended Rasch Modeling. 2021. Available online: https://cran.r-project.org/web/packages/eRm/index.html (accessed on 29 June 2023).
53. Benoit, K.; Watanabe, K.; Wang, H.; Nulty, P.; Obeng, A.; Müller, S.; Matsuo, A. Quanteda: An R Package for the Quantitative Analysis of Textual Data. *J. Open Source Softw.* **2018**, *3*, 774. [CrossRef]
54. Friedman, J.; Tibshirani, R.; Hastie, T. Regularization Paths for Generalized Linear Models via Coordinate Descent. *J. Stat. Softw.* **2010**, *33*, 1–22. [CrossRef]
55. Meyer, D.; Dimitriadou, E.; Hornik, K.; Weingessel, A.; Leisch, F. e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien. 2023. R Package Version 1.7-13. Available online: https://rdrr.io/rforge/e1071/ (accessed on 29 June 2023).
56. Therneau, T.; Atkinson, B. rpart: Recursive Partitioning and Regression Trees, 2022. R Package Version 4.1.19. Available online: https://cogns.northwestern.edu/cbmg/LiawAndWiener2002.pdf (accessed on 29 June 2023).
57. Liaw, A.; Wiener, M. Classification and Regression by Random Forest. *R News* **2002**, 2, 18–22.
58. Fritsch, S.; Guenther, F.; Wright, M.N. *neuralnet: Training of Neural Networks*, 2019. R Package Version 1.44.2. Available online: https://journal.r-project.org/archive/2010/RJ-2010-006/RJ-2010-006.pdf (accessed on 29 June 2023).
59. Craig, C.C. A Note on Sheppard's Corrections. *Ann. Math. Stat.* **1941**, *12*, 339–345. [CrossRef]
60. Chen, J.; de Hoogh, K.; Gulliver, J.; Hoffmann, B.; Hertel, O.; Ketzel, M.; Bauwelinck, M.; van Donkelaar, A.; Hvidtfeldt, U.A.; Katsouyanni, K.; et al. A comparison of linear regression, regularization, and machine learning algorithms to develop Europe-wide spatial models of fine particles and nitrogen dioxide. *Environ. Int.* **2019**, *130*, 104934. [CrossRef]

61. Dong, Y.; Zhou, S.; Xing, L.; Chen, Y.; Ren, Z.; Dong, Y.; Zhang, X. Deep learning methods may not outperform other machine learning methods on analyzing genomic studies. *Front. Genet.* **2022**, *13*, 992070. [CrossRef]
62. Su, J.; Fraser, N.J.; Gambardella, G.; Blott, M.; Durelli, G.; Thomas, D.B.; Leong, P.; Cheung, P.Y.K. Accuracy to Throughput Trade-offs for Reduced Precision Neural Networks on Reconfigurable Logic. *arXiv* **2018**, arXiv:1807.10577. [CrossRef]
63. Benedetto, L.; Cappelli, A.; Turrin, R.; Cremonesi, P. Introducing a Framework to Assess Newly Created Questions with Natural Language Processing. In *Lecture Notes in Computer Science*; Springer International Publishing: Cham, Switzerland, 2020; pp. 43–54. [CrossRef]
64. Benedetto, L.; Cappelli, A.; Turrin, R.; Cremonesi, P. R2DE: A NLP approach to estimating IRT parameters of newly generated questions. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*; ACM: New York, NY, USA, 2020. [CrossRef]
65. Ehara, Y. Building an English Vocabulary Knowledge Dataset of Japanese English-as-a-Second-Language Learners Using Crowdsourcing. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan, 7–12 May 2018; European Language Resources Association (ELRA): Miyazaki, Japan, 2018.
66. Lee, J.U.; Schwan, E.; Meyer, C.M. Manipulating the Difficulty of C-Tests. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; Association for Computational Linguistics: Florence, Italy, 2019; pp. 360–370. [CrossRef]
67. Pandarova, I.; Schmidt, T.; Hartig, J.; Boubekki, A.; Jones, R.D.; Brefeld, U. Predicting the Difficulty of Exercise Items for Dynamic Difficulty Adaptation in Adaptive Language Tutoring. *Int. J. Artif. Intell. Educ.* **2019**, *29*, 342–367. [CrossRef]
68. Qiu, Z.; Wu, X.; Fan, W. Question Difficulty Prediction for Multiple Choice Problems in Medical Exams. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; ACM: New York, NY, USA, 2019.
69. Ha, L.A.; Yaneva, V.; Baldwin, P.; Mee, J. Predicting the Difficulty of Multiple Choice Questions in a High-stakes Medical Exam. In Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, Florence, Italy, 2 August 2019; Association for Computational Linguistics: Florence, Italy, 2019; pp. 11–20. [CrossRef]
70. Xue, K.; Yaneva, V.; Runyon, C.; Baldwin, P. Predicting the Difficulty and Response Time of Multiple Choice Questions Using Transfer Learning. In Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications, Online, 10 July 2020; Association for Computational Linguistics: Seattle, WA, USA, 2020; pp. 193–197. [CrossRef]
71. Yaneva, V.; Ha, L.A.; Baldwin, P.; Mee, J. Predicting Item Survival for Multiple Choice Questions in a High-Stakes Medical Exam. In Proceedings of the Twelfth Language Resources and Evaluation Conference,Marseille, France, 11–16 May 2020; European Language Resources Association: Marseille, France, 2020; pp. 6812–6818.
72. Yin, Y.; Liu, Q.; Huang, Z.; Chen, E.; Tong, W.; Wang, S.; Su, Y. QuesNet. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; ACM: New York, NY, USA, 2019.
73. Hsu, F.Y.; Lee, H.M.; Chang, T.H.; Sung, Y.T. Automated estimation of item difficulty for multiple-choice tests: An application of word embedding techniques. *Inf. Process. Manag.* **2018**, *54*, 969–984. [CrossRef]
74. Lin, L.H.; Chang, T.H.; Hsu, F.Y. Automated Prediction of Item Difficulty in Reading Comprehension Using Long Short-Term Memory. In Proceedings of the 2019 International Conference on Asian Language Processing (IALP), Shanghai, China, 15–17 November 2019; IEEE: Piscataway, NJ, USA, 2019.
75. McTavish, D.G.; Pirro, E.B. Contextual content analysis. *Qual. Quant.* **1990**, *24*, 245–265. [CrossRef]
76. Stipak, B.; Hensler, C. Statistical Inference in Contextual Analysis. *Am. J. Political Sci.* **1982**, *26*, 151. [CrossRef]
77. Martinková, P.; Štěpánek, L.; Drabinová, A.; Houdek, J.; Vejražka, M.; Štuka, Č. Semi-real-time analyses of item characteristics for medical school admission tests. In Proceedings of the 2017 Federated Conference on Computer Science and Information Systems, Prague, Czech Republic, 3–6 September 2017; IEEE: Piscataway, NJ, USA, 2017.
78. Erosheva, E.A.; Martinková, P.; Lee, C.J. When zero may not be zero: A cautionary note on the use of inter-rater reliability in evaluating grant peer review. *J. R. Stat. Soc. Ser. (Stat. Soc.)* **2021**, *184*, 904–919. [CrossRef]
79. Van den Besselaar, P.; Sandström, U.; Schiffbaenker, H. Studying grant decision-making: A linguistic analysis of review reports. *Scientometrics* **2018**, *117*, 313–329. [CrossRef]
80. Penfield, R.D.; Camilli, G. Differential item functioning and item bias. In *Psychometrics*; Rao, C.R., Sinharay, S., Eds.; Handbook of Statistics; Elsevier: Amsterdam, The Netherlands, 2006; Volume 26, pp. 125–167. [CrossRef]
81. Martinková, P.; Drabinová, A.; Liaw, Y.L.; Sanders, E.A.; McFarland, J.L.; Price, R.M. Checking equity: Why differential item functioning analysis should be a routine part of developing conceptual assessments. *CBE-Life Sci. Educ.* **2017**, *16*, rm2. [CrossRef]
82. Hladká, A.; Martinková, P. difNLR: Generalized Logistic Regression Models for DIF and DDF Detection. *R J.* **2020**, *12*, 300–323. [CrossRef]
83. Martinková, P.; Hladká, A.; Potužníková, E. Is academic tracking related to gains in learning competence? Using propensity score matching and differential item change functioning analysis for better understanding of tracking implications. *Learn. Instr.* **2020**, *66*, 101286. [CrossRef]
84. Chall, J.S.; Dale, E. *Readability REVISITED: The New Dale-Chall Readability Formula*; Brookline Books: Cambridge, MA, USA, 1995.
85. Gunning, R. *The Technique of Clear Writing*; McGraw-Hill: New York, NY, USA, 1952.
86. McLaughlin, G.H. SMOG Grading: A New Readability Formula. *J. Read.* **1969**, *12*, 639–646.

87. Tränkle, U.; Bailer, H. Kreuzvalidierung und Neuberechnung von Lesbarkeitsformeln für die deutsche Sprache. *Zeitschrift für Entwicklungspsychologie und Pädagogische Psychologie* **1984**, *16*, 231–244.
88. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**. arXiv:1301.3781.