Master Thesis

Idea Quality Classification in Ideation Contests: Leveraging Textual and Non-Textual Features with Machine Learning

By: Darnell Kluit

Student number: s2216159

Supervisors:

Dr. A. Leszkiewicz

Dr. I. Skute

Faculty:

BMS

Business Administration - Digital Business

Abstract

This research paper investigates the evaluation of idea quality in innovation contests using a machine learning approach. The study aims to explore the factors contributing to higher ranked ideas by equating idea quality with performance in such contests. It examines the role of machine learning models in evaluating idea quality and compares their performance to a baseline no information rate. As the dataset size increases, the cross-validated machine learning models approach statistical significance. The research identifies several factors that significantly impact the performance of machine learning models. These include team size, the level of elaboration in the idea challenge and solution, and the readability of the idea itself. The study also delves into the influence of feedback quality on idea quality, specifically analysing the relationship between feedback sources' expertise and the idea topic. It suggests that diverse expertise among coaches positively impacts the perceived quality of ideas. This paper offers a valuable framework for assessing idea quality based on contest rankings. The insights gained from team size, idea elaboration, and feedback expertise provide practical guidance for participants and organisers to enhance idea quality and maximize the benefits of innovation contests.

Contents

Introduction

A popular way to produce new ideas for problem solving, design creation, and product or process improvement is ideation contests. In this type of contest, a firm or an organisation (the seeker) posts an ideation task online to a distributed group of independent agents (solvers) through an open invitation. The solvers then submit their ideas, and the seeker selects the best idea or ideas and rewards the solver(s) with a prize..

Ideation contests are relevant because they provide a way for organisations to leverage the shared intelligence of a large and diverse group of people. They can be used to generate new ideas for products, services, and processes, and to solve complex problems that require a fresh perspective. Ideation contests can also be used to engage employees and customers in the innovation process, which can lead to increased job satisfaction, customer loyalty, and brand awareness.

To mitigate the information disconnect between the seekers and the solvers, feedback between the parties plays an informative part. This in turn induces solvers to exercise more efforts in the contest. Feedback can increase the quality of the ideas generated in ideation contests (Chen et al., 2019). Feedback can also help solvers learn from their mistakes and improve their future performance. Seekers can provide feedback in various forms, such as comments, ratings, and rankings.

Li et al. (2021) found that the textual features of an idea description were significantly associated with the quality of the idea. Specifically, the use of concrete language, vivid imagery, and sensory details in the idea description was positively associated with idea quality. The study also found that the use of abstract language, jargon, and technical terms was negatively associated with idea quality. Thus, it is essential for seekers to provide clear and concise instructions to solvers and to encourage them to use concrete language and vivid imagery in their idea descriptions.

The University of Twente Entrepreneurial Challenge (UT Challenge) is a competition that offers students the opportunity to further develop their own, ingenious ideas, prototypes, and research projects in Minimal Viable Products and business plans. The challenge started in 2017 and is now held yearly.

The mission and main purpose of the UT Challenge is to challenge students to create innovative solutions to societal problems and translate them into real products, services, and business plans. The students have the opportunity to interact with and make use of the innovative and knowledge partners of the challenge. The partners offer coaching assistance to students, helping them with the development of their products, services, business plans, and start-ups. The challenge also offers students the opportunity to develop their personal skills.

The UT Challenge has grown into a platform that every student can benefit from. Hundreds of students have already developed ideas and prototypes for important societal issues, drawn up business plans and exchanged valuable knowledge with the businesses locally and elsewhere in the country.

The four best ideas get a prize money and get to compete in the Dutch 4TU Impact Challenge. The mission of the Dutch 4TU Impact Challenge is to offer the brightest minds of the four technical universities in the Netherlands a platform for entrepreneurship[1]. The four technical universities are the University of Twente, the University of Eindhoven, the Delft University of Technology, and Wageningen University & Research. Each university held a competition like the UT Challenge prior to the 4TU Impact Challenge.

1.1 Problem Statement

A large number of students submit their ideas to compete in the UT Challenge. However, not all ideas are created equally. It is important for the students to submit a well thought out and well described problem and solution, because experts and other people decide whether the idea is good enough to progress in the competition. The aim of this thesis is to find opportunities for improvement, by determining which key features are important in the evaluation process, for all the students that want to participate in the next UT Challenge. Machine learning is used to find these opportunities.

1.2 Research Question(s)

How can the idea quality be evaluated using a machine learning approach, based on participants' submissions in an innovation contest?

[1] https://4tuimpactchallenge.nl/

How does the quality of feedback influence idea quality, based on how the feedback source's expertise matches the idea topic?

1.3 Contributions

This thesis aims to offer a confirmation of studies, that are discussed in the following chapter 'Theoretical framework', regarding textual features and other features of the ideas in crowdsourcing competitions. This paper provides empirical evidence of the importance of a well written problem as well as a solution in the context of an innovation challenge, like the UT Challenge. The idea needs to cover all aspects of the challenge and solution. This paper also contributes to the theory with a model and a set of variables that are used in determination of high quality ideas. The model can automatically classify the ideas into successful and less or not successful categories based on the given variables.

The practical contributions this paper offers are a set of guidelines for students to take into account when forming a team and submitting their idea to the challenge. By identifying what characteristics are important in a well written challenge and solution, the students can focus on making sure they fulfil these characteristics as optimally as they can. This paper also offers the coaches a set of points they can focus on when coaching the students. The algorithms used can also identify where an idea is lacking, making it a more time efficient process to give feedback, as the points of feedback are automatically generated.

Theoretical framework

2.1 Related work

In this section, prior research on similar topics as this thesis are discussed. Following the discussion a hypothesis is formulated to tie in the prior research with this thesis.

Crowdsourcing and Ideation contests

Crowdsourcing is a method of obtaining ideas, content, or services from a large group of people, usually online. It is a way of outsourcing tasks to a large group of people, rather than to a single individual or organisation. Crowdsourcing can be used for varying purposes, including product development, market research, and content creation. Complex problems can also be solved using crowdsourcing, such as in scientific research, and to fund projects, such as films or music albums.

The American Psychological Association (APA) provides a guide to crowdsourcing for beginners, which describes how crowdsourcing can be used in a behavioural science context. A large and diverse group of people can be utilised for research opportunities (APA, 2016). Multiple top journals have published research utilising crowdsourcing according to the APA. However, in some disciplines, the unfamiliarity with crowdsourcing for both reviewers and readers is a roadblock in the adoption of the full use of crowdsourcing (Landers & Behrend, 2015).

Westland and Mallapragada (2011) provide guidance for future research and a review of literature on crowdsourcing. They note that crowdsourcing has several benefits, including the ability to tap into a large pool of talent, the capability to generate ideas quickly and inexpensively, and the ability to engage with customers and other stakeholders. However, they also note that there are several negatives to crowdsourcing, including the potential for low-quality work, the potential for exploitation of workers, and the potential for intellectual property disputes.

According to Schiavone, Appio, and Arreola-Risa (2021), crowdsourcing can be used as a tool for open innovation, and it can help firms to generate new ideas, reduce costs, and improve their competitive position. The authors provide a systematic literature review of crowdsourcing and open innovation, and propose an integrative framework for understanding

the relationship between these two concepts. The article also identifies several research gaps in the literature, and proposes several future research directions.

Ideation contests makes use of crowdsourcing. Ideation is the process of generation or conceiving ideas and concepts that may be useful for attaining some desired outcome (Conolly, Jessup, and Valacich, 1990). Ideation contests are a type of ideation techniques and a modern form of electronic brainstorming. Ultimately, the purpose of every ideation technique is to develop good, or outstanding ideas. In terms of ideation contests, good ideas are ideas that contain novel information, that are practical to implement, that would achieve the goal, and that would not create new unacceptable conditions. There is not one specific process or sequence of events to organise an ideation contest. An ideation contest is a form a crowdsourcing, where the initiator of the contest seeks for input from the crowd for a given task. The initiator rewards the best ideas based on a given set of requirements.

Gefen, D., Gefen, G., and Carmel (2015) have done research on how the project description length and expected duration affect bidding and project success in crowdsourcing software development. Projects that were described at greater length were presumably expected to be more carefully described than projects described at shorter lengths. Projects that were described too briefly could create misunderstandings and reduce the potential for success. Whereas projects described at greater length had a smaller chance of being misunderstood, and would therefore be more likely to obtain investments. Projects described at a greater length were presumably assumed to be larger projects. The findings suggest that there is a higher likelihood of success for larger projects compared to shorter duration projects.

Hypothesis 1. *A more descriptive challenge and solution will overall result in a more successful project.*

Ahmed and Fuge (2017) described how to identify or filter high quality ideas presented by participants collaborating in online communities. They used many features as an indicator of the quality of an idea, such as community feedback, author location, text descriptors, text readability, and more. They found that winning ideas used on average more long words than ideas in the evaluation stage and ideas in the initial stage. They also found that winning ideas on average were comprised of 70 sentences contrasted by only 60 sentences for ideas in the evaluation stage and only 26 sentences for ideas in the initial stage.

The vocabulary also differed between the winning ideas, 471 unique words, the ideas in the evaluation stage, 427 unique words, and the ideas in the initial stage, 215 unique words.

Hypothesis 2. *Submissions that use a large vocabulary size for the description of the challenge and solution will be more successful.*

Hoornaert, Ballings, Malthouse, and Van den Poel (2017) researched the effects the 3Cs have on predicting the likelihood of an idea being implemented. The 3Cs are content-based idea selection, contributor-based idea selection, and crowd-based idea selection. The content-based selection of an idea concerns with the description of the idea by the contributor. The dimension also includes whether any media was included. The researchers used text mining techniques to predict the likelihood of idea implementation. Two variables are derived based on the contents, the novelty of an idea and the degree of similarity an idea has compared to previously submitted ideas.

The contributor dimension takes into account any previous ideas the contributor has submitted. A notion is made that high quality ideas come from a person with existing knowledge and high expertise. A total of four variables are derived for the contributor-based selection. The number of previous comments by the contributor, the total sum of ideas and the sum of ideas that have been implemented submitted by the contributor, and the number of days a contributor was active before submitting an idea.

Hypothesis 3. *Submissions by higher educated participants will perform better than submissions by lower educated people.*

The crowd dimension refers to the contribution of the crowd in terms of votes, comments, ratings, rankings, etc., and how it affects the likelihood of an idea being implemented. The variables created for this dimension are the number votes by the crowd and the number comments by the crowd.

The researchers find that including crowd-based features improved the performance of the model compared to only including content-based and contributor-based features. The contributor-based features are of the lowest importance for the Random Forest model that was used for prediction. Crowd-based features was of the biggest importance. One interesting finding was that the relative distinctiveness of an idea would have to be at the extreme ends,

meaning that the idea has to be either very similar to previous ideas or not similar at all to previous ideas, to have a higher likelihood of being implemented.

Hypothesis 4. *Submissions that have pictures or videos perform better than submissions that do not contain any pictures or videos.*

Hypothesis 5. *Submissions that have a low similarity rate compared to other submissions will be more successful.*

A machine learning method was used by Rhyn and Blohm (2017) for classifying textual data in crowdsourcing. In their study, the principles of text mining and machine learning were built upon to automate the process of splitting higher quality contributions from the lower quality contributions in part. Their results showed it is possible to describe and predict the quality of contributions sourced from a crowd based on a set of textual features. The found that next to the length and uniqueness of the contribution, the readability of the contribution also played a role using a Random Forest model. The model achieved an overall accuracy of 80.03%. The specificity of the model was 87.73%, indicating that the model classifies low quality contributions exceptionally well. In contrast, the model achieves a 60.27% sensitivity, indicating that the model has a harder time correctly classifying high quality contributions.

Hypothesis 6. *The readability of the submission positively influences the success of the project.*

A study by Curral, Forrester, Dawson, and West (2001) described the relationship between team inputs, such as task type and team size, and team processes in 87 teams, with both low and high requirements for innovation. They found that larger teams have poorer performance in team processes. They also found that larger teams perform worse under a relatively high pressure to innovate. The same sentiment is found in researches by Frome (2019) and Tamvada (2011), who found that the optimal team size for innovation is between two to five members and three members, respectively.

Hypothesis 7. *Smaller teams (fewer than five people) will perform better than larger teams.*

An extensive amount of research has been done on the effects different types of feedback have on the quality of the ideas or the participation intensity of the participants.

Jiang, Huang, and Beil (2019) found that to achieve the highest quality ideas in a contest, both feedback in the earliest stages of the idea as well as feedback in the later stages are effective. In addition, they observed that feedback can help to guide the contributors in their exploration and exploitation decision-making, but can also discourage contributors from making follow-up actions. Thus, to have more high quality ideas overall, feedback in later stages is most effective. Camacho, Nam, Kannan, and Stremersch (2019) found that participation intensity increased when negative feedback was given during the early stages of a contest. Additionally, they found that negative feedback is overall better for participation intensity than positive feedback.

Wooten and Ulrich (2017) found that winning ideas are not affected by feedback in terms of quality. Not winning ideas are positively influenced by feedback. However, this feedback must (in)directly come from the people who decide the winners of the contest. General feedback does not significantly affect the quality of the ideas.

In the ideation contests by the Dutch universities, the participants can reach out to partnered coaches for feedback on their ideas. Based on the papers above the following is hypothesised:

Hypothesis 8. *More feedback, meaning more coaches helping the participants, will overall result in a more successful project.*

2.2 Background Information

The purpose of a theoretical framework is to give a foundation to the thesis. In this section web scraping, data mining, text mining, and natural language processing are explained more thoroughly.

4TU Federation[2]

The UT Entrepreneurial Challenge, together with the innovation contests held by the Universities of Eindhoven, Wageningen, and Delft, is part of the 4TU Impact initiative. The four universities of technology are working together to strengthen and pool technical knowledge aiming for the production of highly qualified engineers and technical designers, as

[2] https://www.4tu.nl/en/

well as the organisation of outstanding and socially relevant research of an international standard, and the promotion of collaboration between research institutes and businesses. This joint venture is called the 4TU Federation. The 4TU Federation combines the strengths in teaching, research, and knowledge valorisation of each technical university. The 4TU Impact plan contains the ambitions of the four technical universities in the area of knowledge transfer.

The 4TU Impact plan aims to stimulate and further expand the development of the knowledge economy. This is achieved through developing existing valorisation programs and cooperation projects. The reason for this plan is that the government wants to further strengthen its position as one of the global leaders in the knowledge economy and as an innovation country. The 4TU universities have proven in the past to be very successful in the field of knowledge generation and innovation by collaborating with SMEs as well as by valorising knowledge. The goal of the plan is to build on these previous successes and acting within the existing policies and visions laid out by the top sectors.

The 4TU Impact plan is implemented through a structured and modular approach. Through this, no uniformity is created, it allows other parties to join the 4TU consortium, and the joint knowledge base will be increased through collaborations. The first module is directed towards research in collaboration with businesses. Businesses, knowledge institutions, the government, and other public organisations finance these research collaborations. This collaboration takes place through long-term industry/university partnerships in which research projects are carried out and innovations are realised on the basis of jointly structured roadmaps. This is called the Impuls Model.

The second module is directed towards the collaborative development and implementation of innovative projects with a relatively short lead time to the market and is called the Living Lab Model. SMEs, students and researchers work together in self-managed teams on these innovative projects to develop innovations that deliver concrete services and products to the market.

The third module, business development and entrepreneurship, is organised from different pillars to bring research results to value for society as a whole. The creation of awareness among researchers, scouting, screening and building together of business

propositions that can grow into licences and/or the establishment of a start-up, the further development of entrepreneurial education for students is crucial in this.

The fourth module focuses on financing. Through a professionally guided financing process with financing funds that starts with Pre-Seed funding, the basis is laid for the market introduction. Funding is provided through the 4TU Seed Fund. The Seed Fund, 75 million Euros in size, will be involved in the Pre-Seed funding and will therefore bring resources and professional guidance to 4TU propositions at an early stage.

Machine learning

Machine learning is a part of the artificial intelligence domain to teach computers how to handle the data more efficiently through iteration without explicitly being programmed to do so. Machine learning algorithms can, in many cases, perform data handling more accurately and faster than humans. With more datasets becoming more accessible and available, the demand for machine learning is on the rise. Machine learning algorithms are often classified into supervised, unsupervised and reinforcement algorithms.

Supervised machine learning algorithms are algorithms that predict outcomes based on previously characterised input data. The dataset is divided into a training dataset and test dataset. The training dataset contains the target variable which needs to be predicted or classified. The supervised learning algorithm tries to learn the relationship between the input data and the target variable. The algorithms apply the learnt relationship to the test dataset for prediction or classification.

Unsupervised machine learning algorithms are algorithms are different from supervised learning in that there is no defined output variable. Unsupervised learning algorithms is used to find unknown structures or relationships in a dataset. It is generally applied for clustering purposes and feature reduction.

Reinforcement learning uses trial and error to decide which actions to perform, such that the most optimal outcome is achieved. Reinforcement learning is similar to unsupervised learning as there is no known outcome variable. However, it differs from unsupervised learning in that it uses trial and error instead of structures or relationships in the dataset.

Learning algorithms can also be used in conjunction. When different learners are joined to create one learner it is called ensemble learning. Research finds that combining

multiple learning algorithms almost always perform better than individual learning algorithms. Two of the most used learning techniques are given below:

1) Bagging[3]: Bagging is applied to improve the accuracy and stability of a machine learning model. The method creates multiple bootstrapped subsamples of the dataset and applies the machine learning algorithm to the subsamples. An algorithm then aggregates the result from each subsample to find the most efficient predictor. Bagging is applicable in both classification and regression purposes. Bagging decreases variance and helps to prevent overfitting.
2) Boosting[3]: Boosting is applied to decrease bias and variance. Boosting builds on a weak learner and sequentially tries to improve the learner where the highest misclassification is made. This results in a stronger learner. The difference between a weak and a strong learner is that a weak learner makes classifications that are weakly correlated with the true classification, whereas a strong learner makes classifications that are strongly correlated with the true classification.

In this research, the data used include the outcome variable. Therefore, supervised machine learning techniques are applied. The following describes common supervised machine learning techniques.

Decision tree

A decision tree model iteratively splits the dataset on the feature that splits the data as well as possible into the various different classes until a specific stop condition is met. A decision tree consists of a root node, inner nodes, and end nodes which are also known as leaves. The root node consists of the feature that best separates the classes. The inner nodes try to further separate the data into the respective classes based on a different feature. Each inner node has one incoming branch and two or more outgoing branches. The leaves are the nodes where the data separation stops. Leaves are created when perfect classification is performed or when the model cannot find a significant difference between the data. A decision tree can be visualised in a format that resembles a tree, which can easily be interpreted by humans. The visualisation shows which rules are applied at each node.

Random forest

Random forest is a model that is based on the bagged decision trees. Opposed to a decision tree, which is a singular tree, random forest is an ensemble of trees (Nagpal, 2017).

[3] https://medium.com/fintechexplained/bagging-vs-boosting-in-machine-learning-8d7512d782e0

The bagged tree model randomly selects several subsets of the training dataset with replacement, bootstrapping[4]. Random forest works similarly to bagged tree, but it also takes a random selection of features instead of all the features to create trees (Nagpal, 2017). A decision tree is trained on each subset of the training dataset, this leads to an ensemble of trees (Nagpal, 2017). The average of all the predictions by the ensemble of decision trees is then taken.

Naïve Bayes

The Naive Bayesian classification model is a probabilistic classifier based on the Bayes Theorem. The Naïve Bayes classifier operates under the assumption that each pair of features in the dataset are independent of each other, given the dependent variable[5]. Another assumption that is made is that all the independent features carry an equal weight on the outcome. The Bayes Theorem is as follows:

$$\mathrm{P(A|B)} = \frac{P(B|A)*P(A)}{P(B)}, \text{ or } \mathrm{P(A}|B_1, \dots, B_n) = \frac{P(B_1|A)*P(B_2|A)\dots P(B_n|A)*P(A)}{P(B_1)*P(B_2)\dots P(B_n)}$$

A in the Bayes Theorem represents the dependent variable and B_i represents the independent variables. Thus, the probability of dependent variable *A* happening given independent variables *B* occurred.

Linear regression[6]

A linear regression model tries to fit a linear equation to the dataset. The dependent target variables needs to be continuous. The relationship between the dependent target variable and the independent variables in the model needs to be linear. The formula for the linear regression model is as follows:

$$z = a + \sum \beta_i X_i$$

In this formula, z represents the dependent variable, α represents the intercept, and β represents the slope, and *X* represents an independent variable.

Logistic regression[6]

A logistic regression model is similar to a linear regression model. The distinction between the linear and logistic models is that in a logistic regression model the logistic

[4] https://en.wikipedia.org/wiki/Bootstrapping_(statistics)
[5] https://scikit-learn.org/stable/modules/naive_bayes.html
[6] https://scikit-learn.org/stable/modules/linear_model.html

function is used to perform a binary classification task. The logistic function in binary cases is as follows:

$$y = \frac{1}{1 + e^{-x}}$$

An ordinal logistic regression model differs from the standard logistic regression model. This model is a regression model for ordinal dependent variables. The proportional odds model forms the basis for the ordinal logistic regression model. The formula for the proportional odds model is as follows:

$$\text{logit}[P(Y \leq j)] = \alpha_j - \sum \beta_i X_i$$

where $j = 1, \ldots, J-1$, and $i = 1, \ldots, M$

In the formula J represents the total number of categories in the ordinal dependent variable, M represents the number of variables, α represents the intercept, β represents the slope, and *X* represents an independent variable. For example, the order of the dependent variable is small -> medium -> large. The equation $\text{logit}[P(Y \leq 1)]$ is interpreted as the odds of getting ranked small versus medium and large. Similarly, the equation $\text{logit}[P(Y \leq 2)]$ is interpreted as the odds of getting ranked small or medium versus large.

Neural network[7]

An artificial neural network has the capability to learn from examples. Artificial neural networks are information processing models inspired by biological neural systems. It consists of a large number of closely coupled processing elements called neurons. It processes information in parallel at all nodes, following a non-linear path. Neural networks are complex adaptive systems. Adaptive means that the internal structure can be changed by adjusting the weights of the input data. A neural network is comprised of an input layer, one or multiple hidden layers, and an output layers. The input layer represents all the information, the independent variables, that is to be fed into the network. The output layer gives the ultimate result. The hidden layer connects the input and output layers through neurons.

[7] https://www.ibm.com/topics/neural-networks

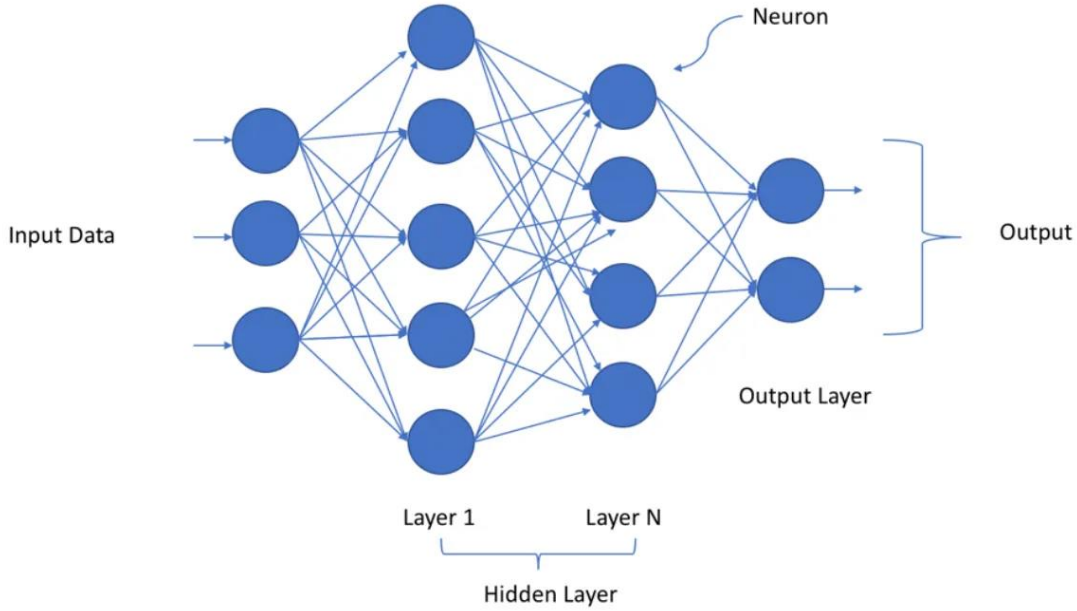


Figure 1: A neural network with three hidden layers (source: https://towardsdatascience.com/a-laymans-guide-to-deep-neural-networks-ddcea24847fb)

Instance-Based Learning (kNN)[8]

An instance-based learning model does classification based on similarity with previously found data points. The k-nearest neighbours model looks at the *k* most similar data points and their classes, it subsequentially assigns the new data point to a class based on the most common class. This model requires a function that calculates the distance to determine the similarity between two data points. A commonly used distance function is the Euclidean distance.

Support Vector Machines[9]

A support vector machine classifies data by trying to find the best way to distinguish between two or more groups of data points. The support vector machine approach tries to create a function that splits the data into classes with the largest possible margin, meaning that the difference between the classes is maximised. Similarly to instance-based learning, a distance function is required to measure the distance between data points, also known as vectors. It is also possible to plot multiple functions to separate data points in case the dataset consists of multiple classes.

[8] https://scikit-learn.org/stable/modules/neighbors.html
[9] https://www.ibm.com/docs/en/spss-modeler/saas?topic=models-how-svm-works

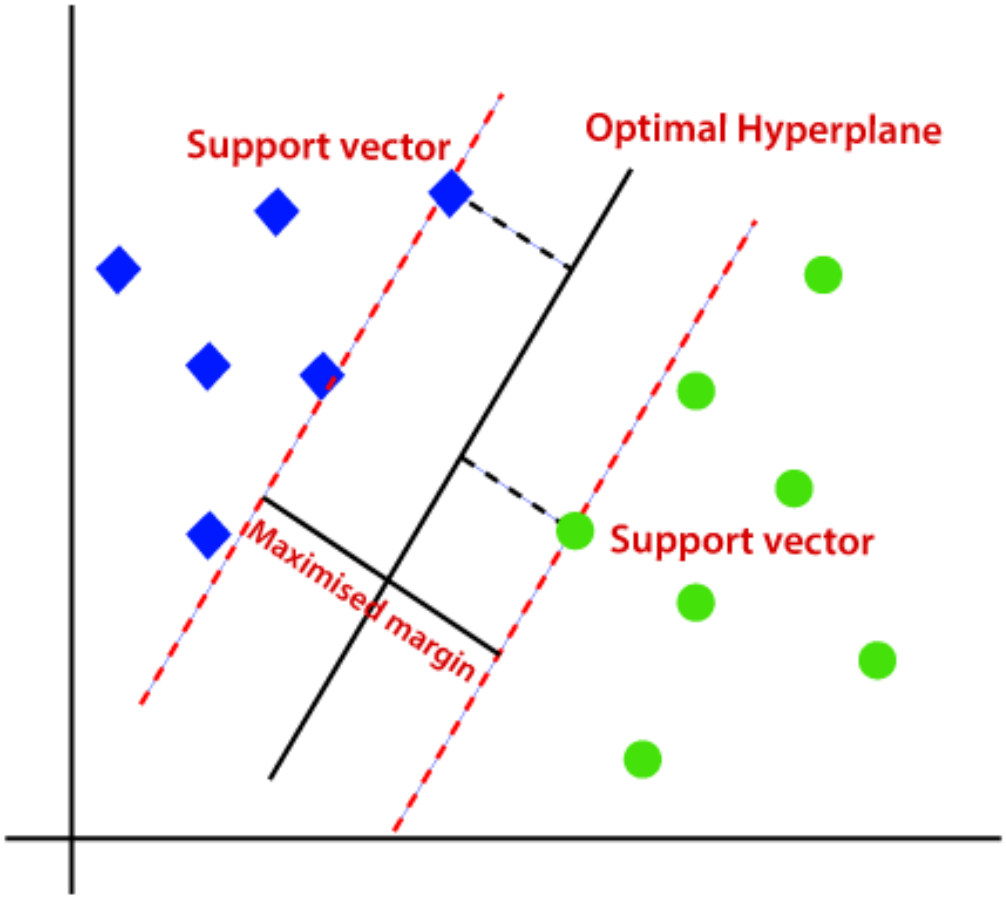


Figure 2: A graphical representation of a support vector machine _source: https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/)

Evaluation of machine learning models

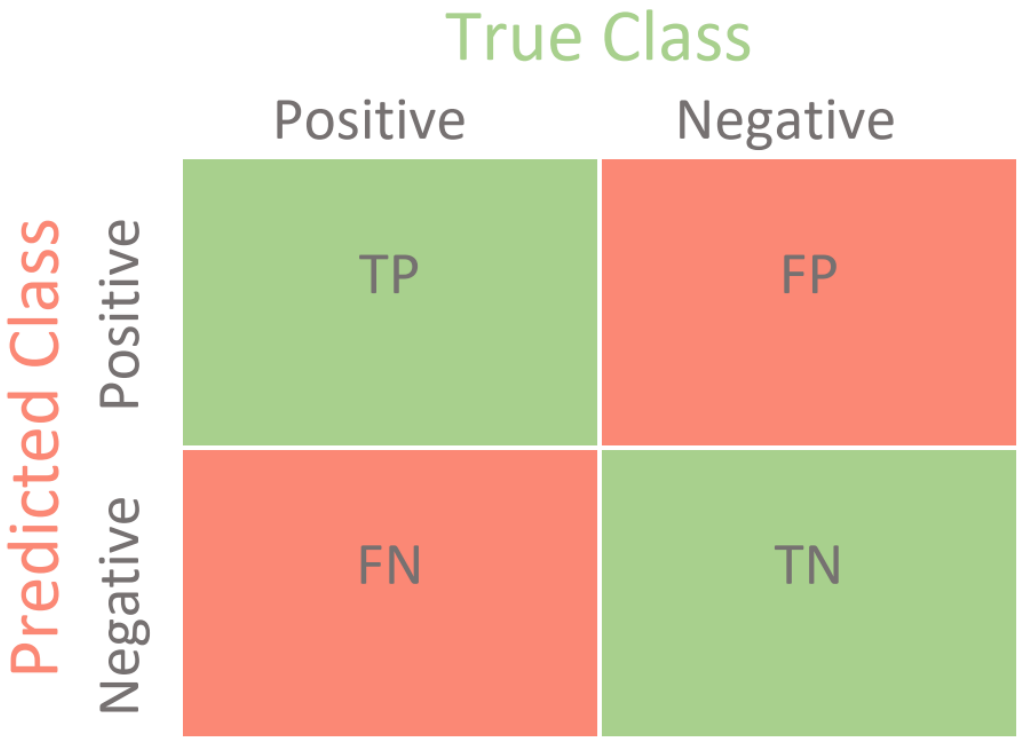
Evaluating the results of machine learning models is an essential part when using machine learning. The results are used by the machine learning models as feedback and to determine whether any changes in following iterations of the models are improvements or not. The results are ultimately used to verify how well the model predicts outcomes based on the given data. Evaluation of classifying machine learning models is generally done by separating the dataset into two parts, a training dataset and a test dataset. The machine learning model is trained using the train dataset. The evaluation of the machine learning model is done by testing the trained model on the test dataset and calculating its performance metrics.

A common problem seen in machine learning is overfitting. Overfitting happens when the machine learning model is trained too well on the training dataset[10]. This leads to the trained model being incapable of generalising appropriately to unseen data and therefore having worse performance metrics. The opposite of overfitting is underfitting. Underfitting

[10] https://www.baeldung.com/cs/ml-underfitting-overfitting

happens when the machine learning model cannot find relationships between the features and the output variable. This means that the model could not properly predict or classify the training dataset, and consequentially will not properly predict or classify the test dataset.

A common approach to overfitting is to use resampling methods. K-fold cross-validation and bootstrapping are the two most common resampling methods. K-fold cross-validation is the procedure of splitting the dataset into K parts. All parts except for one are then merged back together to form the training dataset, where the left-out part forms the test dataset. Each part will subsequentially be used as the test dataset, where the rest form the training dataset[11]. The performance metrics are obtained by taking the mean of all validation processes. Bootstrapping is the process of creating multiple new training datasets by random sampling of data from the whole dataset. For each new training dataset, the data that are not sampled form the test dataset. Similar to cross-validation, the mean of the performance metrics of the trained models is taken. The confusion matrix is a table that visualises the performance of a classifying machine learning model. In figure 3 below, the rows in the confusion matrix represent the predicted classes, while the columns of the matrix represent the true classes.



[11] https://machinelearningmastery.com/k-fold-cross-validation/

Figure 3: A confusion matrix (source: https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826)

In figure 3 an example of a confusion matrix is shown. The example shows two possible classes, positive and negative. In case a machine learning model predicts the positive class and the actual class is also positive, then the prediction is a true positive (TP). When the predicted class and the actual class do not match, where the predicted class is positive and the actual class is negative, then the prediction is a false positive (FP). Similarly, if the class of the prediction is negative and the actual class is positive, then the prediction is a false negative (FN). When both the predicted class is negative and actual class is also negative, then the prediction is a true negative (TN). A confusion matrix can be extended beyond two classes. The following performance metrics can be determined using the confusion matrix:

- Accuracy: The accuracy of a model equates to number of correctly predicted cases compared to all the cases in the sample. The main problem with accuracy is that the results can be heavily skewed due to an imbalanced dataset. For example, a dataset contains 100 cases and 95 cases have the positive class and five cases have the negative class. A machine learning model can predict all 100 cases to be positive, achieving a 95% accuracy rate. This model would completely misclassify a whole class, but still achieve a high accuracy rate. The formula for accuracy is:

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

- Benchmarking: Benchmarking is the procedure of judging the value of the performance metric to a baseline value, to reinforce its statement. In classification, a common reference value is the no information rate. The no information rate calculates the accuracy when the whole dataset is classified to the majority class. The example given in the description of accuracy has an accuracy of 95%. However, the no information rate would also be equal to 95%. Benchmarking gives more context to how well a machine learning model performs.
- Precision[12]: The precision metric is also known as the positive prediction value, the metric measures the instances that are correctly classified as true instances among all predicted instances classified as positive. The precision metric concerns the Type-I

[12] https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

errors. A Type-I error occurs when a true null hypothesis is rejected. The precision metric is calculated for each class. The formula for precision is:

$$\text{Precision} = \frac{TP}{TP + FP} \; and/or \; \frac{TN}{TN + FN}$$

- Recall[12]: The recall metric is also known as the sensitivity for the positive class and the specificity for the negative class. The recall metric evaluates the fraction of correctly classified instances as true among all predicted true instances The recall metric concerns the type-II errors. A type-II error occurs when we accept a false null hypothesis. The recall metric is calculated for each class. The formula for recall is:

$$\text{Recall} = \frac{TP}{TP + FN} \; and/or \; \frac{TN}{TN + FP}$$

- F1-score[12]: The F1-score metric combines the precision and recall metrics by utilising the harmonic mean between the two metrics. The F1-score is most effective when both the precision and the recall metrics of the model must be important. A low F1-score does not say where the weakness of the model lies, in either type-I or type-II errors. The formula for the F1-score is:

$$\text{F1} = 2 * \frac{Precision * Recall}{Precision + Recall}$$

- AUC-ROC curve[13]: AUC-ROC is the abbreviation for Area Under Curve-Receiver Operating Characteristic. The AUC-ROC curve is a graphical representation of the performance of a model on varying threshold values. The x-axis of the graphical representation shows the false positive rate plotted, while the y-axis shows the true positive rate plotted. The AUC-ROC curve metric shows the capability of the machine learning model to distinguish the classes. A higher AUC means that the model can distinguish the classes better than when the AUC is lower.

Web scraping

Web scraping is the extraction of structured information from semi-structured data, it is a form of data mining A web scraper can obtain data of interest such as text, media, or hyperlinks from a web page (Chow, 2012). Web scrapers perform the same tasks as a human would when accessing a web server to obtain the desired data from said server. However, it does this in a much shorter time, especially when multiple web pages need to be scraped. A web scraper accesses a web page and searches the source code, or the HTML code, of the

[13] https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

web page using regular expressions (Schnell & Redlich, 2019). When the specific information the web scraper was searching for is found, the information is extracted and copied to an external output file. The acquired information typically needs to be cleaned and checked after mining before it can be used in other processes. A web scraper come in the form of a library, a framework or as a desktop-based environment. A desktop-based environment simplifies the process of web scraping by enabling it without prior knowledge of programming languages (Glez-Pena, Lourenco, Lopez-Fernandez, Reboiro- Jato, & Fdez-Riverola., 2013).

There are advantages and disadvantages to using a web scraper. The advantages of web scraping are that it is time efficient, more extensive, the option of regular data collection within shorter periods of time, and the cut of costs compared to manual extraction. (Hoekstra, ten Bosch, & Harteveld, 2010). The most time consuming part of web scraping is creating the scraper. However, when the scraper is built, an incredible amount of data can be obtained within a much shorter time than when it is obtained manually.

The disadvantages of web scraping are the requirement for the skills to write a web scraper when no helper programmes are available, websites changing their codebase, absence of automatic checks when extracting data, and ethical and legal issues (Hoekstra et al., 2010). When different web pages need to be scraped, the web scraper also needs to be adapted. There is no one web scraper that can obtain the same things, such as page title or page contents, from different web pages. There is also the possibility that the web scraper tries to access websites too many times which could overload the servers. This may result in a raise in expenses for the web page owner, as more bandwidth is required. This objection may be trivial, considering the increased bandwidth and capacity of web servers. However, ethical and legal issues can still arise when obtaining data unauthorised. Web scrapers might also be blocked from scraping certain web pages, which makes scraping not possible.

Empirical approach

3.1 Data collection

To test the research model, data is collected from four different open innovation contests. The data is taken from the four technical universities, who participated in the 4TU impact challenge from the period 2018-2020. Each university has its own website, where it hosts all the data of the participants of the innovation challenge. Students can submit their ideas to the contest in three different categories; ideation, prototype, and start-up. They propose a solution to a societal challenge. Through their submission, the students get access to online coaching by reaching out to partnered organisations, as well as workshops, training, and speed dates to improve on their proposed innovative solution.

A few assumptions are made regarding the submissions from the different universities and what success in the contests means:

- All submissions are held to the same standard
- Higher quality ideas are more successful in the contests
- There are no significant differences between how the contests at different universities and between different years are ranked
- All submissions are independently ranked

3.2 Sample selection

The sample choice is based on relevance and completeness. The sample that is chosen contains the data from the UT Challenge 2019, TU Delft contest 2019 and 2020, and TU Eindhoven contest 2019. The reason for choosing these specific contests is that these four contests have ranked the submissions in a similar fashion. There are finalists, submissions that ranked in the top 40, and the submissions that did not make it into the top 40. Other contest years had either a different ranking system. At the point of writing this paper, this sample is the latest and most complete data. The sample contains 251 entries. Each entry describes the challenge which is tackled and the solution on how the challenge is tackled. It also provides information on which coaches helped the entry, and how far they reached in the competition. Other information that was given are the team size, team composition, names, and nationalities. The sample data is scraped from the respective websites using Python and the BeautifulSoup4 package.

Python, a programming language, is used to scrape the data from the submissions' web pages. The HTML codes of the web pages are retrieved using the requests package in Python. The requests package allows Python to access and download web pages. The HTML codes are then parsed using the BeatifulSoup4 package in python. The information obtained from the parsing is written to a csv file. Each row in the csv file represents an idea submission. Each row contains the project title, the contest the project entered in, the names of the contestants, the educational level of the contestants, the category in which the project entered, the coaches of the project, the description of the challenge, the description of the solution to the challenge, the presence of media (images and/or videos).

3.3 Data selection

The first step of the data mining process concerns selecting or segmenting the data that are relevant to the research. A dataset does not always only contain information that is relevant to the user. For example, when a researcher is interested in the growth over time of a certain group, then their educational level might not be relevant to the research and are, therefore, removed from the dataset.

In many cases, the nationality of the participants is not known. Due to this limitation and the fact that the relevancy of this characteristic to the research is dubious, the choice is made to not include this data in the research. Another characteristic that is left out is the category in which the submission is submitted. The distribution of the categories is too unequal, start-ups 10%, prototypes 45%, and ideation 45%. This skewed distribution could affect the reliability of the results negatively and will therefore be omitted. The year of participation as well as the names of the participants is not relevant to the research and are, therefore, also omitted from the dataset. The names of the participants is, however, transformed into useable data, which will be addressed in the transformation section.

Of the 251 total entries, three entries do not contain any information. These entries were either used for testing purposes of the website or they were used as placeholders. Therefore, these three entries are removed from the dataset. This leads to a total dataset of 248 entries.

The TU Delft contest 2020 has a major difference in number of submissions that have coaches compared to the other contests. The data of the TU Delft contest 2020 will, therefore, only be used when the variables related to the coaches are not used. This way, it is possible to

maximise the usage of the limited dataset that is available. This leads to two working datasets, one dataset with 248 observations, and one dataset with 164 observations.

Table 1: Percentage of submissions without coaches per contest

	UT Challenge 2019	TU Eindhoven contest 2019	TU Delft contest 2019	TU Delft contest 2020
number of submissions without coaches	6	5	18	59
Finalist	0	0	5	4
Top 40	3	0	3	14
Participant	3	5	10	41
Total number of submissions	49	61	54	84
percentage without any coaches	12%	8%	33%	70%

3.4 Pre-processing and transformation

The second step of the data mining process is concerned with the cleaning of the data after selection. Data cleaning is the process of simplifying the data and removing garbage. The pre-processing is very important, because, for example, outliers within the dataset can heavily skew the results when actual data mining takes place. Depending on whether it is necessary, noisy data and outliers are removed. The third step in the process is transformation. In this step the data is made usable and navigable. The data is transformed in order for data mining to be applicable.

In many cases the educational level of the participants is formatted differently from the other cases that have the same meaning. The cases in which the formatting is different are manually changed so that all the values with the same meaning have the same value. The data with missing values are not removed from the dataset, because they can still provide valuable information or a missing value can still have meaning. Some pages also provided the same participant's name twice, in those cases, it is assumed that those two entries are meant to be one entry. Therefore, the duplicate is removed.

Team size

The names of the participants do not provide useful information, but they are used to create a new feature, the number of participants in one group. This is simply done by counting the number of names.

Education level

There are three education levels: Bachelor, Master, and PHD. Each submission is sent in by one person or a group of people with one of the three education levels. In this research the submissions are categorised into two groups. The first group contains the submissions where all the submitters are exclusively doing a Bachelor's programme. The second group contains all the submissions where the submitters have at least one person in the group that does a Master's or PHD programme. The number of PHD students is significantly lower than the other two, and therefore is grouped with the Master students.

Presence of media

The scraped data contains information on whether the submission contained an image or a video. This data is combined into the variable "Media". Submissions either have media, represented by a 1, or submissions do not have media, represented by a 0.

Simple text analysis

The description of the challenge and solution are also transformed. To reduce the number of variables, the title, the description of the challenge, and the description of the solution are combined into one variable called "Description". For the sake of consistency and to reduce feature dimensions, the description variable is spellchecked using Microsoft Word's spell checker and normalised by converting it into GB English. The description can then be further transformed into the length of the description, which is measured by counting the total number of words. This process creates a new variable with an integer value representing the number of words. The number of sentences is also counted in a similar way and turned into a variable. The previously mentioned processes are automatically done using the *readability* Python package.

In-depth text analysis

More in depth transformation is done on the description. For the number of unique words, the words need to lemmatised. Before lemmatisation, the text needs to be tokenised.

Tokenisation is the process of splitting the entire text into single words. Every word in the text is a token. After tokenisation, the tokens are lemmatised. Lemmatisation is the procedure of combining together the inflected forms of words so they can be studied as a single element, which is identifiable by the word's lemma.[14] The lemmatisation process involves another process called part-of-speech tagging. Part-of-speech tagging is the practice of assigning a word in a corpus matching to a part of speech.[15] The possible tags a word can have in this research are noun, verb, adjective, or adverb. The tags give the lemmatisation process context of the word, so words are properly lemmatised. After lemmatisation, stop words and other redundant words are removed. Stop words are words that do not add meaning to a sentence and are, therefore, considered to be noise. Tokenisation, stop words removal, lemmatisation, and part-of-speech tagging are done using the *spaCy* package for Python. It is then possible to count the number of unique words used by presenting all tokens in a list and counting all unique tokens in the list. Longer descriptions are more likely to have more unique words. Therefore, the number of unique words is normalised to the total number of words in the description.

Readability

The readability of the descriptions is measured by performing the Flesch reading-ease test and the Dale-Chall readability formula using the previously mentioned "readability" package in Python. The tests is designed to indicate the difficulty to understand a passage in English. In contrast to Rhyn and Blohm (2017), who use the Coleman-Liau index, this research uses the Flesch reading-ease score as there is no minimum requirement for number of words. The Flesch reading-ease score is calculated using the following formula:

$$Flesch\ reading\ ease\ score = 206.835 - 1.015\ \frac{total\ words}{total\ sentences} - 84.6\ \frac{total\ syllables}{total\ words}$$

Table 2 shows how the scores can be interpreted.

[14] *Collins English Dictionary*, entry for "lemmatise"
[15] "POS tags". *Sketch Engine*. Lexical Computing

Table 2: Flesch reading-ease scoring table (source: https://en.wikipedia.org/wiki/Flesch%E2%80%93Kincaid_readability_tests)

Score	School level (US)	Notes
100.00–90.00	5th grade	Very easy to read. Easily understood by an average 11-year-old student.
90.0–80.0	6th grade	Easy to read. Conversational English for consumers.
80.0–70.0	7th grade	Fairly easy to read.
70.0–60.0	8th & 9th grade	Plain English. Easily understood by 13- to 15-year-old students.
60.0–50.0	10th to 12th grade	Fairly difficult to read.
50.0–30.0	College	Difficult to read.
30.0–10.0	College graduate	Very difficult to read. Best understood by university graduates.
10.0–0.0	Professional	Extremely difficult to read. Best understood by university graduates.

Another readability test is the Dale–Chall readability formula. It measures the comprehension difficulty for readers of a text and provides a numeric representation of said difficulty. A list of 3000 common words that fourth-grade American students can understand is used. Words outside this list are considered to be difficult. The formula for the readability test is as follows:

$$Dale - Chall = 0.159 * 100 \frac{difficult\ words}{words} + 0.0496 \frac{words}{sentences}$$

Table 3 shows how the scores can be interpreted.

Table 3: Dale-Chall scoring table (source: https://en.wikipedia.org/wiki/Dale%E2%80%93Chall_readability_formula)

Score	Notes
4.9 or lower	easily understood by an average 4th-grade student or lower
5.0–5.9	easily understood by an average 5th- or 6th-grade student
6.0–6.9	easily understood by an average 7th- or 8th-grade student
7.0–7.9	easily understood by an average 9th- or 10th-grade student
8.0–8.9	easily understood by an average 11th- or 12th-grade student
9.0–9.9	easily understood by an average college student

Submission and Top Consortia for knowledge and innovation (TKI's)

A separate dataset is used for the coaches of the submissions. Each coach specialises in one or several clusters. The possible clusters a coach can specialise in are High Tech Systems & Materials, Chemics & Materials, Digital & Internet, Finance & Technology, Energy & Sustainability, Transport & Automotive, Buildings & Physics, High Tech to Feed the World, Nature 2.0, and Life Sciences & Health. The contests do not specify or describe

the clusters. Therefore, in this research the clusters are described by the innovation agendas and missions of the TKI or parties that are a part of the TKI (see appendix A).

The coaches of each submission specialise in one or more clusters. For each submission, dummy variables are created for these clusters. Dummy variables are binary variables, where a 1 represents the presence of the variable and a 0 represents the absence of the variable. For example, if submission X has coaches Y and Z who specialise in Nature 2.0, Digital & Internet, and Buildings & Physics, then the value of the dummy variables for Nature 2.0, Digital & Internet, and Buildings & Physics is 1. The value of all the other dummy variables is then 0, indicating the absence of these clusters. This information can then be used to determine what fraction of the clusters is covered by the coaches. This fraction is stored in another variable.

Submission similarity

To find how similar a submission is compared to the other submissions in the contest, the cosine similarity of the TF-IDF scores between the submission and the other submissions is calculated. A similar approach is applied by Walter and Back (2013), who use text mining techniques such as TF-IDF to gauge the quality of submissions to crowdsourcing contests.

TF-IDF stands for Term Frequency – Inverse Document Frequency, it is a statistical measure that reflects the relative importance a word has in a document compared to all the documents from the corpus. The corpus is the complete collection of documents that are to be analysed or that are used in the analysis. This is achieved by examining how many times a word occurs in a document while also taking into account how many times the same word also occurs in other documents in the corpus[16]. The formula for TF-IDF is as follows:

$$TF - IDF = \log(1 + f(w,d)) * \log\left(\frac{N}{f(w,D)}\right)$$

Where *f(w,d)* is the frequency of word *w* in document *d.*

Where *N* is the number of documents in the corpus.

Where *f(w,D)* is the frequency of word *w* in the corpus.

[16] https://towardsdatascience.com/tf-term-frequency-idf-inverse-document-frequency-from-scratch-in-python-6c2b61b78558

Cosine similarity is a similarity metric that measures the similarity among two or more vectors of an inner product space. It is based on the cosine of the angle between vectors and indicates whether the vectors are pointing in approximately the same direction. By performing TF-IDF first, the vectors are weighted. The formula for cosine similarity is as follows:

$$\cos(\theta) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Where A_i and B_i are components of vector A and B, respectively.

The TF-IDF scores calculations as well as the cosine similarity calculations are done using the *sklearn* Python package. This package also has the option to calculate the TF-IDF for *n-gram* words, meaning that instead of measuring the importance of one word, it measures the importance of a specific sequence of *n* words. In this research unigram, a one word sequence, and bigram, a two word sequence, are used. It is important to note that all text in the corpus are processed. This entails that tokenisation, stop word removal, lemmatisation, and part-of-speech tagging have taken place. When the highest cosine similarity is high, it can be concluded that the submission is very similar to another submission in the contest. However, when the highest cosine is low, it can be concluded that the submission is not very similar to any other submission in the contest. The highest cosine similarity value can, thus, be used to determine the dissimilarity of the idea compared to all the other ideas within the same competition.

Similarity between the submission and coach specialisations using coding

Data analysis can be approached in different ways depending on the data. In case of textual, qualitative data, the data needs to be coded. Coding is done deductively and/or inductively. The grounded theory approach is an inductive method which generates codes and theories solely based on the collected data (Blair, 2015). The start list approach is a deductive coding technique that derives codes from the theoretical framework and other sources relevant to the research (Basit, 2003). Blair (2015) mentions that it can be hard to separate inductive and deductive coding. The purpose of coding is to identify themes within the text with the intent to categorise them.

In this research a combination of deductive and inductive coding is used to analyse the data. The central themes in which the codes are categorised are based on the ten clusters in which the submissions could be placed in according to the UT Challenge. As previously mentioned, in this research the clusters are described by the innovation agendas and missions of the TKI or parties that are a part of the TKI. Submissions that do not contain any codes that fit a certain theme are put into the "other" category. The deductively created codes are created using the description of the previously mentioned clusters. The inductively created codes emerged through the analysis of the data. Due to large overlaps between the 10 clusters, certain phrases in the data can refer to multiple codes and can, therefore, be categorised in multiple categories. For example, the production of thin coatings and films is a collaborative programme between the high tech systems & materials sector and the chemicals & materials sector.

The data from coding is then used quantitatively. Through the coding, it is now known to which clusters each submission belongs. It is thus possible to determine the overlap between the clusters of each idea and its coaches. To do this, the Jaccard index is used. The Jaccard index is the proportion of the size of the intersection of the sample sets to the size of the union of the sample sets[17]. The Jaccard index is used to measure the similarity between the submission and the coaches that helped said submission based on the clusters. Yu and Miao (2021) use the Jaccard index in a similar way. The study uses the Jaccard index to match a list of projects, using the description of said project, with a related subgroup of keywords. The goal is to match projects and researchers in a similar way this research aims to investigate the effect a Jaccard similarity has between a submission and the coaches. The Jaccard index is calculated for each coach and then averaged in this research. Yu and Miao (2021) found that the Jaccard similarity indeed helped to select relevant projects. The formula for the Jaccard index is as follows:

$$\mathrm{J(A,B)} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Through the feature creation process, the following list of variables is created.

[17] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.jaccard_score.html

Table 4: Variable operationalisation

Variable	Definition of variable	Type and scale
Input Variables		
Media	The availability of a picture and/or a video on the submission page measured on a binary scale	Binary 1 media present 0 media not present
Highest education level	The education level of one of the members with the highest currently following education level of the group that have made a submission measured on a binary scale	Binary 1 Bachelor 0 Master/PHD
Team size	The size of the team that made a submission to the contest measured by the number of people in the team	Numeric ≥ 0
Number of coaches	The number of coaches that assisted a team measured by the number of coaches	Numeric ≥ 0
Jaccard index	The similarity coefficient between the coaches' fields of expertise and the submission's clusters based on the Jaccard index	Numeric 0 - 1
Total word count	The total number of words used in the title, problem, and solution combined	Numeric > 0
Total sentences	The total number of sentences used in the title, problem, and solution combined	Numeric > 0
Long words normalised	The ratio of the number of long words compared to the total number of words in the title, problem, and the solution	Numeric 0 - 1
Complex words normalised	The ratio of the number of complex words compared to the total number of words in the title, problem, and the solution	Numeric 0 - 1
Flesch reading-ease score	The readability of the problem and solution measured by the Flesh reading-ease score	Numeric 100 - 0
Dale-Chall score	The readability of the problem and solution measured by the Dale-Chall score	Numeric 0 - 10
Highest similarity	The similarity between the target submission and the most similar submission within the same contest measured by the cosine similarity using TF-IDF	Numeric 0 - 1
Coach coverage	The ratio of the combined fields of expertise of the coaches of a submission compared to the total number of clusters	Numeric 0 - 1
Output Variable		
Rank	The rank of the submission within its respective contest measured by an ordinal scale consisting of 3 categories	Ordinal 1 Participant 2 Top 40 3 Finalist

This paper uses two datasets as previously described, one dataset containing all 248 submissions, and one dataset containing 164 submissions, where the dataset with 248 observations omit the variables that have a relation to the coaches. The dataset with 248 submissions, thus, omit the variables coach coverage, Jaccard index, and number of coaches, whereas the dataset with 164 submissions includes all variables. The distribution of classes of both datasets are depicted in figure 4.

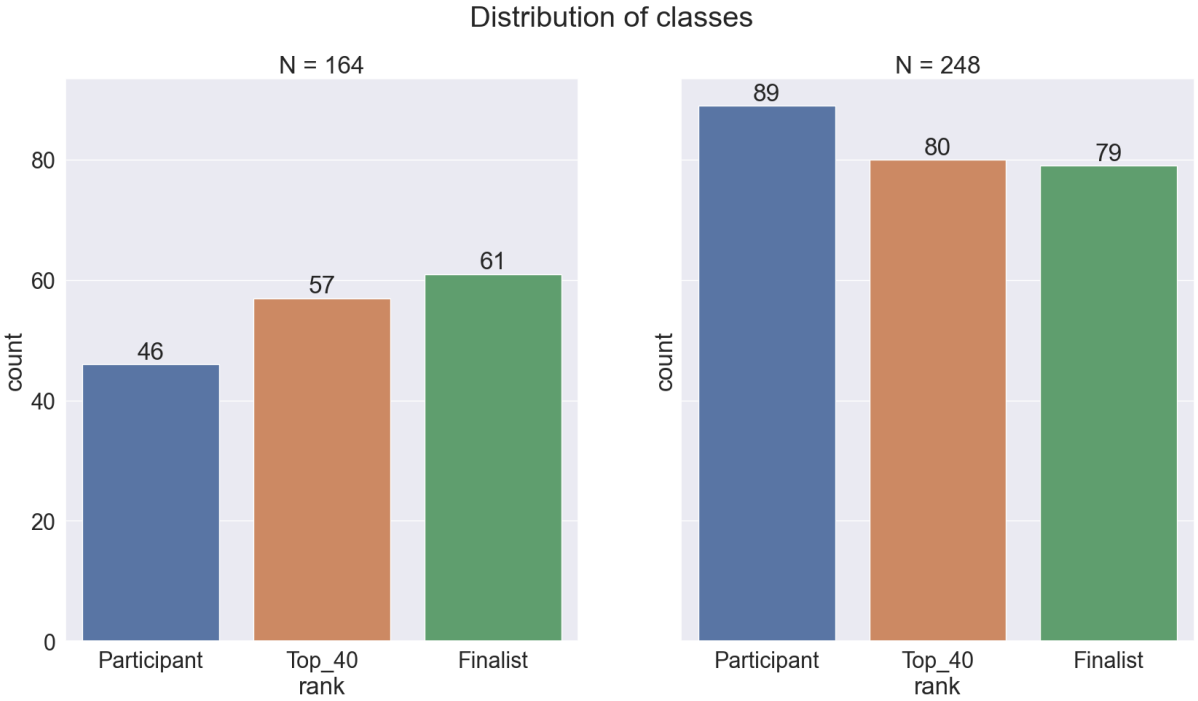


Figure 4: Class distribution for both datasets

3.5 Machine learning models

The scikit-learn library offers a variety of supervised learning algorithms via the programming language Python. In this research the following machine learning models are used to compare and find the most accurate model for the given datasets. To improve the machine learning models, hyper-parameters of an estimator are passed into the model and tuned to the datasets. This should improve the overall performance of the models[18].

RandomForestClassifier[19] is the machine learning model that implements the random forest method. A random forest is a model based on bagged decision trees. The hyper-parameters that are tuned for this model are n_estimators, which determines the number of trees in the model, max_depth, which determines the depth of the trees, criterion, which

[18] https://scikit-learn.org/stable/modules/grid_search.html
[19] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

determines the function to calculate the quality of the split, and class_weight, which determines the weights of each of the classes.

GradientBoostingClassifier[20] is the machine learning model that imlpements boosted decision trees. The hyper-parameters that are tuned for this model are n_estimators, max_depth, and learning_rate, which shrinks the contribution of each tree.

LogisticRegression[21] is the machine learning model that uses logistic regression for classification. The hyper-parameters that are tuned for this model are C, which is a penalty that reduces overfitting, solver, which is the algorithm that is used by the model in the optimisation problem, and class_weight.

SVC[22] implements support vector machines in its machine learning models. The hyper-parameters that are tuned for this model are C, kernel, which is an algorithm for pattern analysis to solve non-linear problems, and class_weight.

DecisionTreeClassifier[23] is a classifier that is based on the decision tree learning algorithm. The hyper-parameters that are tuned for this model are max_depth, criterion, and class_weight.

KNeighborsClassifier[24] implements an instance-based learning method for classification. The hyper-parameters that are tuned for this model are n_neighbors, which is the number of neighbours to be considered, weights, which is a weight function utilised for the prediction, and p, a power parameter.

GaussianNB[25] implements the Gaussian Naive Bayes algorithm for classification. GaussianNB does not support hyper-parameter tuning.

MLPClassifier[26] implements a neural network and allows for multiple hidden layers. The hyper-parameters that are tuned for this model are hidden_layers_sizes, which determines the amount of neurons in a layer, alpha, which is a penalty that reduces overfitting, learning_rate_init, which is the initial learning rate, activation, which is a function that decides if a neuron is to be activated, and solver.

---

[20] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html
[21] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
[22] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[23] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
[24] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
[25] https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html#sklearn.naive_bayes.GaussianNB
[26] https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

3.6 Feature selection

Selecting the appropriate features from a dataset is an important step in the application of machine learning methods. Datasets frequently consist of too many variables to build effective machine learning models. In many cases not all the variables in the datasets are relevant to the classification. However, the relevance of the variables is not known beforehand. A subset of useful variables may exclude many redundant, but relevant, variables.

Kohavi and John (1997) state that many machine learning algorithms perform worse when the number of variables is much higher than optimal. Therefore, for practical reasons, it is desirable to choose the smallest feature set that produces the best possible classification results. This problem, called the minimal-optimal problem, has been extensively investigated, and many algorithms have been developed to reduce the feature set to a reasonable size (Nilsson et al., 2007). Working with overly large feature sets can inhibit the speed of algorithms, use up more resources, and can be inconvenient.

Multicollinearity is a common problem in machine learning models. It happens when two or more independent variables are highly correlated with each other in a machine learning model. This can impact model interpretability because it can be difficult to determine which predictor variable is having the biggest impact on the target variable. Multicollinearity can also reduce the accuracy of your predictions and the stability of a machine learning model. This is because minor changes in the data may lead to large changes in the model.

To mitigate multicollinearity, two main approaches are used according to Chen and Li (2022): the variable selection approach and the modified estimator approach. Variable selection methods involve selecting a subset of the predictor variables that are most important in determining the outcome variable. Modified estimator methods involve modifying the coefficients of the predictor variables to reduce the impact of multicollinearity.

In this research, multicollinearity is mitigated by first calculating the relationship between to features/variables. Pearson's r is used. Chee (2015) defines Pearson's r as a statistic that quantifies the linear relationship between two variables of interval or ratio scale. Pearson's r can have a value between -1 to 1. A value of 0 for the correlation coefficient implies that the variables are unrelated, while a value of 1 indicates that the variables have a perfect positive relationship. A value of -1 indicates that the variables have a perfect negative relationship.

Recursive Feature Elimination (RFE) is a feature selection method that aims to identify the most relevant features in a dataset by iteratively removing the least important features. It is commonly used in machine learning and data analysis to improve the performance of predictive models by reducing the dimensionality of the data and removing redundant or irrelevant features. Yang and Pedersen (1997) found that RFE performs well in terms of both classification accuracy and computational efficiency, making it a promising method for feature selection in text categorisation tasks.

The RFE algorithm works by fitting a model to the data and ranking the features based on their calculated importance. The feature with the lowest importance is then eliminated, and the model is re-trained on the remaining features. This process is recurring until a desired number of features is reached or the performance of the model plateaus. However, it is important to take into account that the performance of RFE can be affected by the choice of the underlying model and the criterion used to rank the features. As such, it is important to carefully consider these factors when using RFE for feature selection. Figure 5, below, shows the process to eliminate the features that do not significantly influence the ranking of a submission using that specific machine learning model.

Algorithm 2: Recursive feature elimination incorporating resampling

```
for Each Resampling Iteration do
    Partition data into training and test/hold–back set via resampling
    Tune/train the model on the training set using all predictors
    Predict the held–back samples
    Calculate variable importance or rankings
    for Each subset size S_i, i = 1 ... S do
        Keep the S_i most important variables
        [Optional] Pre–process the data
        Tune/train the model on the training set using S_i predictors
        Predict the held–back samples
        [Optional] Recalculate the rankings for each predictor
    end
end
Calculate the performance profile over the S_i using the held–back samples
Determine the appropriate number of predictors
Estimate the final list of predictors to keep in the final model
Fit the final model based on the optimal S_i using the original training set
```

Figure 5: Example Recursive Feature Elimination algorithm (source: https://topepo.github.io/caret/recursive-feature-elimination.html)

This paper deals with feature selection in the following manner. Firstly, all the aforementioned features are created. Then a Pearson's r test is run for every combination of features. For every pair that has a correlation higher than 0.7, one of each pair is eliminated, reducing the number of features. The data is then used to train a machine learning model. The hyper-parameters are first tuned using GridSearchCV() from the *sklearn* Python package. Then RFE is applied using RFECV(), also from Python's *sklearn* package. Appendix B shows the complete code. The functions GridSearchCV() and RFECV() are used, this entails that cross-validation is used for both hyper-parameter tuning, as well as RFE. The functions also allow for the definition of an evaluation metric, which a model tries to maximise during training. In this paper the evaluation metric f1 macro is used. Macro means that it calculates the f1 metrics for each class, and finds their unweighted mean. This disregards class imbalances.[27] The reason for choosing f1 macro is that the aim is that correct classification into each class is regarded as of the same importance. The reason for choosing the f1 macro score is further addressed in the paragraph 'Model performance assessment'.

However, reducing the number of features through RFE and multicollinearity elimination, possibly relevant features could be eliminated. The machine learning models are, therefore, trained using four different methods. The first methods includes both RFE and multicollinearity elimination. The second method includes only RFE. The third method includes only multicollinearity elimination. And the fourth methods uses all created features.

Figure 6 shows the correlation matrix. The features '*nr_coaches'* and '*coach_coverage'* are highly correlated and therefore is *nr_coaches* eliminated. The features '*long_words_norm',* '*complex_words_norm'* and *'DaleChall'* are highly correlated and therefore are *complex_words_norm* and *DaleChall* eliminated. The features '*total_word_count'* and '*total_sentences'* are highly correlated and therefore is *total_sentences* eliminated. Table 5 shows the different models and their characteristics.

[27] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html#sklearn.metrics.f1_score

Table 5: All models' characteristics

	N = 164				N = 248			
Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
team_size	X	X	X	X	X	X	X	X
coach_coverage	X	X	X	X				
Jaccard	X	X	X	X				
nr_coaches			X	X				
long_words_norm	X	X	X	X	X	X	X	X
complex_words_norm			X	X			X	X
total_word_count	X	X	X	X	X	X	X	X
total_sentences			X	X			X	X
DaleChall			X	X			X	X
FRE	X	X	X	X	X	X	X	X
highest_similarity	X	X	X	X	X	X	X	X
media	X	X	X	X	X	X	X	X
Highest_edu	X	X	X	X	X	X	X	X
RFE	X		X		X		X	

The X in the table means that the feature is present in the model

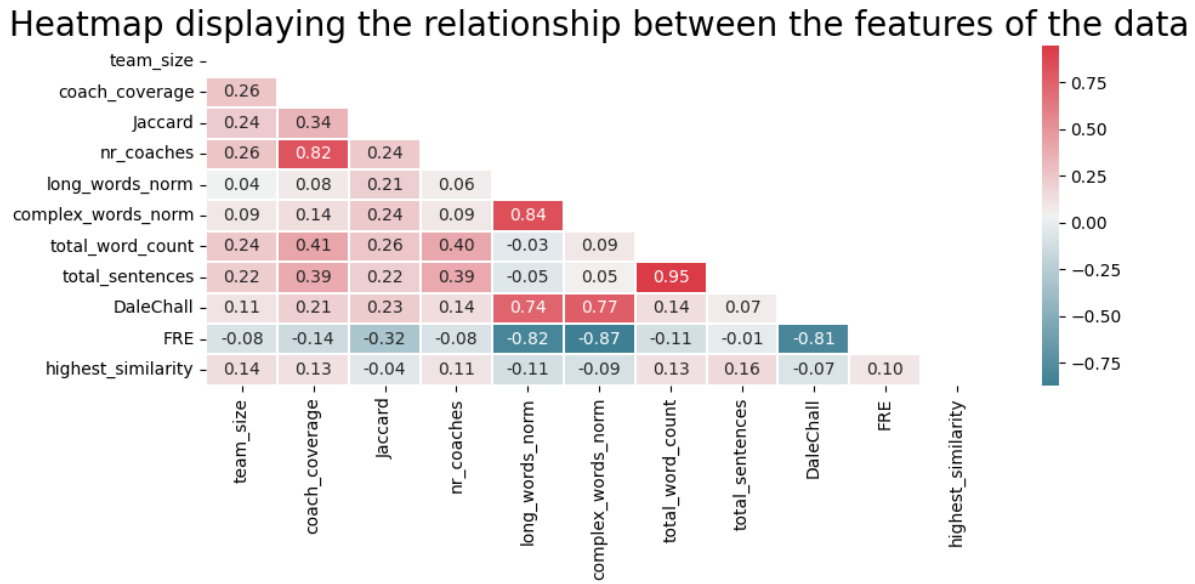


Figure 6: The correlation matrix between all features using Pearson's r

3.7 Model performance assessment

In the chapter on machine learning in this paper, it is specified that there are multiple evaluation methods for machine learning models. Using the *sklearn* package in Python, the performance can be calculated through the confusion matrix function. However, depending on the dataset, one performance indicator (accuracy, precision, recall, f1, etc.) could fit the data better than others. The evaluation method used in this paper is f1 macro score, the

dataset is unbalanced and the aim is to correctly categorise into each class. Precision can be used in email spam detection.[28] An email that is incorrectly labelled as spam can be very costly to the user, so the number of false positives need to be as low as possible. Recall can be used in sick patient detection.[28] If a sick patient is flagged as a not sick patient, this can be detrimental to their health. The number of false negatives has to be reduces as much as possible in this case. The f1-score seeks to find a balance between precision and recall and be similar to the accuracy. However, the f1-score shows its power in when there is an imbalance between classes.[28] The classes in this research are imbalanced (figure 4) and the aim is to reduce false negatives and false positives, as such f1 macro is used. The confusion matrix is used to visualise the classification.

The dataset is strategically split into seven equal parts. As there is limited data, each part is used as a test dataset, where the other six parts are used for training. This allows for cross-validation. The proportion of submissions for each class label is equal between the train and test dataset. This ensures that the model is evaluated on a representative sample of the dataset. The training dataset is used to train the model. The machine learning models, *RandomForestClassifier*, *GradientBoostingClassifier*, *LogisticRegression* , *SVC*, *DecisionTreeClassifier*, *KNeighborsClassifier*, *GaussianNB*, *MLPClassifier*, are all trained using the same training dataset. As there are four different methods for feature selection, there will be 32 models (eight different machine learning models times four feature selection methods) to evaluate. Additionally, a dataset with 248 submissions and a dataset with 164 submissions are used, doubling the total number of trained models to 64 models. Using the accuracy, it is possible to determine which model most optimally classifies the test dataset. To give the result more weight, the accuracy of the models is benchmarked against the no information rate. A binomial test is used to check if the difference is significant. Thus, it is possible to determine which machine learning model best fit the dataset and to determine if machine learning models are effective in determining the quality of the submissions.

Benchmarking

[28] https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

Results

In this chapter the resulting findings from the machine learning models are discussed. Firstly, the multivariate analysis is discussed for the datasets with 164 and 248 observations. Thereafter, the results of training and testing all the different machine learning models with the datasets are presented.

Multivariate analysis

Figure 7 shows the multivariate analysis of the dataset with 164 observations. Each variable is plotted against the target 'rank' variable. Boxplots are used for numeric variables and bar graphs are used for the binary variables. The first variable in figure 7 is 'team_size'. The median and third quartile are greater when the rank is better. However, the maximum for the best rank 'Finalist' is lower than for the middle rank. This might suggest that the ideal team size should not exceed five members. The second variable 'coach_coverage' shows that the first quartile, the median, and the third quartile are higher with a better rank. This could indicate that a better coach coverage could lead to a better ranking. The Same correlation is found for the variable 'coach_coverage'. The variable named 'Jaccard' shows a higher median for the best rank 'Finalist', but the third quartile is the same between 'Top 40' and "Finalist. The variable 'long_words_norm' shows a higher first quartile, median, and third quartile the better the rank is. No such correlation is found for the minimum and maximum. This result might suggest that more longer words in the description might lead to better results, even though the minimum and maximum show higher variance. The variable 'complex_words_norm' does not show any clear relationship between the variable and the target variable. The variable 'FRE' shows the opposite correlation between rank and said variable. For the Flesh Reading-ease score, a lower score depicts a higher level of writing. A higher level of writing might be favourable to achieve a better ranking. 'total_word_count', 'total_sentences', and 'DaleChall' all show a higher first quartile, median, and third quartile the better the rank is. This might suggest that more sentences and more words could perform better in terms of the ranking of a submission. A better readability score following the DaleChall formula also shows that a better score is correlated with a better rank. The variable 'highest_similarity' does not show a clear correlation between rank and the 'highest_similarity' variable. The variable 'media' shows that submissions with any form of media rarely achieve the worst rank, whereas most submissions without any media did achieve the worst rank. The contrary is true for the reverse, where submissions with media

tend to finish with the best rank and submissions without media tend to not finish with the best rank. The final variable ‘Highest_edu’ teams with at least one person doing a Master’s or PHD programme tend to finish with a higher rank than a lower rank. Teams composed of students only following a Bachelor’s programme do not seem to achieve a particular rank.

Figure 8 shows the multivariate analysis of the dataset with 248 observations. Each variable is plotted against the target ‘rank’ variable. Boxplots are used for numeric variables and bar graphs are used for the binary variables. The findings for the dataset with 248 observations are largely the same as the dataset with 164 observations. Where the findings differ are for the variable ‘long_words_norm’, where a positive correlation between the variable and the target variable is not clearly depicted. The slight difference between ‘Top 40’ and ‘Finalist’ has disappeared. In a similar way, the difference between ‘Top 40’ and ‘Finalist has disappeared for the multivariate analysis between the FRE and the rank. The correlations found between the other variables and the target variable remain the same.

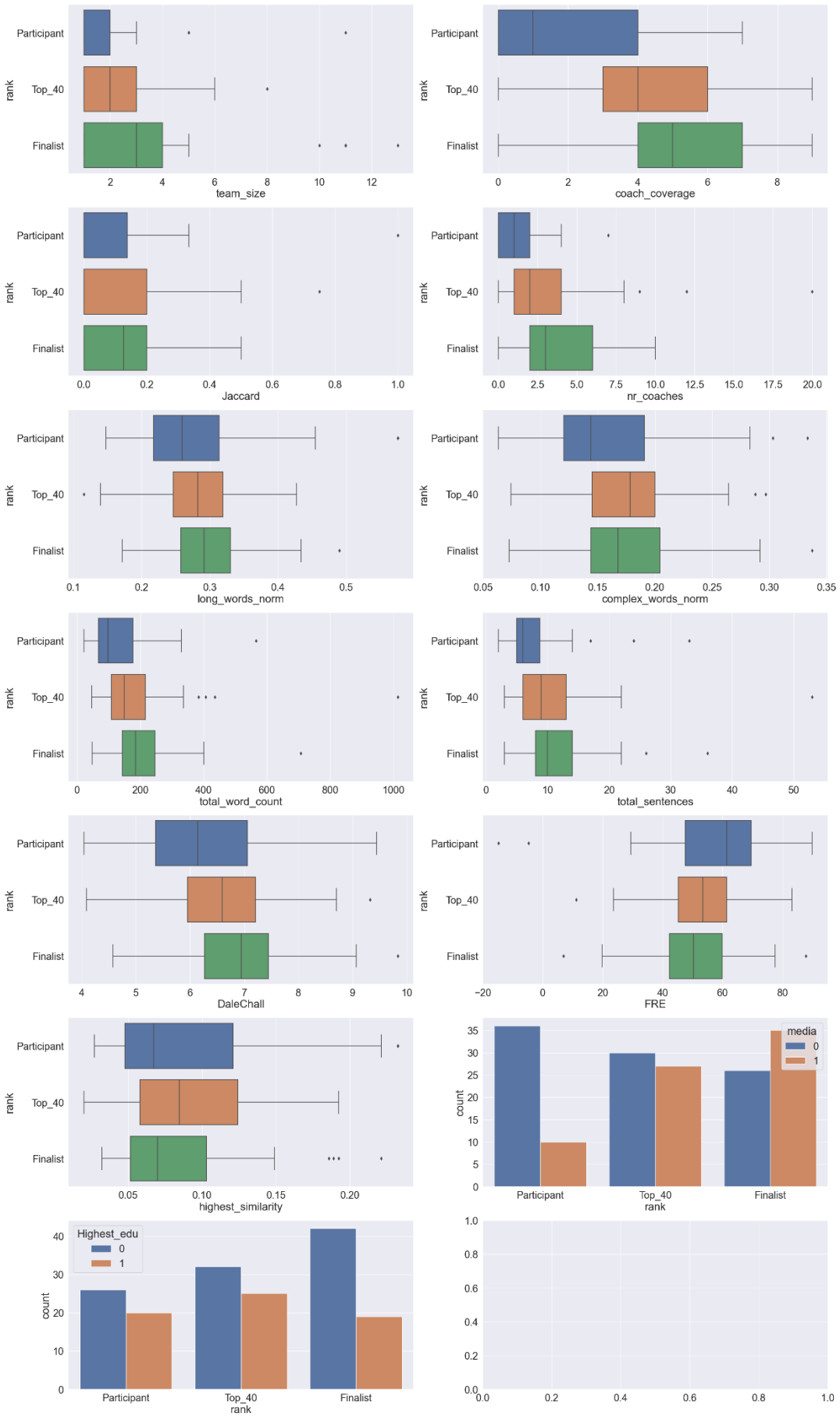


Figure 7: The multivariate analysis of dataset with N=164

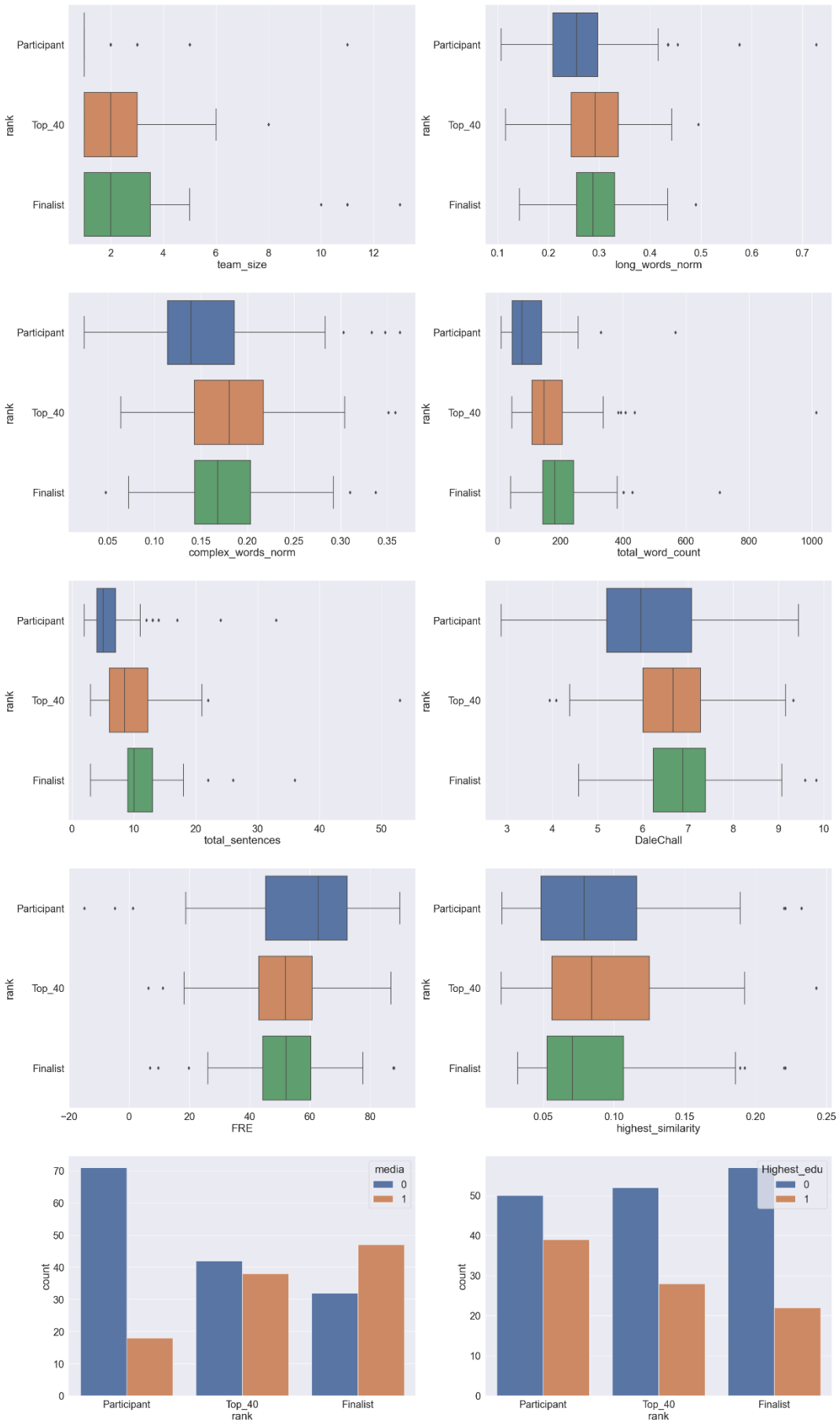


Figure 8: The multivariate analysis of dataset with N=248

Machine learning models

Table 6 depicts the accuracy and f1 macro score of each machine learning model using each variation of the dataset with 164 observations. The values in the columns are the average of seven models for cross-validation. Table 6 shows that on average the logistic regression machine learning model performed the best with an accuracy of 0.477 with a standard deviation of 0.032 and an f1 macro score of 0.469 with a standard deviation of 0.035. This result is 2.6% better than the average of all machine learning models in terms of f1 macro score and accuracy. This finding suggests that the logistic regression machine learning model fits this particular dataset best.

Using a binomial test, the accuracy of the average can be benchmarked against the no information rate to see if the average is statistically significantly better.. Figure 4 shows the distribution of classes for the dataset with 164 observations. The no information rate classifies all observations into the most common class. The most common class is 'Finalist' with 61 observations so the no information rate would be 61/164=0.3720. The accuracy of the models are obtained by testing a test dataset against the trained model. The test dataset is $1/7^{th}$ the size of the complete dataset, so the accuracy is obtained using 23 or 24 observations.

With 23 observations and a no information rate of 0.3720, an accuracy of at least 0.565 ($p < 0.05$) or 0.652 ($p < 0.01$) is needed to be significant at a 95% or 99% confidence level, respectively. Neither the average of the best performing machine learning model nor the best performing singular model, DecisionTree with no multicollinearity and no RFE applied, have an accuracy higher than the requirement.

With 24 observations and a no information rate of 0.3720, an accuracy of at least 0.583 ($p < 0.05$) or 0.667 ($p < 0.01$) is needed to be significant at a 95% or 99% confidence level, respectively. Neither the average of the best performing machine learning model nor the best performing singular model, DecisionTree with no multicollinearity and no RFE applied, have an accuracy higher than the requirement.

Table 7 depicts the accuracy and f1 macro score of each machine learning model using each variation of the dataset with 248 observations. The values in the columns are the average of seven models for cross-validation. Table 7 shows that on average the decision tree model performs best in terms of the f1 macro score, and the MLP model performs best in terms of accuracy. The decision tree model performs 3.3% and 1.9% better in terms of f1 macro score and accuracy, respectively. The MLP model performs 0.7% and 2.6% better in terms of f1 macro score and accuracy, respectively.

Using a binomial test, the accuracy of the average can be benchmarked against the no information rate to see if the average is statistically significantly better.. Figure 4 shows the distribution of classes for the dataset with 248 observations. The no information rate classifies all observations into the most common class. The most common class is 'Participant' with 89 observations so the no information rate would be 89/248=0.3589. The accuracy of the models are obtained by testing a test dataset against the trained model. The test dataset is $1/7^{th}$ the size of the complete dataset, so the accuracy is obtained using 35 or 36 observations.

With 35 observations and a no information rate of 0.3589, an accuracy of at least 0.514 ($p < 0.05$) or 0.571 ($p < 0.01$) is needed to be significant at a 95% or 99% confidence

level, respectively. With 36 observations and a no information rate of 0.3589, an accuracy of at least 0.528 ($p < 0.05$) or 0.583 ($p < 0.01$) is needed to be significant at a 95% or 99% confidence level, respectively. The average accuracy of all MLP models does not exceed 0.528. However, SVC model 8, LogisticRegression model 7, DecisionTree model 5 & 6, and MLP model 5, 7 & 8 all have a higher accuracy than 0.528 and perform therefore statistically significantly better than the no information rate.

Table 6: Results from the machine learning models for dataset N=164

N = 164, mean(std. dev.)		Model 1	Model 2	Model 3	Model 4	Mean(std. dev.) for N = 164
SVC	**F1 macro**	0.484(0.094)	0.463(0.035)	0.417(0.062)	0.453(0.104)	0.454(0.028)
	Accuracy	0.488(0.108)	0.470(0.045)	0.426(0.065)	0.457(0.108)	0.460(0.032)
RandomForest	**F1 macro**	0.434(0.126)	0.410(0.087)	0.386(0.092)	0.458(0.079)	0.422(0.031)
	Accuracy	0.440(0.115)	0.402(0.079)	0.395(0.096)	0.475(0.060)	0.428(0.023)
LogisticRegression	**F1 macro**	0.477(0.054)	0.449(0.050)	0.469(0.103)	0.482(0.121)	**0.469(0.035)**
	Accuracy	0.482(0.060)	0.458(0.061)	0.476(0.107)	0.494(0.124)	**0.477(0.032)**
DecisionTree	**F1 macro**	0.465(0.100)	**0.522(0.066)**	0.388(0.056)	0.425(0.103)	0.450(0.024)
	Accuracy	0.470(0.093)	**0.531(0.054)**	0.403(0.049)	0.438(0.110)	0.461(0.030)
KNeighbors	**F1 macro**	0.411(0.083)	0.442(0.077)	0.356(0.121)	0.414(0.108)	0.406(0.036)
	Accuracy	0.421(0.086)	0.452(0.089)	0.390(0.104)	0.428(0.114)	0.423(0.013)
GaussianNB	**F1 macro**	0.496(0.094)	0.492(0.087)	0.418(0.076)	0.453(0.087)	0.465(0.037)
	Accuracy	0.493(0.098)	0.482(0.090)	0.433(0.079)	0.469(0.092)	0.469(0.008)
GradientBoosting	**F1 macro**	0.425(0.117)	0.442(0.072)	0.402(0.025)	0.397(0.042)	0.417(0.040)
	Accuracy	0.426(0.115)	0.445(0.070)	0.408(0.030)	0.396(0.048)	0.419(0.036)
MLP	**F1 macro**	0.453(0.039)	0.477(0.026)	0.438(0.101)	0.463(0.082)	0.458(0.016)
	Accuracy	0.452(0.038)	0.482(0.029)	0.463(0.104)	0.476(0.086)	0.468(0.036)
Mean of mean	**F1 macro**					0.443(0.024)
	Accuracy					0.451(0.023)

Model 1: No multicollinearity between features + RFE applied, Model 2: No multicollinearity between features + RFE not applied, Model 3: Multicollinearity between features + RFE applied, Model 4: Multicollinearity between features + RFE not applied

Table 7: Results from the machine learning models for dataset N=248

N = 248, mean(std. dev.)		Model 5	Model 6	Model 7	Model 8	Mean(std. dev.) for N = 248
SVC	**F1 macro**	0.475(0.112)	0.487(0.149)	0.474(0.083)	0.512(0.095)	0.487(0.018)
	Accuracy	0.497(0.102)	0.498(0.154)	0.489(0.085)	0.529(0.092)	0.503(0.031)
RandomForest	**F1 macro**	0.445(0.123)	0.432(0.129)	0.437(0.084)	0.496(0.094)	0.453(0.029)
	Accuracy	0.465(0.134)	0.445(0.140)	0.452(0.093)	0.505(0.092)	0.467(0.026)
LogisticRegression	**F1 macro**	0.483(0.120)	0.490(0.124)	0.517(0.107)	0.513(0.104)	0.501(0.010)
	Accuracy	0.502(0.129)	0.510(0.131)	0.534(0.114)	0.525(0.110)	0.517(0.011)
DecisionTree	**F1 macro**	0.524(0.153)	**0.541(0.122)**	0.477(0.069)	0.481(0.067)	**0.506(0.042)**
	Accuracy	0.529(0.146)	0.545(0.117)	0.484(0.071)	0.492(0.068)	0.513(0.038)
KNeighbors	**F1 macro**	0.463(0.082)	0.465(0.105)	0.440(0.162)	0.437(0.087)	0.451(0.015)
	Accuracy	0.497(0.080)	0.493(0.117)	0.472(0.148)	0.461(0.092)	0.481(0.030)

GaussianNB	**F1 macro**	0.437(0.091)	0.447(0.099)	0.476(0.124)	0.468(0.106)	0.457(0.018)
	Accuracy	0.464(0.088)	0.477(0.104)	0.509(0.120)	0.497(0.109)	0.487(0.013)
GradientBoosting	**F1 macro**	0.440(0.087)	0.446(0.088)	0.448(0.103)	0.447(0.096)	0.445(0.007)
	Accuracy	0.457(0.086)	0.465(0.085)	0.460(0.102)	0.456(0.099)	0.459(0.009)
MLP	**F1 macro**	0.485(0.107)	0.433(0.065)	0.500(0.128)	0.503(0.116)	0.480(0.032)
	Accuracy	0.541(0.085)	0.468(0.062)	0.533(0.104)	**0.545(0.101)**	**0.522(0.019)**
Mean of mean	**F1 macro**					0.473(0.024)
	Accuracy					0.494(0.024)

Model 5: No multicollinearity between features + RFE applied, Model 6: No multicollinearity between features + RFE not applied, Model 7: Multicollinearity between features + RFE applied, Model 8: Multicollinearity between features + RFE not applied

The effects of the independent variables cannot be determined by the cross-validated means of each combination of machine learning model and variation of the dataset. Therefore, the performance of individual test and training folds is assessed. Table 8 shows the ten best performing models overall and figure 9 shows how each model classified the test dataset. The table shows that out of the ten best performing models, six models were trained with the dataset of 248 observations, while four were trained using the dataset of 164 observations. Another interesting finding is that only two different folds were used as the test dataset. For cross-validation the datasets are split into seven equal parts, these parts are the folds mentioned in the table. This finding could suggest that folds 5 and 7 best represent the rest of the dataset.

The table shows that the tenth best performing model uses the GaussianNB model with an f1 macro score of 0.651 and an accuracy of 0.652. The accuracy is well above the required 0.583 accuracy to be statistically significant. This also means that all the other models in the top ten perform significantly better than the no information rate. This is further signified by the row 'One-sided binomial test (p-value) using accuracy', which shows a p-value < 0.05 for each model.

Figure 9 shows that in general the machine learning models can predict the rank "top 40" most accurately followed by "participant. The rank "finalist" is the worst predicted rank. This can be seen by the values on the diagonal from top left to bottom right compared to the other values in the column

The row 'Features importance' shows how important a value is to the eventual f1 macro outcome. The feature importance is determined by how much the f1 macro score decreases when a variable is not available. The method is called permutation importance and instead of completely removing the variable, it replaces it with random noise. The variable is still in the machine learning model, but the information in no longer useful[29]. For example, in the overall best performing model, the DecisionTreeClassifier(), the f1 macro score decreases on average by 0.259 if 'team_size' is "left out".

Table 9 shows whether the variable positively affected the f1 macro score of all the models that performed statistically significantly better than the no information rate with a confidence level of 95%. The first column shows the variables. The third column shows in how many models the variable was used in. Due the removal of correlated variables, as well as leaving out some variables in the dataset with 248 observations, the number of models a

[29] https://eli5.readthedocs.io/en/latest/blackbox/permutation_importance.html

variable can be included in can differ. Table 9 shows that the variables ‘team_size’, ‘total_word_count’, ‘coach_coverage’, and ‘nr_coaches’ positively affected the f1 macro score in at least 75% of the models they could be included in. They also show a mean permutation value of between 0.088 and 0.1062. ‘DaleChall’, ‘total_sentences’, ‘FRE’, and ‘media’ positively influenced the f1 macro score in at least 50% of the models they could be included in. These variables also show a mean permutation importance value between 0.0227 and 0.0332. The variables ‘Jaccard’, ‘Highest_similarity’, ‘Highest_edu’, ‘long_words_norm’, and ‘complex_words_norm’ were the least significant variables. These variables only positively affected the f1 macro score in 50% of the possible models at most. On average these variables at best have a permutation importance value of 0.0118, in case of ‘Jaccard’, and at worst it is a negative value, in case of ‘Highest_edu’.

Table 10 shows the same statistics as table 9, except for a confidence level of 99%. The variables ‘team_size’, ‘coach_coverage’, and ‘total_word_count’ positively affected the f1 macro score in at least 75% of the models they could be in. With a sample size of two, ‘coach_coverage’ on average has a permuatation importance value of 0.215. The variables ‘team_size’ and ‘ total_word_count’ have a mean permutation importance value between 0.11-0.12. ‘DaleChall’, ‘total_sentences’, ‘FRE’, ‘media’, ‘nr_coaches, and ‘Jaccard’ positively affected the f1 macro score in at least 50% of the possible models they could have been in. With the exception of ‘nr_coaches’, which has a mean permutation value of 0.069, the mean permutation value of the variables is around 0.03 and 0.04. ‘Highest_similarity’, ‘Highest_edu’, ‘long_words_norm’, and ‘complex_words_norm’ were the least significant variables. These variables only positively affected the f1 macro score in 50% of the possible models at most and have a mean permutation importance value of at most 0.0115.

Table 11 shows the distribution of the models that were significantly better than the no information rate based on the number of observations in the dataset. Models that were trained using the dataset with 248 observations account for 83% and 96% of all the models that were statistically significantly better than the no information rate for a confidence level of 95% and 99%, respectively.

Overall best 10 models	1	2	3
Machine learning model	DecisionTreeClassifier()	DecisionTreeClassifier()	LogisticRegression()
N	248	248	164
F1 macro score	0.717	0.717	0.716
Accuracy	0.714	0.714	0.739
Hyperparameters	class_weight = balanced, criterion = gini, max_depth = 2	class_weight = balanced, criterion = gini, max_depth = 2	max_iter = 100000, C = 0.2, class_weight = balanced, solver = lbfgs
Features importance (mean score decrease(std. dev.))	team_size: 2.59E-01(1.25E-01) total_word_count: 2.38E-01(7.84E-02) Highest_edu: 0.00E+00(0.00E+00) media: 0.00E+00(0.00E+00) FRE: 0.00E+00(0.00E+00) long_words_norm: 0.00E+00(0.00E+00)	team_size: 2.57E-01(4.69E-02) total_word_count: 2.26E-01(2.30E-02) Highest_edu: 0.00E+00(0.00E+00) media: 0.00E+00(0.00E+00) highest_similarity: 0.00E+00(0.00E+00) FRE: 0.00E+00(0.00E+00) long_words_norm: 0.00E+00(0.00E+00)	FRE: 1.75E-01(6.15E-02) DaleChall: 1.65E-01(4.05E-02) nr_coaches: 1.38E-01(8.72E-02) total_word_count: 1.33E-01(9.47E-02) coach_coverage: 1.10E-01(6.70E-02) team_size: 7.82E-02(2.27E-02) Highest_edu: 7.34E-02(2.33E-02) Jaccard: 6.25E-02(3.54E-02) media: 5.27E-02(4.53E-02) highest_similarity: 5.03E-02(3.33E-02) total_sentences: 0.00E+00(0.00E+00) complex_words_norm: 0.00E+00(0.00E+00) long_words_norm: 0.00E+00(0.00E+00)
Multicollinearity between features	No	No	Yes
RFE applied	Yes	No	No
Test fold	7	7	7
One-sided binomial test (P-value) using accuracy	3.91E-05	3.91E-05	1.51E-04

Table 8: The 10 best performing machine learning models

Overall best 10 models	4	5	6
Machine learning model	LogisticRegression()	MLPClassifier()	SVC()
N	164	248	248
F1 macro score	0.687	0.683	0.673
Accuracy	0.696	0.686	0.686
Hyperparameters	max_iter = 100000, C = 0.1, class_weight = None, solver = lbfgs	max_iter = 100000, activation = logistic, alpha = 0.1, hidden_layer_sizes = [3], learning_rate_init = 0.01, solver = adam	gamma = auto, C = 0.3, class_weight = balanced, kernel = linear
Features importance (mean score decrease(std. dev.))	coach_coverage: 3.20e-01(6.29e-02) DaleChall: 1.19e-01(5.23e-02) team_size: 8.27e-02(5.52e-02) media: 6.76e-02(6.48e-02)	team_size: 1.70E-01(8.20E-02) total_word_count: 1.19E-01(5.19E-02) media: 9.77E-02(3.61E-02) FRE: 9.16E-02(3.55E-02) DaleChall: 7.55E-02(2.24E-02) highest_similarity: 7.25E-02(2.98E-02) total_sentences: 5.05E-02(6.69E-02) Highest_edu: 0.00E+00(0.00E+00) complex_words_norm: 0.00E+00(0.00E+00) long_words_norm: 0.00E+00(0.00E+00)	team_size: 1.88E-01(4.38E-02) total_word_count: 1.85E-01(6.41E-02) media: 1.64E-01(3.28E-02) FRE: 6.43E-02(3.69E-02) highest_similarity: 1.90E-02(3.93E-02) long_words_norm: 0.00E+00(0.00E+00) Highest_edu: -4.55E-02(2.99E-02)
Multicollinearity between features	Yes	Yes	No
RFE applied	Yes	No	No
Test fold	7	5	7
One-sided binomial test (P-value) using accuracy	1.57E-04	1.57E-04	1.57E-04

Overall best 10 models	7	8	9	10
Machine learning model	GaussianNB()	MLPClassifier()	GaussianNB()	GaussianNB()
N	248	248	164	164
F1 macro score	0.664	0.655	0.651	0.651
Accuracy	0.686	0.657	0.652	0.652
Hyperparameters		max_iter = 100000, activation = relu, alpha = 0.2, hidden_layer_sizes = [3], learning_rate_init = 0.01, solver = sgd		
Features importance (mean score decrease(std. dev.))	DaleChall: 9.49E-02(2.49E-02) total_word_count: 8.90E-02(5.48E-02) long_words_norm: 8.12E-02(2.96E-02) total_sentences: 7.58E-02(5.67E-02) Highest_edu: 7.30E-02(1.66E-02) media: 7.03E-02(4.80E-02) complex_words_norm: 5.65E-02(2.75E-02) team_size: 5.44E-02(4.48E-02) highest_similarity: 2.44E-02(2.47E-02)	team_size: 1.47E-01(5.04E-02) total_sentences: 8.56E-02(7.38E-02) total_word_count: 6.14E-02(2.77E-02) DaleChall: 4.85E-02(5.81E-02) FRE: 4.71E-02(2.22E-02) highest_similarity: 1.43E-02(3.05E-02) Highest_edu: -2.82E-02(1.57E-02)	coach_coverage: 9.76E-02(1.03E-01) FRE: 7.10E-02(4.97E-02) total_word_count: 4.30E-02(8.37E-02) Highest_edu: 3.87E-02(3.50E-02) long_words_norm: 2.95E-02(5.81E-02) team_size: 2.26E-02(4.58E-02) media: 1.92E-02(3.53E-02) Jaccard: 6.07E-03(3.41E-02) highest_similarity: 1.25E-03(6.42E-02)	coach_coverage: 9.76E-02(1.03E-01) FRE: 7.10E-02(4.97E-02) total_word_count: 4.30E-02(8.37E-02) Highest_edu: 3.87E-02(3.50E-02) long_words_norm: 2.95E-02(5.81E-02) team_size: 2.26E-02(4.58E-02) media: 1.92E-02(3.53E-02) Jaccard: 6.07E-03(3.41E-02) highest_similarity: 1.25E-03(6.42E-02)
Multicollinearity between features	Yes	Yes	No	No
RFE applied	Yes	Yes	Yes	No
Test fold	5	7	7	7
One-sided binomial test (P-value) using accuracy	5.56E-04	2.85E-03	2.85E-03	2.85E-03

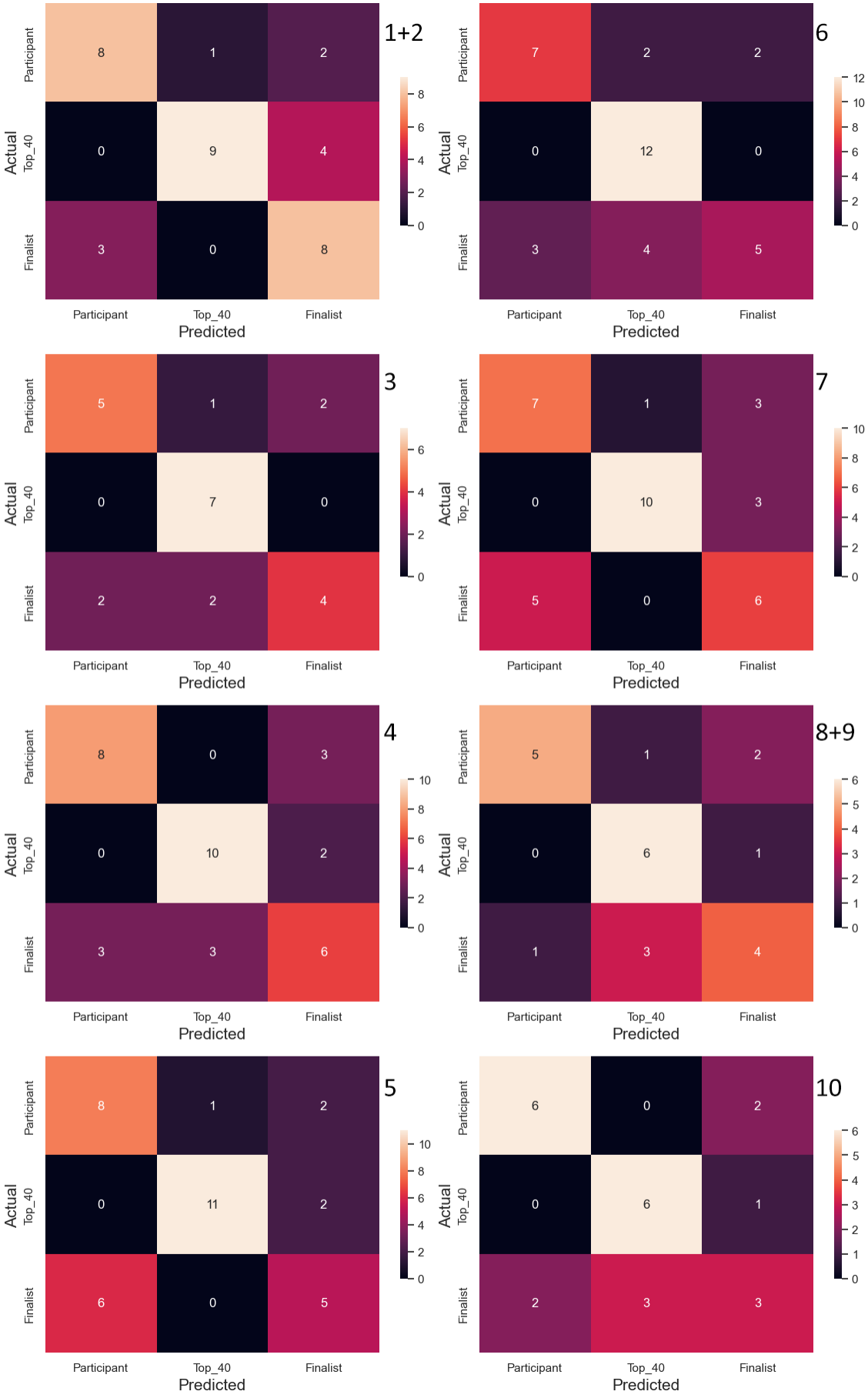


Figure 9: Confusion matrices for the 10 best performing models

Table 9: Accumulation of the variables based on whether they have a positive influence on the performance of the machine learning model. Only machine learning models that are significantly better than the no information rate are included at a 95% confidence level

	95% confidence level			
Variables	**Actual models**	**Possible models**	**Ratio**	**Mean perm. Importance**
team_size	94	103	91%	0.0880
total_word_count	92	103	89%	0.0957
coach_coverage	15	17	88%	0.1062
nr_coaches	7	8	88%	0.0967
DaleChall	34	52	65%	0.0329
total_sentences	33	52	63%	0.0332
FRE	61	103	59%	0.0306
media	57	103	55%	0.0227
Jaccard	8	17	47%	0.0118
Highest_similarity	45	103	44%	0.0082
Highest_edu	31	103	30%	-0.0001
long_words_norm	31	103	30%	0.0083
complex_words_norm	13	52	25%	0.0084

Table 10: Accumulation of the variables based on whether they have a positive influence on the performance of the machine learning model. Only machine learning models that are significantly better than the no information rate are included at a 99% confidence level

	99% confidence level			
Variables	**Actual models**	**Possible models**	**Ratio**	**Mean perm. Importance**
team_size	52	52	100%	0.1178
coach_coverage	2	2	100%	0.2150
total_word_count	50	52	96%	0.1120
DaleChall	20	27	74%	0.0497
total_sentences	19	27	70%	0.0434
FRE	34	52	65%	0.0383
media	29	52	56%	0.0311
nr_coaches	1	2	50%	0.0690
Jaccard	1	2	50%	0.0313
Highest_similarity	25	52	48%	0.0115
Highest_edu	17	52	33%	0.0025
complex_words_norm	6	27	22%	0.0089
long_words_norm	10	52	19%	0.0069

Table 11: Number of models that are significantly better than the no information rate split on dataset size and confidence level

	confidence level	
Observations	**95%**	**99%**
164	17	2

	248	86	50
	total	103	52

Hypothesis testing

Hypothesis 1. *A more descriptive challenge and solution will overall result in a more successful project.*

Tables 9 and 10 show that the total word count significantly affected the performance of the machine learning models. At a 95% confidence level, the total word count positively affected the performance in 89%, with a mean permutation importance value of 0.0957, of the models that were significantly better than the no information rate. At a 99% confidence level the ratio is 96% and a mean permutation importance value of 0.1120. Figures 7 and 8 show a positive relationship between rank and total word count. Based on the previously mentioned findings, the hypothesis cannot be rejected.

Hypothesis 2. *Submissions that use a large vocabulary size for the description of the challenge and solution will be more successful.*

In this research a large vocabulary size is quantified by the number of long words and the number of complex words. Figures 7 and 8 show a positive relationship between the ratio of long words to total words and the rank. They do not show a clear relationship between the ratio of complex words to total words and the rank. In addition to these findings, tables 9 and 10 show that the variables ‘long_words_norm’ and ‘complex_words_norm’ are the least effective variables, having a positive effect on the f1 macro score in at most 30% of the models. This means that in 70% or more of the models that performed significantly better than the no information rate, these two variables did not positively affect the accuracy of the model. The mean permutation importance value for these variables is at most 0.0089, meaning that the inclusion of these variables affected the performance of the machine learning models at most by less than 1%. Based on the findings of this research, the hypothesis should be rejected.

Hypothesis 3. *Submissions by higher educated participants will perform better than submissions by lower educated people.*

Submissions are made either individually or in a group. In this research, the highest education level in the group is used in the dataset. A group consisting of an individual following a PHD/Master's programme where the rest of the group are following a Bachelor's programme is still regarded as a group with a high education level. Figures 7 and 8 show a positive relationship between a higher education and rank. Figure 8 shows a negative relationship between a lower education level and rank. Tables 9 and 10 show that in at most 33% of the models that are significantly better than the no information rate, the variable 'Highest_edu' positively affected the performance of the models. Table 9 even shows that on average the inclusion of this variable negatively impacted the performance of the models. The findings of this research do not support the hypothesis that higher educated people perform better than lower educated people. The hypothesis should be rejected.

Hypothesis 4. *Submissions that have pictures or videos perform better than submissions that do not contain any pictures or videos.*

In this research, pictures and videos are combined into the binary variable 'media'. Figures 7 and 8 show a positive relationship between the presence of media and rank. Figures 7 and 8 shows a negative relationship between the absence of media and rank. Tables 9 and 10 show that in 55-56% of the models that are significantly better than the no information rate, the variable 'media' positively affected the performance of the models by around 2-3% based on the permutation importance value. A relationship can be seen between the presence of media and rank and more than half of the models are positively affected by the variable 'media', therefore, the hypothesis that submissions that had pictures or videos perform better than submissions that do not contain any pictures or videos cannot be rejected.

Hypothesis 5. *Submissions that have a low similarity rate compared to other submissions will be more successful.*

Figures 9 and 10 show that the variable 'Highest_similarity' positively affected the performance of the models in fewer than 50% of the cases. Additionally, figures 7 and 8 do not show a clear relationship between rank and 'Highest_similarity'. Based on these findings, the hypothesis that submissions that have a low similarity rate compared to other submissions will be more successful should be rejected.

Hypothesis 6. *The readability of the submission positively influences the success of the project.*

The readability of the submissions is depicted by the variables ‘FRE’ and ‘DaleChall’ in this research. Figures 7 and 8 show a positive relationship between rank and ‘DaleChall’ and a mostly negative relationship between rank and ‘FRE’. Table 9 shows that ‘DaleChall’ positively affected the performance, at a confidence level of 95%, of the models in 65% of the cases, whereas ‘FRE’ positively affected the performance in 59% of the cases. Table 10 shows it positively affected the performance in 74% of the cases at a confidence of 99%, whereas ‘FRE’ positively affected the performance in 65% of the cases. The variables positively influenced the performance of the machine learning models between 0.0306 and 0.0497. These findings suggest that the hypothesis cannot conclusively be rejected.

Hypothesis 7. *Smaller teams (fewer than five people) will perform better than larger teams.*

Figures 7 and 8 show that for the highest rank the maximum team size is five when outliers are ignored. The figures also show that the lowest rank on average have a team size between one and three. The maximum team size for the middle rank is six. Therefore, a non-linear relationship is found between team size and rank. Tables 9 and 10 show that the variable for team size positively affected the performance of the models in most to all of the cases. The inclusion of the variable for team size positively influenced the performance of the machine learning models by 8.8% and 11.78% at a confidence level of 95% and 99%, respectively. The findings in this research support the hypothesis that teams smaller than five people perform better than teams larger than five. Therefore, the hypothesis cannot be rejected.

Hypothesis 8. *More feedback, meaning more coaches helping the participants, will overall result in a more successful project.*

Three variables in this research are used to study this hypothesis. The variables ‘coach_coverage’, ‘Jaccard’, and ‘nr_coaches’. Figure 7 shows a positive relationship between rank and ‘coach_coverage and a positive relationship between ‘nr_coaches’ and rank. Figure 7 also shows a positive relationship between rank and ‘Jaccard’, however, this relationship is less significant and less clear in the visualisation. Tables 9 and 10 show that the variable ‘Jaccard’ positively affected the performance of the models in 8 out of 17 and 1 out of 2 cases. A perfect match between all the fields of expertise of the coaches and the clusters of the submission does not seem impact the performance of the models significantly. ‘coach_coverage’ and ‘nr_coaches’ show a high impact on the performance of the machine

learning model, where, at a 99% confidence level, ‘coach_coverage’ shows a mean permutation value of 0.2150 over 2 samples. Due to the positive relationship between rank and these variables and the significant impact they have on the performance of the models, this hypothesis cannot be rejected.

Discussion

The aim of this research paper is to answer the following research questions: “How can the idea quality be evaluated using a machine learning approach, based on participants’ submissions in an innovation contest?”. In this paper the idea quality is equated to the performance in an innovation contest, a higher rank in the contest is therefore equated to a better idea. The research does not show that there is one specific machine learning model that performs significantly better than the no information rate or the other machine learning models. Overall the MLPclassifier() was the best performing model and came the closest to being significantly better than the no information rate. However, this is only the case when the larger dataset with 248 observations is considered. The research found that the team size is an important factor in the quality of the idea. The ideal size of a team should be around two to five people. Another factor that is important is how well the idea challenge and solution are elaborated. A positive relationship is found between the number of words used and the rank in the competition. A positive relationship is also found between the number of sentences and the rank in the competition. The readability of the description of the idea is another factor that is found to impact the idea quality. When an idea is formulated on a more academic level, the idea is regarded as of higher quality.

The second research question in this paper is as follows: **“**How does the quality of feedback influence idea quality, based on how the feedback source’s expertise matches the idea topic?”. The research found that a perfect match between the expertise of the coaches and the idea topic does not necessarily impact the quality of the idea. Rather, coaches with different fields of expertise have shown to positively influence the perception of the quality of the idea.

There are a few points that can be explored in future research. This paper finds that a majority of the significantly better performing machine learning models were using the bigger dataset. Additionally, it is found that the ten best performing models came from two of the seven possible training datasets. The ten best performing models also show that all of

these models have a hard time accurately predicting if a submission belongs to the rank "finalist". This finding is in line with Rhyn and Blohm (2017), who also found that their model had a harder time classifying higher quality contributions compared to the classification of lower quality contributions. It is unclear whether this is due to the fact that the test dataset was representative of the training dataset or whether there was too much garbage-in-garbage-out in the other training datasets. In future research more innovation contests can be added to see whether the performance of the machine learning models further improve.

Another point is the manual coding of the idea description to find the clusters the ideas belong to. Due to the manual nature of coding, the interpretation of each idea can vary from researcher to researcher. One way to address this is to find an accurate way to automatically cluster the ideas through machine learning or other available means. Additionally, a more in-depth approach into the manner in which the coaches help the idea submitters can be explored. This research only looked at the surface of the relationship between the submissions and the coaches, the actual interactions between the parties are not considered.

One weakness of this research is the usage of accuracy as a performance metric of the machine learning models. Accuracy is not a good metric when using imbalanced datasets. However, in this research, the accuracy is used to determine whether a model performs significantly better than the no information rate. Future research could look into using a different methodology to the determine how well a machine learning model performed.

Another limiting factor of this research is the requirement for data to be selected for this research. The submissions needed to be ranked into three categories, namely finalist, top 40, and the rest. However if an innovation contest did not use this exact way of ranking, the data from the innovation contest could not be used. This severely limited the amount of data that could be gathered for this paper.

Trustworthiness

The trustworthiness of the research depends on four different qualitative approaches. First, credibility is achieved by engaging with the participants over a longer period of time, by explaining how the data is assessed (Morrow, 2005; Drisko, 2005). Credibility focuses on internal validity. Second, transferability, which focuses on the external validity, looks at the

applicability of the research into a different context (Krefting, 1991). Transferability is achieved through explanation by the researcher and the way the research is set up. Third, dependability indicates whether a research is repeatable by assessing whether research techniques are consistently used (Morrow, 2005). Last, the confirmability of the research reflects on the objectivity of the research, by reflexivity, and by triangulation (Krefting, 1991).

In terms of the internal validity of this research, the following can be stated. The data sample used in this research was chosen due to the availability and due to how it fit the research. The sample was not cherrypicked to achieve a favourable result. The use of accuracy in this paper may not be the most appropriate metric to assess the models and could be seen as measurement bias. The presented resulted are, however, the result of cross-validation and should be reliable. One of the caveats of classification models is that it is hard to accurately explain the effects of individual independent variables have on the dependent variable, unlike in regression models. The permutation importance used in this paper is an approximation of the effect of the independent variables.

The transferability and dependability of this paper rely mainly on the chapter ‘Empirical approach’. Every step in terms of sampling, data collection, and data analysis is thoroughly explained in the empirical approach and could therefore be replicated by other researchers. The process can be used, with small alterations, in other contexts. The empirical approach is not limited to innovation contests and can be applied in other contests, where at least textual information is a key component of the contest.

The confirmability is addressed in the discussion section of this paper.

References

Ahmed, F., & Fuge, M. (2017). Capturing Winning Ideas in Online Design Communities. *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing - CSCW '17*. Presented at the the 2017 ACM Conference. https://doi.org/10.1145/2998181.2998249

Basit, T. (2003). Manual or electronic? The role of coding in qualitative data analysis. *Educational Research, 45*(2), 143-154.

Blair, E. (2015). A reflexive exploration of two qualitative data coding techniques. *Journal of Methods and Measurement in the Social Sciences, 6*(1), 14. https://doi.org/10.2458/v6i1.18772

Blohm, I.; Bretschneider, U.; Leimeister, J. M. & Krcmar, H. (2010). Does collaboration among participants lead to better ideas in IT-based idea competitions - An empirical investigation. *Hawai'i International Conference on System Sciences (HICSS*) 2010, Kauai, USA.

Camacho, N., Nam, H., Kannan, P. K., & Stremersch, S. (2019). Tournaments to Crowdsource Innovation: The Role of Moderator Feedback and Participation Intensity. Journal of Marketing, 83(*2*), 138–157. https://doi.org/10.1177/0022242918809673

Chen, Y., Li, Y., & Li, Y. (2019). Employee innovation using ideation contests: Seven-step process to align strategic challenges with the innovation process. Journal of Business Research, 98, 1-11. https://doi.org/10.1016/j.jbusres.2019.01.012

Chen, Y., Li, Y., & Li, Y. (2021). Seeker exemplars and quantitative ideation outcomes in crowdsourcing ideation contests. Information Systems Research, 32(1), 1-18. https://doi.org/10.1287/isre.2021.1054

Chow, T. E. (2012). "We Know Who You Are and We Know Where You Live": A Research Agenda for Web Demographics. *In Crowdsourcing Geographic Knowledge* (pp. 265–285). https://doi.org/10.1007/978-94-007-4587-2_15

Connolly, T., Jessup, L. M., & Valacich, J. S. (1990). Effects of Anonymity and Evaluative Tone on Idea Generation in Computer-Mediated Groups. *Management Science*, 36(*6*), 689–703. https://doi.org/10.1287/mnsc.36.6.689

Curral, L. A., Forrester, R. H., Dawson, J. F., & West, M. A. (2001). It's what you do and the way that you do it: Team task, team size, and innovation-related group processes. *European Journal of Work and Organizational Psychology*, 10(*2*), 187–204. https://doi.org/10.1080/13594320143000627

Drisko, J. W. (2005). Writing up qualitative research. *Families in Society, 86*(4), 589-593.

Gefen, D., Gefen, G., & Carmel, E. (2016). How project description length and expected duration affect bidding and project success in crowdsourcing software development. *Journal of Systems and Software*, 116, 75–84. https://doi.org/10.1016/j.jss.2015.03.039

Glez-Peña, D., Lourenço, A., López-Fernández, H., Reboiro-Jato, M., & Fdez-Riverola, F. (2013). Web scraping technologies in an API world. *Briefings in Bioinformatics*, 15(*5*), 788–797. https://doi.org/10.1093/bib/bbt026

Hoekstra, R., ten Bosch, O., & Harteveld, F.. (2012). Automated data collection from web sources for official statistics: First experiences. Statistical Journal of the IAOS, 28(3,4), 99–111. https://doi.org/10.3233/SJI-2012-0750

Jiang, Z. (Zoey), & Huang, Y. (2016). The Role of Feedback in Dynamic Crowdsourcing Contests: A Structural Empirical Analysis. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.2884922

Krefting, L. (1991). Rigor in qualitative research: the assessment of trustworthiness. *American Journal of Occupational Therapy*, *45*, 214-222. https://doi.org/10.5014/ajot.45.3.214

Li, Y., Chen, Y., & Li, Y. (2021). The textual features of an idea description and their effects on idea quality in crowdsourcing ideation contests. Journal of Business Research, 135, 1-12. https://doi.org/10.1016/j.jbusres.2021.08.027

Morrow, S. L. (2005). Quality and trustworthiness in qualitative research in counseling psychology. *Journal of Counseling Psychology, 52*(2), 250.

Nagpal, A. (2017). Decision tree ensembles- bagging and boosting

Rhyn, M. & Blohm, I. (2017). A Machine Learning Approach for Classifying Textual Data in Crowdsourcing. *13th International Conference on Wirtschaftsinformatik (WI)*, St. Gallen, Switzerland.

Schnell, R., & Redlich, S. (2019). Web Scraping Online Newspaper Death Notices for the Estimation of the Local Number of Deaths. *Proceedings of the 12th International Joint Conference on Biomedical Engineering Systems and Technologies*. doi: 10.5220/0007382603190325

Pujari, A.K. (2013). Data Mining Techniques

Tamvada, J.P. (2011). *Entrepreneurial Teams, Optimal Team Size, and Founder Exits.*

Walter, T. P., & Back, A. (2013). A Text Mining Approach to Evaluate Submissions to Crowdsourcing Contests. *46th Hawaii International Conference on System Sciences*. doi:10.1109/hicss.2013.64

Wooten, J., & Ulrich, K. T. (2011). Idea Generation and the Role of Feedback: Evidence from Field Experiments with Innovation Tournaments. *SSRN Electronic Journal.* https://doi.org/10.2139/ssrn.1838733

Yang, Y., & Pedersen, J. O. (1997, July). A comparative study on feature selection in text categorization. In Icml (Vol. 97, No. 412-420, p. 35).

Schiavone, F., Appio, F. P., & Arreola-Risa, A. (2021). Crowdsourcing and open innovation: a systematic literature review, an integrative framework, and future research directions. Journal of Open Innovation: Technology, Market, and Complexity, 7(2), 1-22.

Purdue University. (n.d.). Reference List: Basic Rules. Purdue OWL. https://owl.purdue.edu/owl/research_and_citation/apa_style/apa_formatting_and_style_guide/reference_list_basic_rules.html

American Psychological Association. (2016). A beginner's guide to crowdsourcing. Psychological Science Agenda. https://www.apa.org/science/about/psa/2016/06/changing-minds

Westland, J. C., & Mallapragada, G. (2011). Crowdsourcing: A review of literature and directions for future research. International Journal of Management Reviews, 13(3), 367-383

Yu, H., & Miao, C. (2021). A Jaccard Similarity-Based Model to Match Stakeholders for Collaboration in an Industry-Driven Portal. Proceedings, 74(1), 15. https://doi.org/10.3390/proceedings2021074015

Nilsson R, P˜na J, Bjorkegren J, Tegner J (2007). "Consistent Feature Selection for Pattern Recognition in Polynomial Time." The Journal of Machine Learning Research, 8, 612.

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. Artificial Intelligence, 97(1-2), 273-324

Chen, Y., & Li, Y. (2022). Mitigating the Multicollinearity Problem and Its Machine Learning Approach: A Review. Mathematics, 10(8), 1283. https://doi.org/10.3390/math10081283

Chee, F. (2015). Pearson's Product-Moment Correlation: Sample Analysis. ResearchGate. https://www.researchgate.net/publication/277324930_Pearson's_Product-Moment_Correlation_Sample_Analysis

Appendix

Appendix A

Top Sectors

Between 2008 and 2011, a global financial crisis occurred. The financial sector lost trust and the Dutch economy entered a recession. As the economy deteriorated, the government reduced its spending, businesses delayed their investments and innovation slowed down significantly. The top sectors were established to coordinate and stimulate innovation and investment across different domains. The top sectors allowed businesses representatitves, science and the government to work together to support innovation and knowledge sharing. The top sectors reflect the international reputation of Dutch knowledge and trade in domains such as energy, water, food production, technology, and so forth. The top sectors include a diversity of large firms, SMEs, start-ups and scale-ups. Over 90% of the top sectors are innovative SME entrepreneurs. In 2011, the following nine top sectors were formed: Agri & Food, Chemistry NL, Creative Industry, Energy, Health Holland (Life sciences & health), Logistics, Holland High Tech (High Tech Systems & Materials, Horticulture & Starting Materials, Water & Maritime, and Dutch digital delta (Team ICT).

The top sectors have one or more Top Consortia for knowledge and innovation (TKI). Entrepreneurs and scientists explore different manners to launch innovative products and innovative services to the market within a top consortium. This is done through fundamental research, experimental development, industrial research or a mix of the formerly mentioned. The top consortia ensure that the network is established, that knowledge is exchanged, and that the projects are managed.

The following shortly describes each TKI that is relevant to this research.

High Tech systems & Materials

High tech encompasses only cutting edge technologies. The high tech industry is moving toward miniaturisation, products with complex shapes, and multi-functional materials. According to Holland High Tech scientific developments in this area can be expected in 6 domains[30]. The first domain is technologies for sustainable energy production and storage.

[30] https://www.hollandhightech.nl/sites/www.hollandhightech.nl/files/inline-files/Roadmap%20High%20Tech%20Materials.pdf

The general theme of this domain is developments in materials that lead to innovations. The second domain is next generation engineering materials. These materials are lighter and stronger than previous generation materials. There is also a focus on nanotechnology, in which technology keeps its functionality in a much smaller size. The third domain is designer functional metamaterials. The materials are complex and have predetermined properties and functionalities. The fourth domain is sustainable materials. The development focuses on reducing the environmental impact of the materials life cycle. The fifth domain is the application of coatings and thin films to give materials different mechanical and functional properties. The sixth domain is the soft and bio-inspired materials, which will be covered in the next paragraph.

Chemistry & Materials

The chemical industry is concerned with converting raw materials into industrial chemicals. The TKI Chemie identifies four domains in which developments are expected[31]. The first domain is Chemistry of Advanced Materials. This domain covers the development of materials with the right functionality, thin films and coatings, and materials for sustainability. The second domain is Chemistry of life. It covers molecular entities for understanding, monitoring and improving health, molecular entities for understanding, monitoring and improving food. The third domain is Chemical Conversion, Process Technology and Synthesis. This domain is about making molecules efficiently, making molecules from biomass, and making functional molecules. The fourth domain is Chemical nanotechnology and devices. This domain covers well-being, energy efficiency and storage.

Digital & Internet

The Rijksdienst voor Ondernemend Nederland describes four themes in ICT research and innovation[32]. They call the first theme 'ICT one can rely on'. This theme is concerned with the security, vitality, and privacy in ICT. The second theme is called 'ICT systems for monitoring and control'. The third theme is called 'ICT for a connected world'. This theme relates to the information chains that are relevant for innovation. Key elements in this theme are standardisation, open data, and services. This has the effect of making supply chains more efficient, available databases for the public, and reduced obstacles in business operations,

[31] https://hollandchemistry.nl/wp-content/uploads/2019/02/Executive-summaries-roadmaps-2.pdf

[32] https://www.rvo.nl/sites/default/files/Roadmap%2016%20ICT%20for%20the%20top%20sectors%20Topsector%20HTSM.pdf

respectively. The fourth theme is called ‘Data, data, data’. This theme concerns data and content exploration.

Energy & Sustainability

The topic of energy and sustainability aims to accelerate the energy transition from fossil energy to sustainable energy. In addition, an energy regime free of CO2 emissions is envisioned. To reach these goals several innovation programmes are formulated. The first programme focuses on electricity and sustainable energy generation. The second programme focuses on urban areas. The goal is to have CO2 neutral urban areas. The third programme focuses on industry and reaching a CO2 neutral industry. The fourth programme focuses on mobility and eliminating emission from transportation of people and goods. The fifth programme focuses on agriculture and innovations to achieve CO2 neutral agriculture.

Transport & Automotive

Transport & automotive is about the transportation of people and goods. The top sector logistics, which covers transport and automotive prioritises three areas of innovation.[33] The first area is sustainable logistics. Sustainability in logistics is realised through innovations in fuels, in applying new technologies in vehicles, and in logistical optimisation & behavioural changes. The second area covers data driven logistics. Using sensors and computers communicating with, traffic and transport can become more efficient, smarter, and safer. The third area is about supply chain management. This area covers research in control towers, different forms of cooperation, and planning solutions on operational, tactical, and strategic level.

Building & Physics

Construction is a crucial contributor to society. TNO focuses on a few aspects of construction.[34] Sustainable buildings are buildings that generate their own energy through solar power and other sustainable energy sources. Digitisation in construction can lead to optimisation and greater efficiency. Managing, monitoring, and maintaining building and constructions can be assisted using sensors and internet networks.

High Tech to Feed the World

[33] https://topsectorlogistiek.nl/wptop/wp-content/uploads/2020/02/Actieagenda-2020-2023.pdf
[34] https://www.tno.nl/en/focus-areas/buildings-infrastructure-maritime/

High tech to feed the world is a crossover between high tech systems & materials and ICT and agriculture and food. Elements of high tech systems & materials and ICT are implemented in the agriculture and food sectors to improve agriculture processes and food production. Foodvalley[35] names four fields of innovation. The first field is about the protein shift from animal proteins to vegetable proteins. The second field is about circular agriculture. Circular agriculture means to keep residuals of agricultural biomass and food processing within the food system as renewable resources.[36] The third field is about food and health. It is about finding innovations that help people age healthily. The last field is about smart and digital technology for health. It is about bringing the latest technologies to the agriculture & food sector, this ranges from personalised nutritional advice to targeted crop breeding.

Nature 2.0

Nature 2.0 is a community of explorers of unforeseen possibilities in exponential growing technology.[37] Nature 2.0 believes that systemchange is not prevented by technological possibilities, but by frozen mindsets. There is, therefore, a need for new narratives and paradigm shifts to advance technological developments.

Life Sciences & Health

The top sector life sciences and health have laid out ten roadmaps to address life science and health.[38] The first roadmap relates to molecular diagnostics, which focuses on developing biomarkers into validated molecular diagnostics for clinical use. The second roadmap concerns developing imaging applications for diagnosis, prognosis, and monitoring. The third roadmap is about homecare and self-management. It is about developing and implementing technologies that can be used to manage one's own health. The fourth roadmap covers regenerative medicine. The fifth roadmap is about pharmacotherapy, which involves the development of personalised medication to cure or prevent diseases. The sixth roadmap is about developing solutions concerning health by combining knowledge from the human, veterinary, and agriculture domains. This is named as One Health. The seventh roadmap

[35] https://www.foodvalley.nl/fields-of-innovation/
[36] https://www.wur.nl/upload_mm/6/e/e/07a9b802-0bbe-4a7e-a2cb-597236a0d359_Circular%20agriculture%20-%20A%20new%20perspective%20for%20Dutch%20agriculture.pdf
[37] https://nature2.ooo/
[38] https://www.health-holland.com/public/downloads/kia-kic/knowledge-and-innovation-agenda-2018-2021.pdf

concerns specialised nutrition for the prevention or curing of diseases. The eight roadmap is about health technology assessment, individual functioning & quality of life. It is concerned with development of methods and knowledge for health technology assessments which impacts individuals. The ninth roadmap is about enabling technologies and infrastructure. It is about developing and offering expertise and infrastructure in cutting-edge molecular life science, and is strongly intertwined with other top sectors. The last roadmap is concerned with global health and developing solutions to diseases that affect a large part of the world

Finance & Technology

Finance technology is the innovation that aims to compete with traditional financial methods in the delivery of financial services. Finance technology is used for insurance, trading, banking services, and risk management.[39] It uses Artificial intelligence, big data, blockchain, and other automation processes for its services and is thus closely related to digital & internet.

Cluster 1: High tech systems & Materials

- Sustainable energy production and storage
- Next generation engineering materials.
- Designer functional metamaterials.
- Sustainable materials
- Coatings and thin films

Cluster 2: Chemics & Materials

- Chemistry of Advanced Materials
- Chemistry of life
- Chemical Conversion, Process Technology and Synthesis
- Chemical nanotechnology and devices

Cluster 3: Digital & Internet

- ICT one can rely on
- ICT systems for monitoring and controlling
- ICT for a connected world
- Data, data, data

Cluster 4: Energy & Sustainability

[39] https://en.wikipedia.org/wiki/Financial_technology

- Sustainable power
- CO2 neutral urban areas
- CO2 neutral industry
- CO2 neutral agriculture
- CO2 neutral transport

Cluster 5: Transport & Automotive

- Sustainability in logistics
- Data driven logistics
- Supply chain management

Cluster 6: Building & Physics

- Sustainable buildings
- Digitisation in construction

Cluster 7: High Tech to Feed the World

- Protein shift
- Circular agriculture. Circular agriculture means to keep residuals of agricultural
- Food and health
- Smart and digital technology for agriculture and food

Cluster 8: Nature 2.0

- New narratives
- Paradigm shift

Cluster 9: Life Sciences & Health

- Molecular diagnostics
- Imaging applications
- Homecare and self-management
- Regenerative medicine
- Pharmacotherapy
- One Health
- Specialised nutrition

Cluster 10: Finance & Technology

- Insurance
- Trading
- Banking services

Appendix B

```
from tinydb import TinyDB, Query
import pandas as pd
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.feature_selection import RFECV, f_classif
from sklearn.model_selection import GridSearchCV,  StratifiedKFold
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, f1_score
from scipy.stats import zscore
import json
from scipy.stats import pearsonr
from multiprocessing.pool import Pool
import eli5
from eli5.sklearn import PermutationImportance
import matplotlib.pyplot as plt
import seaborn as sns
from random import randrange
sns.set(style='darkgrid')


pd.options.mode.chained_assignment = None  # default='warn'

#####with contest 4 and without contest 4####### STANDARDISE ALL DATA FIrST

# Initialize models
models = [
        RandomForestClassifier(random_state=15),
        LogisticRegression(max_iter=100000, random_state=15, penalty='l2'),
        SVC(random_state=15, gamma='auto'),
        DecisionTreeClassifier(random_state=15),
        KNeighborsClassifier(),
        GaussianNB(),
        GradientBoostingClassifier(random_state=15),
        MLPClassifier(max_iter=100000, random_state=15),
    ]
```

```
# Define the parameter grids for each model
param_grids = {
    RandomForestClassifier: {
        'n_estimators': [5, 10, 15],
        'max_depth': [10, 15, 20],
        'criterion': ['gini', 'entropy'],
        'class_weight': [None, 'balanced'],
    },
    LogisticRegression: {
        'C': [0.05, 0.1, 0.2, 0.3],
        'solver': ['lbfgs', 'sag', 'newton-cg'],
        'class_weight': [None, 'balanced'],
    },
    SVC: {
        'C': [0.1, 0.20, 0.30, 0.4],
        'kernel': ['linear', 'rbf'],
        'class_weight': [None, 'balanced']
    },
    DecisionTreeClassifier: {
        'max_depth': [ 2, 5, 15, 20],
        'criterion': ['gini', 'entropy'],
        'class_weight': [None, 'balanced']
    },
    KNeighborsClassifier: {
        'n_neighbors': [2, 4, 6, 8],
        'weights': ['uniform', 'distance'],
        'p': [1, 2],
    },
    GaussianNB: {
    },
    MLPClassifier: {
        'hidden_layer_sizes': [(1,), (2,), (3,)],
        'alpha': [0.05, 0.1, 0.2],
        'learning_rate_init': [0.001, 0.01, 0.1],
        'activation': ['identity', 'logistic', 'tanh', 'relu'],
        'solver': ['sgd', 'adam'], #'lbfgs'
    },
    GradientBoostingClassifier: {
        'n_estimators': [10, 100, 1000],
```

```
        'learning_rate': [0.1, 0.01, 0.001],
        'max_depth': [3, 5, 7]
    }
}

def conf_matrix(y_test, y_predict, target):
    """
    Plot a confusion matrix of the model
    """
    if target == 'finalist':
        categories = ['Finalist', 'Participant + Top_40']
    if target == 'participant':
        categories = ['Top_40 + Finalist', 'Participant']
    if target == 'rank':
        categories = ['Participant', 'Top_40', 'Finalist']
    plt.figure(figsize=(7, 5))
    sns.heatmap(confusion_matrix(y_test, y_predict),
                annot=True, fmt='d', cbar_kws={'shrink': .5},
                xticklabels=categories,yticklabels=categories)
    plt.xlabel('Predicted', fontsize=15)
    plt.ylabel('Actual', fontsize=15)
    return plt

def kfolds_fun(model, X, y, columns, j, score, train_index, test_index):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
    X_train, X_test = pd.DataFrame(X_train, columns=columns), pd.DataFrame(X_test, columns=columns)
    X_train['media'] = pd.Categorical(X_train['media'])
    X_train['Highest_edu'] = pd.Categorical(X_train['Highest_edu'])
    X_test['media'] = pd.Categorical(X_test['media'])
    X_test['Highest_edu'] = pd.Categorical(X_test['Highest_edu'])
    # Perform RFE for each model
    round_results = {}

    rkf = StratifiedKFold(n_splits=5, random_state=15, shuffle=True)
    # Initialize the RFE object with the current model and number of features

    if model.__class__ is RandomForestClassifier or model.__class__ is SVC or model.__class__ is
KNeighborsClassifier or model.__class__ is MLPClassifier or model.__class__ is GaussianNB:
```

```
        rfe = RFECV(PermutationImportance(model,  scoring=score, n_iter=5, random_state=15, cv=rkf), step=1,
cv=rkf, scoring=score)
    else:
        rfe = RFECV(model, step=1, cv=rkf, scoring=score)
    rfe = rfe.fit(X_train, y_train)

    # Create the parameter grid for the current model
    param_grid = param_grids[model.__class__]

    # Create the grid search object with the RFE object and parameter grid
    grid_search = GridSearchCV(model, param_grid, cv=rkf, scoring=score)



    # Fit the grid search object to the data
    grid_search.fit(X_train.iloc[:, rfe.get_support()], y_train)

    # Get the best RFE object from the grid search
    selector = grid_search.best_estimator_

    RFEcolums = X_train.iloc[:, rfe.get_support()].columns
    perm = PermutationImportance(selector, random_state=15, scoring=score).fit(X_test.iloc[:,
rfe.get_support()], y_test)
    featimps = eli5.format_as_dict(eli5.explain_weights(perm))['feature_importances']['importances']
    for feat in featimps:
        feat_ind = int(feat['feature'].replace('x', ''))
        feat_name = RFEcolums[feat_ind]
        round_results[feat_name] = f"{feat['weight']:.2e}({feat['std']:.2e})"

    # test model
    y_pred = selector.predict(X_test.iloc[:, rfe.get_support()])

    ## Create the confusion matrix
    # confmat = conf_matrix(y_test, y_pred, "rank")
    #
confmat.savefig(f'C:/Users/kluit/Desktop/discordbot/utchallenge2/matrixes/confmtrx_rank_{model}_{randrang
e(1000000)}', bbox_inches='tight')

    return {'model': f'{model}', 'params': grid_search.best_params_, 'fold': j, 'f1 macro': f1_score(y_test, y_pred,
average="macro"), 'accuracy': accuracy_score(y_test, y_pred), 'features': round_results}
```

```
def somefunc(df, model, target, db, score, splitting):
    if splitting == 7 and target =="rank":
        return
    # Initialize the dictionary to store the significance levels
    #####with contest 4 and without contest 4#######
    X = df.drop(columns=['rank', 'contest', 'Project ID', 'finalist', 'participant'])
    columns = X.columns.tolist()
    X = X.to_numpy()
    y = df[target]
    obs = len(y)
    # Setup cross validation
    splits = splitting
    kf = StratifiedKFold(n_splits=splits, random_state=15, shuffle=True)
    with Pool() as pool:
        args = [(model, X, y, columns, j, score, train_index, test_index) for j, (train_index, test_index) in
enumerate(kf.split(X, y), start=1)]
        # temper = [x for x in pool.starmap(kfolds_fun, args) if x]
        db.insert_multiple([x for x in pool.starmap(kfolds_fun, args) if x])


def kfolds_fun2(model, X, y, columns, j, score, train_index, test_index):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
    X_train, X_test = pd.DataFrame(X_train, columns=columns), pd.DataFrame(X_test, columns=columns)
    X_train['Highest_edu'] = pd.Categorical(X_train['Highest_edu'])
    X_test['Highest_edu'] = pd.Categorical(X_test['Highest_edu'])
    X_train['media'] = pd.Categorical(X_train['media'])
    X_test['media'] = pd.Categorical(X_test['media'])
    # Perform RFE for each model
    round_results = {}

    rkf = StratifiedKFold(n_splits=5, random_state=15, shuffle=True)
    # Create the parameter grid for the current model
    param_grid = param_grids[model.__class__]

    # Create the grid search object with the RFE object and parameter grid
    grid_search = GridSearchCV(model, param_grid, cv=rkf, scoring=score)
```

```
    # Fit the grid search object to the data
    grid_search.fit(X_train, y_train)

    # Get the best RFE object from the grid search
    selector = grid_search.best_estimator_
    # print(grid_search.best_estimator_)

    RFEcolums = X_train.columns
    perm = PermutationImportance(selector, random_state=15, scoring=score, cv='prefit').fit(X_test, y_test)
    featimps = eli5.format_as_dict(eli5.explain_weights(perm))['feature_importances']['importances']
    for feat in featimps:
        feat_ind = int(feat['feature'].replace('x', ''))
        feat_name = RFEcolums[feat_ind]
        round_results[feat_name] = f"{feat['weight']:.2e}({feat['std']:.2e})"

    # test model
    y_pred = selector.predict(X_test)

    ## create the confusion matrix
    # confmat = conf_matrix(y_test, y_pred, "rank")
    #
confmat.savefig(f'C:/Users/kluit/Desktop/discordbot/utchallenge2/matrixes/confmtrx_rank_{model}_{randrang
e(1000000)}', bbox_inches='tight')

    return {'model': f'{model}', 'params': grid_search.best_params_, 'fold': j, 'f1 macro': f1_score(y_test, y_pred,
average="macro"), 'accuracy': accuracy_score(y_test, y_pred), 'features': round_results}



def somefunc2(df, model, target, db, score, splitting):
    if splitting == 7 and target == "rank":
        return
    # Initialize the dictionary to store the significance levels
    #####with contest 4 and without contest 4#######
    X = df.drop(columns=['rank', 'contest', 'Project ID', 'finalist', 'participant'])
    columns = X.columns.tolist()
    X = X.to_numpy()
    y = df[target]
    obs = len(y)
    # Setup cross validation
```

```
    splits = splitting
    kf = StratifiedKFold(n_splits=splits, random_state=15, shuffle=True)
    with Pool() as pool:
        args = [(model, X, y, columns, j, score, train_index, test_index) for j, (train_index, test_index) in
enumerate(kf.split(X, y), start=1)]
        # temper = [x for x in pool.starmap(kfolds_fun2, args) if x]
        db.insert_multiple([x for x in pool.starmap(kfolds_fun2, args) if x])



def correlation_calc(df):
    df = df.drop(columns=['rank', 'contest', 'Project ID', 'finalist', 'participant'])

    # Get the column names
    columns = df.columns

    corr_colls= []
    # Loop over all pairs of columns
    for i in range(len(columns)):
        for j in range(i+1, len(columns)):
            # Calculate the correlation and P-value between column i and column j
            corr, p_value = pearsonr(df[columns[i]], df[columns[j]])
            if p_value < 0.05 and (corr > 0.7):
                corr_colls.append((columns[i],columns[j]))

    return corr_colls

def invert(data):
    return 1 - data



def main():
    # Load data
    df = pd.read_excel('C:/Users/kluit/Desktop/discordbot/utchallenge2/TheData.xlsx')
    df['rank'] = df['rank'].astype(pd.CategoricalDtype(categories=['Participant', 'Top_40', 'Finalist'],
ordered=True))
    df['media'] = pd.Categorical(df['media'])
    df['Highest_edu'] = pd.Categorical(df['Highest_edu'])
    df['finalist'] = df['rank'].str.contains('Finalist').astype(int).where(df['rank'].str.contains('Finalist'), 0)
    df['finalist'] = df['finalist'].apply(invert)
    df['finalist'] = pd.Categorical(df['finalist'])
```

```
    df['participant'] = df['rank'].str.contains('Participant').astype(int).where(df['rank'].str.contains('Participant'), 0)
    df['participant'] = pd.Categorical(df['participant'])
    df = df.drop(columns=['relevance'])


    # process dataset with contests 1 2 3
    df123 = df[df['contest'].isin([1,2,3])]
    df123columnsnorm = ['team_size', 'coach_coverage', 'Jaccard',
                        'total_word_count', 'total_sentences', 'DaleChall', 'FRE', 'highest_similarity']
    for col in df123columnsnorm:
        zscorecol = df123[col]
        df123[col] = zscore(zscorecol)
    # calculate correlations between variables and eliminate one of correlated variables
    correlated_vars = correlation_calc(df123)
    df123_nocorr = df123.drop(columns=[x[1] for x in correlated_vars])


    # process dataset without coach variables
    all_df_no_coach = df.drop(columns=['coach_coverage', 'nr_coaches', 'Jaccard'])
    all_df_columnsnorm = ['team_size',
                        'total_word_count', 'total_sentences', 'DaleChall', 'FRE', 'highest_similarity']
    for col in all_df_columnsnorm:
        all_df_no_coach[col] = zscore(all_df_no_coach[col])
    # calculate correlations between variables and eliminate one of correlated variables
    correlated_vars = correlation_calc(all_df_no_coach)
    all_df_no_coach_nocorr = all_df_no_coach.drop(columns=[x[1] for x in correlated_vars])

    for splitser in [5, 7]:
        for target in ['rank', 'participant', 'finalist']:
            for score in ['f1_macro']:#'accuracy', 'precision_macro', 'recall_macro']: #f1_macro
                with open(f'C:/Users/kluit/Desktop/discordbot/utchallenge2/db0505-{target}-{score}-split-
{splitser}.json', "w") as f:
                    pass
                for model in models:
                    User = Query()
                    dbrank = TinyDB(f'C:/Users/kluit/Desktop/discordbot/utchallenge2/db0505-{target}-{score}-split-
{splitser}.json')
                    dbrank1 = dbrank.table('164 obs - no correlation - yes rfe')
                    dbrank2 = dbrank.table('164 obs - no correlation - no rfe')
                    dbrank3 = dbrank.table('164 obs - yes correlation - yes rfe')
```

```
        dbrank4 = dbrank.table('164 obs - yes correlation - no rfe')

        dbrank5 = dbrank.table('248 obs - no correlation - yes rfe')
        dbrank6 = dbrank.table('248 obs - no correlation - no rfe')
        dbrank7 = dbrank.table('248 obs - yes correlation - yes rfe')
        dbrank8 = dbrank.table('248 obs - yes correlation - no rfe')

        somefunc(df123_nocorr, model, target, dbrank1, score, splitser)
        somefunc2(df123_nocorr, model, target, dbrank2, score, splitser)
        somefunc(df123, model, target, dbrank3, score, splitser)
        somefunc2(df123, model, target, dbrank4, score, splitser)

        somefunc(all_df_no_coach_nocorr, model, target, dbrank5, score, splitser)
        somefunc2(all_df_no_coach_nocorr, model, target, dbrank6, score, splitser)
        somefunc(all_df_no_coach, model, target, dbrank7, score, splitser)
        somefunc2(all_df_no_coach, model, target, dbrank8, score, splitser)

if __name__ == "__main__":
    main()
```