Online Technical Assessment Classification Based on Personalization

Lim Zheng Yu [#1], Liyana Shuib [#3]
[#] *Faculty of Computer Science and Information Technology, Universiti Malaya*
50603 Kuala Lumpur, Malaysia
[1] u2102809@siswa.um.edu.my
[3] liyanashuib@um.edu.my

Abstract **– Online technical assessments are slowly becoming the thing of the future, especially after COVID-19. Institutions are conducting more online assessments, but this is not a viable solution as traditional assessments have always been a disadvantage for most students. There is an urgency to curate a multilabel classification model to suggest suitable types of online assessments for students taking technical subjects like programming. Six machine learning models were trained, and their performances were compared on various metrics. The results show that the baseline model of gradient boosting is the best, returning a Hamming loss of 0.228, ROC AUC of 0.693, F1-Score, precision, and recall with a macro average of 0.83, 0.82, and 0.84 respectively, and an accuracy of 0.238.**

Keywords **– Online technical assessments; Multilabel classification; Gradient boosting; VAK; Learning styles; Personalization; Programming; Coding.**

I. Introduction

Online assessments have become increasingly commonplace throughout the globe, with the COVID-19 pandemic confirming this trend. The reason is the pandemic has significantly hindered the progress of education, forcing institutions to shift their teaching to online platforms (Abduh, 2021), such as Microsoft Teams and Google Classroom. To further prove it, MOOC registrations are becoming more and more prevalent, with enrolments sometimes going up to three or four times higher than the previous year in the period of 2012 to 2022 (Papadakis, 2023). This in turn means that eventually, students who are in the field of Computer Science, Engineering, and other tech-related fields will have a lot of their technical subjects assessed online as well.

Here comes the problem, a lot of existing programming certification examinations, such as such as the Sun's Java Certification Examination and Novell's certification examinations are multiple-choice (Roberts & Verbyla, 2003), serving no significant purpose in evaluating programming students. Traditional assessments are also standardized, making them "indirect and inauthentic" (Bailey, 1997), meaning that they are effective in measuring students' capabilities at one point in time but ineffective in informing student progression (Dikli, 2003).

The VAK (Visual-Auditory-Kinaesthetic) Learning Styles Model (Barbe et al., 1979) describes that everyone is different in showcasing their skills and absorbing information. Visual learners typically learn through observation; therefore, they learn effectively by using graphs, posters, and other visual tools. Auditory learners learn by listening, and hence, they gather information through discussions, lectures, and stories. Finally, kinesthetic learners learn best by doing, and therefore, learn through physical activities and touch.

According to this model, it would be unwise to assume everyone is similar in terms of their abilities. The same can be said for students' performances in assessments. Not everyone does well in certain exam formats; some students may be excellent in memorization, while some display exceptional critical thinking skills instead. In the context of technical subjects, assessments would be futile if all programming tests were conducted in essay or multiple-choice format. Certain formats do not properly assess students, and it is certainly unfair to assume that a student who performs well in assessments that involve heavy memorization is better than a student who does badly in the same assessment. Hence, in this context, it is suggested that it would be fairer to evaluate every student differently in online technical assessments.

II. Literature Review

Throughout the years, many studies have been conducted about learning objects, learning styles, online students' performance, programming test performance, and so on. Yet, there is very limited research on the amalgamation of said subjects, meaning that to date, there may not be a classification model for online technical assessments based on students' personalities. Discussed below are some relevant literature.

Garcia et al. (2005) realized the importance of profiling students based on their learning styles in computer-based education. He highlighted that an adaptive and personalized online education platform is highly sought-after to accommodate various types of learners. The goal of their research is to identify students' learning styles based on the Felder-Silverman model using Bayesian Networks. Eventually, they achieved 100% accuracy for the understanding dimension, 80% for the perception dimension, and 80% for the processing dimension. However, the input of their research is too few, a mere 10 computer science and engineering students, and hence, the results of the research should be taken with a grain of salt.

Ulloa-Cazarez et al. (2018) conducted an intriguing study to predict online students' performance using genetic programming. The dataset consisted of 245 students' grade records and log records. Their study was conducted to predict the final grades of online students based on their grades from the beginning stages of the course. The results of the genetic programming model returned a Mean Absolute Residuals (MAR) of 14.38 and Median Absolute Residuals (MdAR) of 9.80, while the results of the linear regression model were MAR of 17.49 and MdAR of 16.53. The genetic programming model was significantly better than the linear regression model by a margin of 90% confidence level on the training data, and 99% confidence level on the test data.

Cetinkaya et al. (2023) did research on 600 secondary students in Turkey to recognize students who were talented in computer programming such that they could be given exposure to programming fields that are considered their strengths. Besides the spatial test that was assigned to the students, data was also retrieved from Code.org to assist with model training. Support vector machine (SVM), decision trees (DT), k-nearest neighbours (KNN), and quadratic discriminant (QD) were the machine learning algorithms used to classify students' programming skills. The results show that Cubic SVM was the best in accurately classifying their abilities, achieving 94.8% accuracy, followed by fine DT classifiers (89.0%), QD (84.3%), and medium KNN (80.8%). The best-performing algorithm of SVM also did well in other metrics: 91.5% Kappa, 93.6% precision, 94.8% recall, and 94.1% F-Score. Yet, this study was unable to solve our problem of classifying online technical assessments.

Dema (2021) invited 16 female and 24 male students enrolled in the program Bachelor of Engineering in Information Technology from the Royal University of Bhutan to conduct his study. He noticed that not only many freshmen were struggling with their C programming module, but the faculty was also ineffective in teaching it, leading to high rates of failure. The study aims to determine Neil Fleming's VARK learning styles of these first-year students and to convey the findings to the faculty such that improvements can be made in teaching. It was found the most dominant learning style is kinaesthetic, constituting 42% of the students. Unfortunately, there were many limitations to the study, including no predictive modeling was employed, no personalization methods are being used, proper sampling methods were not utilised, and hence, the findings cannot be generalized for all students in the tech-related fields. Only descriptive statistics was used to analyse the findings and the study has a lot of wasted potential.

The research of Ocepek et al. (2013) is aimed at curating an adaptive learning recommender system for 272 undergraduate students from the University of Ljubljana to suggest the appropriate multimedia type (or learning object) based on their learning styles. It incorporates a variety of learning styles: Kolb's Rancourt's, VAK, and the hemispheric to profile a student's characteristic in detail. This was achieved using a multi-target regression tree, predicting the usage of multimedia types. By combining the learning styles, they were able to determine the characteristics of students in a robust and unbiased manner. The findings from the tree were, textual content with colour content proved to the best multimedia type, and hence all subjects should include textual content in their teachings. The findings are not definitive, however, since there may be scenarios where the educator prepares only video and audio lectures. Other findings based on the learning styles are that students who are non-visual, assimilators, left-hemispheric and empirical are the types of people who prefer audio learning material, on the other hand, visual, rational, non-right-hemispheric and divergers are least associated with audio materials. The limitations of this research are that the results of the tree were not evaluated and is only used to inform findings, and their classification targets are on multimedia types instead of assessment types.

Seyal et al. (2015) pioneered a study relating students' Kolb learning style and their performance in programming courses. The targeted students were first-year Bachelor of Internet Computing students. The students first-year programming concept module results were obtained and assigned a Kolb's Learning Style Questionnaire to identify their learning style. The study suggests that the students were mostly of the convergers and assimilators learning type, their learning styles were found to have an impact on their programming test scores, and gender played a role in their programming test scores. Most importantly, the relationship between programming results and learning styles was concluded using the Chi-square test of independence, with the study showing a difference between convergers and assimilators on their programming results: 39% of convergers passed as opposed to 15% who are assimilators. It is stated that the limitations of their study include having a small sample size, and the conducted survey was only at a single point in time, eventually leading up to the problem of generalization due to the specifics of the settings.

III. Problem Statement

From the previous section, it is apparent that there is limited study on our proposed topic: classification of online technical assessments based on students' VAK learning styles. The relevant literature mostly discusses the domain of learning styles or learning objects with classification models or online performance. They focus on performance instead of classifying the most suitable types of assessments for technical subjects and do not include personalization methods.

This study has three main objectives:

1. To develop a classification model of online technical assessments based on personalization.

2. To evaluate the performance of the classification model of online technical assessments based on personalization.
3. To develop a data product of the classification model of online technical assessments based on personalization.

IV. Methodology

The Cross Industry Standard Process for Data Mining, known as CRISP-DM, is the most widespread methodology for data-related projects, and it is used as the methodology for this data science project. It has six phases: business understanding, data understanding, data preparation, modelling, evaluation, and deployment. As shown in Figure 1, the flow is not one-way, rather, there are backward flows between phases as a lot of revisions are done in this methodology to clean the dataset to fit our requirements.

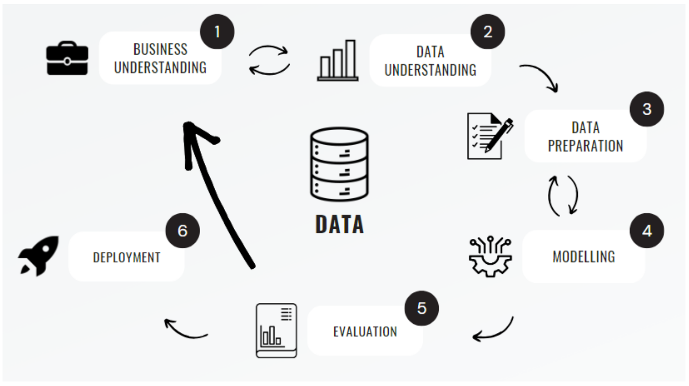


Figure 1: CRISP-DM Methodology

A. Business Understanding

The first phase of this methodology helps us identify our project objectives and requirements. It begins with collecting and obtaining data and understanding the dataset briefly to see if the dataset fits our use case. The data used for this project was collected using Google Forms from January 2021 to November 2023. The survey form contains four parts: personal details, learning object preference, learning style awareness, and learning style test.

Then, it is followed by determining our business objectives which was already identified in section III. Problem Statement, and finally to produce a project plan.(García et al., 2005)

B. Data Understanding

The survey form has 103 questions, resulting in 103 variables in the initial dataset with 1052 rows and 836 missing values, a column 'dominant_VAK' is added as a new variable before proceeding with other works.

To further understand the dataset in detail, the 'ProfileReport' method from the 'ydata-profiling' package is invoked, and a html file describing every column is produced. Each column's description included the column name, number and percentage of distinct values, number and percentage of missing values, memory size, and a visualization showing its distribution based on the type of data. All columns in the dataset were originally read as text data. Hence, all the visualizations were word clouds.

Here is a summary on the columns of the dataset:

The first 13 columns of the dataset consisted of demographics and other information, such as Timestamp, Email Address, Gender, Level of Study, Field of Study, Household Income, Preferred Social Media Platform, and Preferred Communication Platform.

The following 13 columns are respondents' learning object preferences and they are required to answer from a scale of 'Not at All', 'Not Really', 'Undecided', 'Somewhat', and 'Very Much'.

The next 40 columns are online instructional strategies/ assessments preferences based on the same scale.

The next column is target towards technical subjects, and respondents are required to manually input their online instructional strategies/ assessments preferences. Some additional cleaning is needed on this column.

Columns 68 to 70 describe respondents' understanding of learning styles and the outputs are 'Yes' and 'No'.

Columns 71 to 100 are learning styles test questions. A scenario is provided, and respondents are required to select one response. These responses are then used to calculate the dominant_VAK (column 104) of the respondents.

Column 101 is a text field gathering feedback on the survey, columns 102 and 103 are the faculty and department of the respondents respectively, both are text input fields.

After some cleaning in the data preparation phase, we revised the dataset and found out there were new issues. Some of the variables were greatly imbalanced as shown in Figure 2. There were also issues with the multiple selection questions in Google Forms. For example, survey respondents who chose YouTube as their preferred social media platform and survey respondents who chose both YouTube and Instagram as their selections would result two different outputs. Eventually, there are 59 respondents who chose only YouTube, 35 respondents chose both Instagram and YouTube, 41 respondents chose Facebook, Instagram, and YouTube, when there is a total of 59 + 35 + 41 = 135 respondents who chose YouTube from these 3 outputs. This resulted in many unique outputs and extra cleaning is needed. The Google survey form also accepts some text inputs from respondents, resulting in very inconsistent and dirty data for those columns.

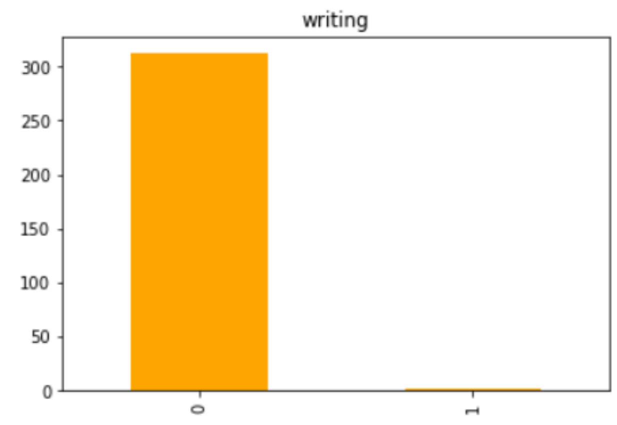


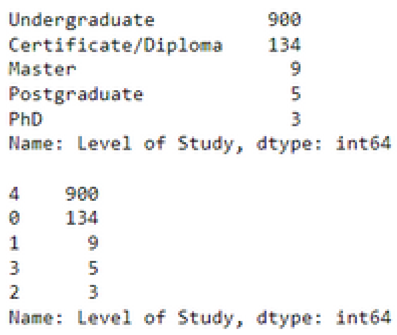
Figure 2: Imbalanced data in 'writing' column

C. Data Preparation

This phase involves many processes, including cleaning, modifying, and transforming data into a format that would help us to fit the data into our models.

1) Missing Values: Some of the columns that contain missing values are 'Faculty', 'Department', 'Institutions', and 'Please share any comments or suggestions related to this issue. Thank You'. Coincidentally, they are not necessary for our analysis as they are deemed irrelevant to our study. These columns are therefore dropped, and no data imputation is needed.

2) LabelEncoder: As mentioned in section IV. Methodology subsection B. Data Understanding, all columns in the dataset are text data. The 'LabelEncoder' method from the scikit-learn library was used to create a function named 'LabelEncoding' which accepts the parameter 'column'. The function prints the distribution of the column using the 'value_counts' method before invoking 'LabelEncoder' and prints out the distribution after invoking. Here is an example on the 'Level of Study' column:



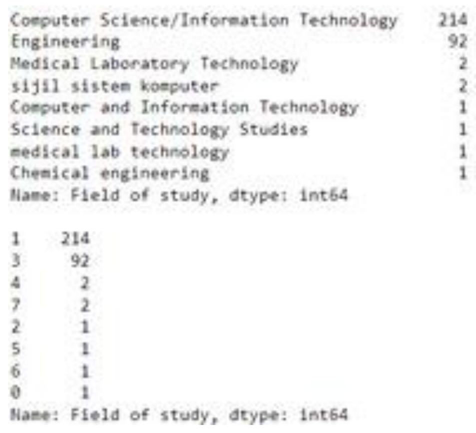
```
Undergraduate          900
Certificate/Diploma    134
Master                   9
Postgraduate             5
PhD                      3
Name: Level of Study, dtype: int64

4    900
0    134
1      9
3      5
2      3
Name: Level of Study, dtype: int64
```

Figure 3: Output of LabelEncoding("Level of Study")

3) Extract Relevant Study Fields: The 'Field of study' column is useful in identifying our intended target. There are also many unique values in this column. Hence, the values in this column are cross-checked with a list named 'relevant_fields', which contain terms such as 'computer', 'engineering', 'tech', 'jurutera', and 'komputer'. If the values in the column matches any of the values in the 'relevant_fields' list, they are then kept into the data frame.

Then, the 'Field of study' column is passed into the 'LabelEncoding' function.



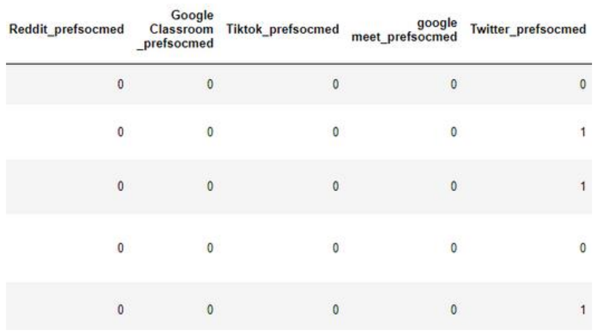
```
Computer Science/Information Technology    214
Engineering                                 92
Medical Laboratory Technology                2
sijil sistem komputer                        2
Computer and Information Technology          1
Science and Technology Studies               1
medical lab technology                       1
Chemical engineering                         1
Name: Field of study, dtype: int64

1    214
3     92
4      2
7      2
2      1
5      1
6      1
0      1
Name: Field of study, dtype: int64
```

Figure 4: Output of LabelEncoding("Field of study")

We only have a subset of the original dataset to work with left, only 314 rows remain.

4) Cleaning Multiple Choice Selections: To solve the problem mentioned in section IV. Methodology subsection B. Data Understanding, the function 'one_hot_encode_multiple_choice' is created. This function produces a set which contains all unique values in the column. The set is type casted as a list, and the list is then iterated through to create new columns indicating whether that entry has that value. If the column of that entry has a substring that matches the list, the newly created column will have a value of 1, otherwise 0. This process is akin to one-hot-encoding.



Reddit_prefsocmed	Google Classroom _prefsocmed	Tiktok_prefsocmed	google meet_prefsocmed	Twitter_prefsocmed
0	0	0	0	0
0	0	0	0	1
0	0	0	0	1
0	0	0	0	0
0	0	0	0	1

Figure 5: Example Output of one_hot_encode_multiple_choice

5) Cleaning Manual Inputs: Some of the columns have data which were manually input by respondents. Example outputs from the 'Online Instructional Strategies/ Assessment Preference for Technical Subjects' column are 'YOUTUBE VIDEO', 'YouTube video', 'live lecture and youtube video', 'Youtube Video and Live Lecture'. With the help of the Natural Language Toolkit (NLTK), the functions 'remove_stopwords' and 'lemmatize_text' are created.

'remove_stopwords' removes stop words if it exists in the text value. Stop words are commonly used words in any language. Words like "how" and "to" are considered stop words in English and they do not bring out much value. By removing them, we can focus on other words which have more valuable meaning.

Lemmatization is also done to process the text data. It is the process of grouping together words in different forms. For example, 'better' and 'good' or 'rocks' and 'rock' are words that will be classified together through lemmatization. This process is done in Python by instantiating a 'WordNetLemmatizer' object and it is processed for each word in the selected column.

6) Imbalanced Data: From a simple look of Figure 3, it is apparent that this column has imbalanced data. The easy workaround to the problem is to compress the minority levels into one level in a way which makes sense. In this context, the minority levels of 'Master', 'Postgraduate' and 'PhD' are compressed or grouped together into the level or label of 'Postgraduate'.

D. Modelling

The 11 target variables used in the modelling phase are:

- _Written assignment_
- _Case Study_
- _Real Time Online Exam_
- _Individual Project/Assignment_
- _Group Project/Assignment_
- _Online Quiz/Test – MCQ_
- _Online Quiz/Test – Essay_
- _Online Quiz/Test – Open Book_
- _Peer Review Assessment Live Presentation_
- _Recorded Presentation_
- _Portfolio_

The name of the mentioned variables begins with the prefix "6. Online Instructional Strategies/Assessment". This makes the problem a multilabel classification problem, with 1 stating that the particular online technical assessment is suitable and 0 meaning unsuitable.

The 'train_test_split' method from scikit learn is used to split the final processed data frame into train and test sets, with a test_size of 0.2 and random_state of 2024. The shape of the input data is 314 rows and 115 columns, while the output data is 314 rows and 11 columns.

Six machine learning algorithms from scikit learn were used in this study, namely gradient boosting (GB), Extreme Gradient Boosting (XGBoost or XGB), random forest (RF), support vector machine (SVM), Gaussian Naïve Bayes (GNB), and logistic regression (LR). Their respective functions were created to initialize the models and their predictions at the same time, with some accepting parameters such as learning rate. All of their hyperparameters are not tuned and hence only the default values are used.

E. Evaluation

The more common evaluation metrics used are Area under the ROC Curve (AUC), F1-score (Macro average), precision, and recall. A niche metric is also adopted called Hamming loss. According to the scikit-learn documentation, the Hamming loss is the fraction of labels that are incorrectly predicted. Hence, the results of the Hamming loss are within the range of 0 to 1, and the lower the Hamming loss, the better the performance of a model. Figure 6 below shows an example of how the Hamming loss works against accuracy.

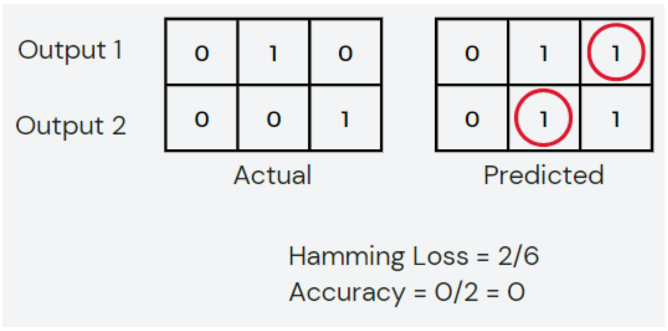


Figure 6: Hamming loss versus Accuracy

In Figure 6, we can see that the results for Hamming loss is 2/6 as there are 2 out of 6 labels which are incorrect. On the other hand, accuracy greatly punishes such inaccuracy, labelling the entire string of outputs to be incorrect even if only one of the labels are incorrect. This is an issue for this study as the outputs are as long as 11 targets, which may result in poor scores for the accuracy metric.

All the mentioned metrics are available in the scikit-learn library.

F. Deployment

The gradient boosting algorithm is found to be the best performing model out of the six, with random forest in close second. The gradient boosting model is then exported in a pickle file into a Streamlit web application and to be deployed as the data product.

Streamlit, an open-source Python web application framework for data applications is used due to its low-code solution that requires no prior experience in front-end and back-end.

V. Results

A. Model Performance

The evaluation of the models using Hamming loss, ROC AUC, Macro F1-score, precision, recall, and accuracy are shown in the tables below.

Algorithm	Hamming loss	ROC AUC	Macro F1-Score
GB	0.228	0.693	0.83
XGB	0.258	0.655	0.81
RF	0.227	0.687	0.83
SVM	0.242	0.636	0.82
GNB	0.553	0.576	0.34
LR	0.276	0.669	0.79

Table 1: Model Evaluation using Hamming loss, ROC AUC, and Macro F1-Score

Algorithm	Precision	Recall	Accuracy
GB	0.82	0.84	0.238
XGB	0.79	0.83	0.254
RF	0.81	0.85	0.238
SVM	0.79	0.86	0.206
GNB	0.81	0.30	0.048
LR	0.80	0.78	0.159

Table 2: Model Evaluation using Precision, Recall, and Accuracy

Five-fold cross validation is done for all metrics to ensure that the results of the model are generalizable and valid. Below are some figures which display the results of gradient boosting from cross validation (note: Hamming loss is not an option in the ‘scoring’ parameter of the ‘cross_validate’ method).

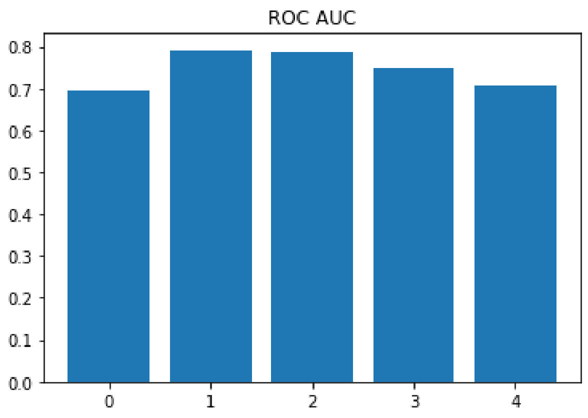


Figure 7: Plot of Five-fold Cross Validation on Gradient Boosting’s ROC AUC

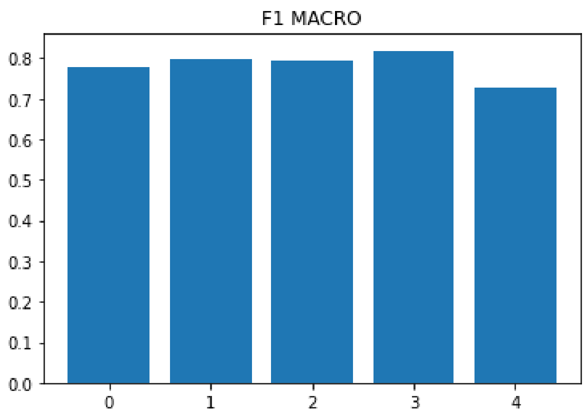


Figure 8: Plot of Five-fold Cross Validation on Gradient Boosting’s Macro F1-Score

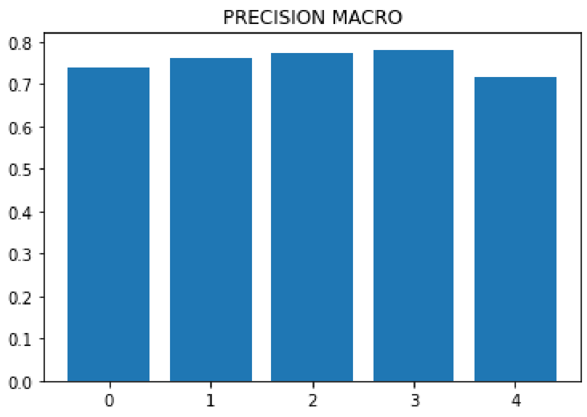


Figure 9: Plot of Five-fold Cross Validation on Gradient Boosting’s Macro Precision

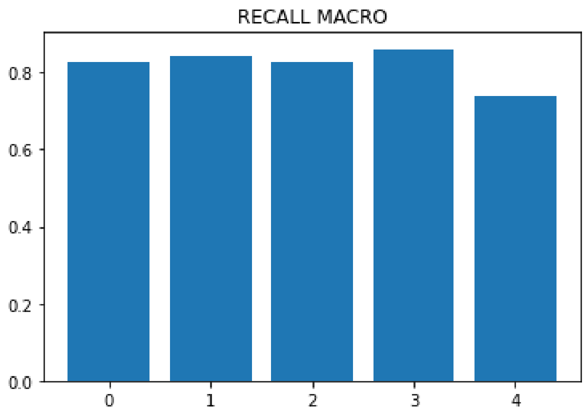


Figure 10: Plot of Five-fold Cross Validation on Gradient Boosting’s Macro Recall

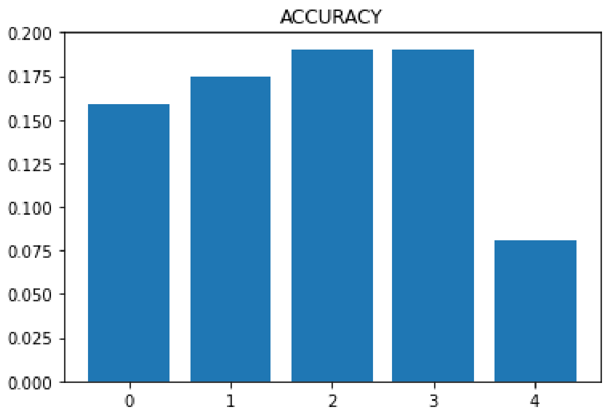
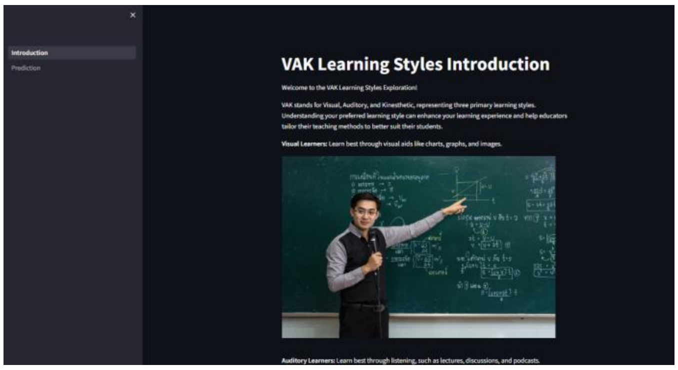


Figure 11: Plot of Five-fold Cross Validation on Gradient Boosting’s Accuracy

B. Web Application

This simple yet functional data product consists of two web pages.

1) Introduction Page: This page provides a brief description of the VAK Learning Styles. It explains the characteristics of each type of learner.



VAK Learning Styles Introduction

Welcome to the VAK Learning Styles Exploration!

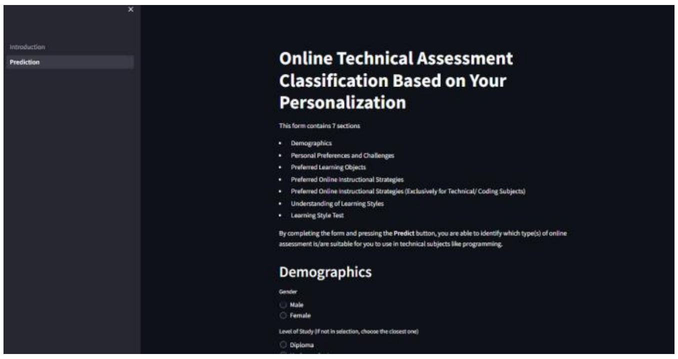
VAK stands for Visual, Auditory, and Kinesthetic, representing three primary learning styles. Understanding your preferred learning style can enhance your learning experience and help educators tailor their teaching methods to better suit their students.

Visual Learners: Learn best through visual aids like charts, graphs, and images.

Auditory Learners: Learn best through listening, such as lectures, discussions, and podcasts.

Figure 12: Introduction Page

2) Prediction Page: The prediction page is essentially an input form, with the top section describing the form's structure. The following sections are parts of the form, which enables users to input their details.



Online Technical Assessment Classification Based on Your Personalization

This form contains 7 sections

- Demographics
- Personal Preferences and Challenges
- Preferred Learning Objects
- Preferred Online Instructional Strategies
- Preferred Online Instructional Strategies (Exclusively for Technical/ Coding Subjects)
- Understanding of Learning Styles
- Learning Style Test

By completing the form and pressing the **Predict** button, you are able to identify which type(s) of online assessment is/are suitable for you to use in technical subjects like programming.

Demographics

Gender

Male

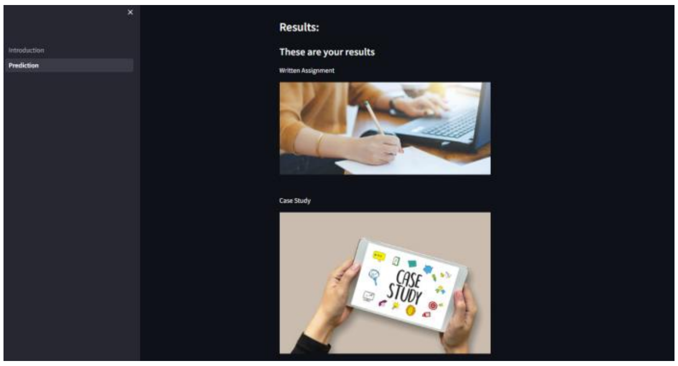
Female

Level of Study (If not in selection, choose the closest one)

Diploma

Figure 13: Prediction Page

In the same page, users can see the output by clicking the "Predict" button after completing the form. The output will suggest the suitable types of online technical assessment for the user based on their input information.



Results:

These are your results

Written Assignment

Case Study

Figure 14: Output from Prediction Page

VI. Discussion

The gradient boosting and the random forest models are very similar in performance, and it is decided that GB is better with the ROC AUC having a more significant margin. The performance are then ranked as follows: XGB, SVM, LR, and the worst performing by a wide margin GNB.

For this study, the metrics Hamming loss, ROC AUC, and F1-Score are given more importance. Hamming loss is emphasized based on the explanation in section IV. Methodology subsection E. Evaluation whereby accuracy may be unsuitable in assessing the performance of the models. ROC AUC and F1-Score are used as they are more robust evaluation metrics than precision and recall.

The plots of the five-fold cross validation in Figures 7 to 11 show that GB performs well on unseen data for all metrics except accuracy. This goes to show that GB is a robust model, and its performance scores are reliable. It also further concretes the stance that accuracy is should not be used in this study.

It should be mentioned that there are flaws in this study. The Google Forms survey was distributed mainly around Universiti Malaya, and some other public universities in Malaysia. Hence, the generalizability of the results is not certain.

VII. Conclusion

A. Outcome

In conclusion, gradient boosting and random forest algorithms do well on this dataset to classify suitable online technical assessments for students. Their results rounded up to 2 decimal places are Hamming loss of 0.23, ROC AUC of 0.69, and F1-Score of 0.83.

B. Limitations and Improvements

This study although revised many times, is not perfect as there are still other optimization works that can be done. The models did not have their hyperparameters tuned. It is suggested to perform GridSearchCV or genetic algorithms to search for the optimal hyperparameters.

There was an attempt to resolve the imbalanced data issue, yet it is still insufficient. Some technique suggestions are resampling techniques through oversampling and data sampling, and data augmentation. Dimensionality reduction and feature selection should also be explored due to the large number of variables that were at disposal. Finally, other algorithms such as ensemble methods can also be explored to improve the results of this study.

C. Future Work

Future researchers on the topic may delve into how other learning styles models will affect the classifications of the online technical assessments. They may also propose other relevant target variables based on the trend of their time. Hopefully in the near future, we may be able to see such recommender systems being used in universities to help improve the learning experience of university students.

Acknowledgment

I would like to express my utmost gratitude to my project supervisor, Associate Prof. Dr. Nor Liyana bt Mohd Shuib for her dedicated and invaluable guidance throughout the development of this data science project. Her expertise and encouragement have been instrumental in enhancing the quality of this work.

I would also like to extend my many thanks to my coursemates Janice Chong See Wai and Jasmine Chong See Yan, whom I have worked with under the same lecturer, for their unconditional help from time to time.

References

Abduh, M. Y. M. (2021). Full-time online assessment during COVID-19 lockdown: EFL teachers' perceptions. *Asian EFL Journal*, *28*(1.1), 26-46.

Bailey, K. M. (1997). *Learning About Language Assessment: Dilemmas, Decisions, and Directions*. Heinle & Heinle.

Barbe, W. B., Swassing, R. H., & Milone, M. N. (1979). Teaching Through Modality Strengths: Concepts and Practices.

Çetinkaya, A., Baykan, Ö. K., & Kırgız, H. (2023). Analysis of Machine Learning Classification Approaches for Predicting Students&rsquo; Programming Aptitude. *Sustainability*, *15*(17), 12917. https://www.mdpi.com/2071-1050/15/17/12917

Dema, K. (2021). Understanding Students' C Programming Language Learning Styles: A Case Study in College of Science and Technology. *Journal of Information Engineering and Applications*, *11*(1), 7-14. https://doi.org/https://doi.org/10.7176/JIEA/11-1-02

Dikli, S. (2003). Assessment at a distance: Traditional vs. alternative assessments. *Turkish Online Journal of Educational Technology-TOJET*, *2*(3), 13-19.

García, P., Amandi, A., Schiaffino, S., & Campo, M. (2005). Using Bayesian networks to detect students' learning styles in a web-based education system. *Proc of ASAI, Rosario*, *115*, 126.

Ocepek, U., Bosnić, Z., Šerbec, I. N., & Rugelj, J. (2013). Exploring the relation between learning style models and preferred multimedia types. *Computers & Education*, *69*, 343-355.

Papadakis, S. (2023). MOOCs 2012-2022: An overview. *Advances in Mobile Learning Educational Research*, *3*, 682-693. https://doi.org/10.25082/AMLER.2023.01.017

Roberts, G. H., & Verbyla, J. L. (2003). An online programming assessment tool. Proceedings of the fifth Australasian conference on Computing education,

Seyal, A. H., Mey, Y. S., Matusin, M. H., Siau, N. H., & Rahman, A. A. (2015). Understanding students learning style and their performance in computer programming course: Evidence from Bruneian technical institution of higher learning. *International Journal of Computer Theory and Engineering*, *7*(3), 241.

Ulloa-Cazarez, R. L., López-Martín, C., Abran, A., & Yáñez-Márquez, C. (2018). Prediction of online students performance by means of genetic programming. *Applied Artificial Intelligence*, *32*(9-10), 858-881.