E-ISSN : 2988-585X (Online)

Journal Elektronik Sistem Informasi (JESII) UNIVERSITAS KEBANGSAAN REPUBLIK INDONESIA (UKRI)

Volume 1 No 2 December 2023

DOI : 10.31848/jesii.xxxx.xxxx

Essay Scoring for Essay Evaluation Using Support Vector Machine in predicting Youth Break the Boundaries Scholarship Applicants

Qoriah Indah Susilowati[1], Züleyha Yılmaz Acar[2]

[1,2]Department of Computer Engınerrıng, Selcuk University

Article Info

Article history:

Received Oct 13, 23
Revised Oct 30, 23
Accepted Dec 12, 23

Keywords:

SVM
Essay Scoring
YBB
Scholarship

ABSTRACT

This study proposes an automated essay scoring system utilizing Support Vector Machine (SVM) to predict successful candidates for the Youth Break the Boundaries Scholarship. The system aims to address the time-consuming and subjective nature of evaluating scholarship applicants' essays. A dataset comprising essays from previous applicants and corresponding human evaluation scores is utilized to train the SVM model. The model incorporates various textual features such as essay length, grammar, vocabulary, coherence, and argument structure. Performance evaluation is conducted through cross-validation techniques and compared against human assessment scores. The results demonstrate that the SVM-based automated essay scoring system achieves a high degree of accuracy in predicting scholarship applicants who meet the evaluation criteria. By leveraging machine learning techniques, this approach reduces the burden of manual evaluation, ensures consistent and objective scoring, and accelerates the selection process. Furthermore, the study identifies the most influential features in determining essay quality and applicant suitability. These insights can assist scholarship committees in refining their evaluation criteria and improving the selection process. The proposed automated essay scoring system enhances efficiency while maintaining fairness and objectivity in the scholarship evaluation process. It allows committees to evaluate a larger pool of applicants accurately and efficiently, providing deserving candidates with a fair opportunity to receive the Youth Break the Boundaries Scholarship.

Corresponding Author :

Qoriah Indah Susilowati
Department of Computer Engınerrıng, Selcuk University

Selçuk Üniversitesi Rektörlüğü, Alaeddin Keykubat Yerleşkesi, Akademi Mah. Yeni İstanbul Cad. No:369 Posta Kodu:42130 Selçuklu-Konya

Email: qoriahindahsusilowati204@gmail.com

1. INTRODUCTION

Education plays an important role in people's lives because education can guarantee the survival of the nation and state. Higher education is the most important part of producing competitive and professional resources in their fields so they can compete globally [1]. The problem that occurs especially in Indonesia is the high cost of education which is not evenly felt by the people [2].

Several educational institutions, both private and state, especially in tertiary institutions, provide scholarships offered to students. Based on the official website of one of the universities in Indonesia (pmb.itera.ac.id) it says that the Government has provided more than 400,000 scholarship quotas [3]. Of course, this number does not cover the total number of students who will reach 9.32 million in 2022, quoted from datadindonesia.id [4]. In Indonesia, scholarships are not only given by the government, but by several agencies, foundations, and scholarships from universities themselves. One of the youth foundations in Indonesia is Youth Break the Boundaries (YBB). This foundation has been established since 2017. Since

2022 YBB has been actively contributing to providing Education funds to Indonesian Students. The YBB scholarship itself is different from other scholarships which require academic grades as the main factor. This is because the most important factor of this scholarship is that the student is active in non-academic activities which are usually described by each candidate recipient in their essay.

The selection process for YBB scholarship program has so for been carried out in a semi-online and manual way, where student register their application in YBB's official website then after registration is closed, the organizer of this scholarship will conduct the interview for selected candidate manually. With manual selection system, the selection process becomes difficult and takes a long time. The reason is in 2023 YBB scholarship applicants reached 8000 where 1500 of them completed their essay submissions. Of course, decision-making like this is very risky to be inconsistent due to an element of subjectivity [5]. By using data mining algorithms on existing data, it will be known which characteristic mapping patterns are taken so that they become easier, more consistent an

2. RESEARCH METHOD

2.1. Automated Essay Scoring System

The research methodology employed in this case is based on the effectiveness of SVM (Support Vector Machine). SVM is an effective machine learning algorithm for separating two classes or categories. In the context of AES, SVM can help distinguish between essays that meet the criteria for success and those that do not. SVM can handle complex classification problems by using kernel transformation techniques, allowing it to effectively handle nonlinear data. In AES research, SVM can handle the complexity of essay quality by employing suitable kernels to transform essay data into higher-dimensional feature spaces. Additionally, SVM incorporates the concept of "support vectors," which enables the algorithm to focus on important examples that support the division between classes. This feature helps reduce overfitting and improve model generalization. The use of SVM in AES research can be beneficial when dealing with essays that exhibit unique or uncommon characteristics. Furthermore, SVM provides better interpretability compared to some other machine learning algorithms. This means we can analyze support vectors, interpret kernel coefficients, and understand the feature contributions in classification decisions. In the context of AES, this can help understand the factors influencing essay evaluation.

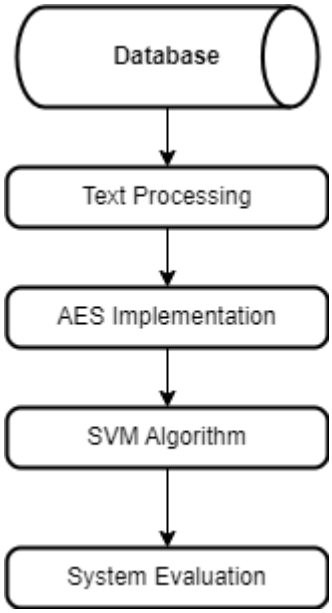

Figure 1 Research Flowchart

The research objective is to develop an AES system using the SVM algorithm to predict recipients of the Youth Break the Boundaries scholarship based on essay quality. Therefore, several steps are needed in its development, such as collecting a dataset consisting of essays from previous Youth Break the Boundaries scholarship applications. The next step is preprocessing, including cleaning the data from special characters, tokenization, removing stop words, stemming, and other necessary transformations to prepare the data for use. Next, it is necessary to identify relevant features from the essays that will be used as input in the SVM algorithm. These features may include essay length, word count, keyword density, vocabulary richness, and other features relevant to essay evaluation. The developed AES model is then evaluated using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, or other relevant metrics. Cross-validation is performed to ensure good generalization. The evaluation results are then analyzed to understand the performance of the developed AES model. The most influential factors in essay evaluation are identified, providing a deeper understanding of successful essay quality in the scholarship program. The final step of this research is the interpretation of findings, as well as the advantages and limitations of the developed AES system. Finally, conclusions are drawn regarding the potential use of this system in the Youth Break the Boundaries scholarship program. The implications of the research are recommendations for further development to enhance the quality of AES and its potential application in other scholarship programs.

3. RESULT AND DISCUSSION

Based on the results of the system design that has been carried out, a learning model is produced with a predictive assessment feature of essay answers. The main sections of the essay answers. The following details related to the findings in the study.

3.1. Data Collection

Data collection activities are carried out directly on the website of the official Youth Break the Boundaries programs. The data used comes from participants who registered with the program and then the data is exported for later analysis.

	essay_id	essay_set	essay	rater1_domain1	rater2_domain1	rater3_domain1	domain1_score
0	1	1	For better Decent Work and Economic Growth whe...	6	5	NaN	11
1	2	1	Dear Sir/Madam, i am particularly interested i...	3	2	NaN	5
2	3	1	Two years ago the world's economy was hampered...	5	6	NaN	13
3	4	1	By 2021, The development of world economics is...	5	5	NaN	10
4	5	1	work is one of the drivers of a decent life fo...	4	4	NaN	8

Figure 2 Example of a dataset

The image is a dataset that has been collected along with the scores given by the panelists from Youth Break The Boundaries. The score for the participant's answer is based on the panelist's assessment. So that the values that have been obtained can be used as a validation test at the final stage. The dataset used for modeling uses training results from the data to find the appropriate model.

3.2. Model Buidling

Making a model is a representation of making using a method that uses machine learning. Making a model makes a dataset can be used as training material to give value to the answers of students who will be tested later. The dataset will perform several processes to become a model where the dataset will be a preprocessing, implementation of Word2Vec as well as SVM and Naïve Bayes Gaussian which is done only for comparison.

3.3. Prepocessing

At this stage, the prepared dataset will carry out preprocessing as well as data cleaning, namely removing unnecessary data such as NaN and some unused columns to help solve arithmetic problems so there is no need to create a function from scratch, so the library used is described in code tester source code so that research activities use a local server and use Collaborators as tools.

```
import numpy as np
import pandas as pd
import nltk
import matplotlib.pyplot as plt
import seaborn as sns
import re
import pickle
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import LinearRegression

#import csv
from google.colab import files
uploaded=files.upload()
```

Figure 1 Function to import csv

Making a model is a representation of making using a method that uses machine learning. Making a model makes a dataset can be used as training material to give value to the answers of students who will be tested later. The dataset will perform several processes to become a model where the dataset will be a preprocessing, implementation of Word2Vec as well as SVM and Naïve Bayes Gaussian which is done only for comparison. Based on the data source code, this is the stage used to prepare a library which consists of several functions that have been used before, so that these functions can be used without having to create the same function from scratch. The function used starts from line one, namely the Numpy function. This function stands for Numerical python, which is commonly used to manage numerical data, where this system is given the pseudonym np. In the second line the library used is pandas which functions to access files containing datasets, the pandas library uses the pseudonym pd to make calling the function easier. Furthermore, the Natural Language Tool Kit is used which functions to manage the words entered from the dataset. Next is the Regular Expression (re) function or commonly called REGEX which functions to determine the text to be used. In the next row there is the pickle library which functions to serialize and serialize python objects. Serialization is the process of converting python objects into byte representations that can be stored in files or sent over a network. While deserialization is the process of returning python objects from representations that are stored or received.

The next library used in this research is sklearn, which is an algorithm used for machine learning and data mining. Sklearn itself has many functions as a machine learning algorithm, preprocessing data, evaluating models used in research. This library provides comprehensive documentation. Based on the data source code, this is the stage used to prepare a library which consists of several functions that have been used before, so that these functions can be used without having to create the same function from scratch. The function used starts from line one, namely the Numpy function. This function stands for Numerical python, which is commonly used to manage numerical data, where this system is given the pseudonym np. In the second line the library used is pandas which functions to access files containing datasets, the pandas library uses the pseudonym pd to make calling the function easier. Furthermore, the Natural Language Tool Kit is used which functions to manage the words entered from the dataset. Next is the Regular Expression (re) function or commonly called REGEX which functions to determine the text to be used. In the next row there is the pickle library which functions to serialize and serialize python objects. Serialization is the process of converting python objects into byte representations that can be stored in files or sent over a network. While deserialization is the process of returning python objects from representations that are stored or received.

The next library used in this research is sklearn, which is an algorithm used for machine learning and data mining. Sklearn itself has many functions as a machine learning algorithm, preprocessing data, evaluating models used in research. This library provides comprehensive documentation. The source code above explains how to access the dataset that will be used using collaborator tools. The first line is a special library call for Google collaborator tools users in uploading or accessing the files to be used so they are stored there. The next line is the initialization of a df variable where df stands for dataframe which describes the dataset that will be used later. The value of the df variable contains dataset file access using the pd library that calls the read_csv function. Next is the function to print the first 5 rows of the data used. Following are the results of the output.

Out[118]:

	essay_id	essay_set	essay	rater1_domain1	rater2_domain1	rater3_domain1	domain1_score	rater1_domain2	rater2_domain2	domain2_score	...	rate
0	1	1	For better Decent Work and Economic Growth whe...	6	5	NaN	11	NaN	NaN	NaN	...	
1	2	1	Dear Sir/Madam, I am particularly interested i...	3	2	NaN	5	NaN	NaN	NaN	...	
2	3	1	Two years ago the world's economy was hampered ...	5	6	NaN	13	NaN	NaN	NaN	...	
3	4	1	By 2021, The development of world economics is...	5	5	NaN	10	NaN	NaN	NaN		
4	5	1	work is one of the drivers of a decent life to...	4	4	NaN	8	NaN	NaN	NaN		

5 rows × 28 columns

Figure 4 Capture Dataset

In the picture above there are 28 data columns that have empty data or NaN, which must be removed during preprocessing. From the uploaded dataset, the columns to be used are essay_id, essay_set, essay, domain1_score. When viewed from the overall data contained in this dataset, it has a range of various values. Of course, the application of the algorithm will have an effect. Therefore, it is necessary to normalize the data. This aims to change the scale of the data so that it has a uniform or standard range.

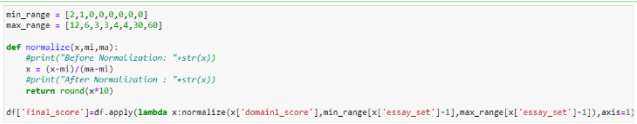

Figure 5 Normalization Function

```
def remove_puncs(essay):
    essay = re.sub("[^A-Za-z ]","",essay)
    return essay

df['clean_essay'] = df['clean_essay'].apply(lambda x:remove_puncs(x))
```

Figure 3 Remove Puncs Function

In this study, the function used is the normalize () function, where x is the value to be normalized, mi is the minimum value in the normalization range and ma is the maximum value in the normalization range. the normalize() function is applied to each row in the 'df' DataFrame by using the apply() function. The value to be normalized is the value in the 'domain1_score' column, while the normalized range used is obtained from the min_range and max_range lists based on the 'essay_set' value in each row. The normalization results are then stored in the 'final_score' column in the 'df' DataFrame. After performing the normalization function, it will proceed with the preprocessing process.

```
def clean_essay(essay):
    x=[]
    for i in essay.split():
        if i.startswith("@"):
            continue
        else:
            x.append(i)
    return ' '.join(x)

df['essay'] = df['essay'].apply(lambda x:clean_essay(x))
```

Figure 7 Clean Essay Function

Create the clean_essay function and use the function in the 'essay' column in the DataFrame 'df' where x is initialized to create an empty list that will be used to store words after the cleaning process. The essay.split function is used to iterate over each word in the essay text which will be split using spaces. Furthermore, in the 4th line is the code to check if the sentence in the essay begins with the @ symbol. This is because if the essay starts with @ it will be executed to continue to the next iteration without adding the word to the list x but because it is not part of the essay and is only the name of a user either social media or other accounts. Using the clean_essay function, the code will clean the text in the 'essay' field of words starting with the "@" symbol and return the cleaned text. The result will be saved back to column 'essay' in DataFrame 'df'.

```
stop_words = set(stopwords.words('english'))
def remove_stop_words(essay):
    word_tokens = word_tokenize(essay)
    filtered_sentence = []
    for w in word_tokens:
        if w not in stop_words:
            filtered_sentence.append(w)
    return ' '.join(filtered_sentence)
```

Figure 8 Stopword Function

This section will function to remove stop words or common words that often appear in an English essay but do not have significant data from a text. In this section, we will also solve the words of each essay with the word_tokenize() function. At the end of this code, a stop word will be executed using the romeve_stop_words function, the results of which will be stored in the clean_essay column in the new df..

Out[156]:

	essay_id	essay_set	essay	domain1_score	final_score	clean_essay
0	1	1	For better Decent Work and Economic Growth whe...	11	9	For better Decent Work Economic Growth still P...
1	2	1	Dear Sir/Madam, I am particularly interested i...	5	3	Dear SirMadam particularly interested economi...
2	3	1	Two years ago the world's economy was hampered...	13	11	Two years ago world s economy hampered outbrea...
3	4	1	By 2021, The development of world economics is...	10	8	By The development world economics starting ...
4	5	1	work is one of the drivers of a decent life fo...	8	6	work one drivers decent life person working h...

Figure 9 Clean Essay

The final stage of preprocessing in this research is removing punctuation from the essay with the remove_puncs function, punctuation will be removed from the text in the 'clean_essay' column and the results will be stored back in the 'clean_essay' column in the 'df' DataFrame. After the preprocessing function is performed, the data changes to the following.

3.4. Prepocessing

In this stage is the process of dividing the text into tokens. Tokens are a series of characters that can be separated by spaces or punctuation. In NLTK, there are 2 types of tokenizers for the word level that are most often used, namely word_tokenize and wordpunct_tokenize [13]. The difference between the two tokenizers is that word_tokenize actually uses the Treebank tokenizer. Meanwhile, wordpunct_tokenize separates tokens using regular expressions. There is a difference in the results of the tokenization of the two tokenizers. Word_tokenize will separate the standard contraction into 2 different tokens, where one of the tokens will contain quotes ('). Meanwhile wordpunct_tokenize will separate into 3 tokens, where the quotation marks (') become separate tokens. In this study, there are 8 steps related to text analysis in essays. In this stage is the process of dividing the text into tokens. Tokens are a series of characters that can be separated by spaces or punctuation. In NLTK, there are 2 types of tokenizers for the word level that are most often used, namely word_tokenize and wordpunct_tokenize [13]. The difference between the two tokenizers is that word_tokenize actually uses the Treebank tokenizer. Meanwhile, wordpunct_tokenize separates tokens using regular expressions. There is a difference in the results of the tokenization of the two tokenizers. Word_tokenize will separate the standard contraction into 2 different tokens, where one of the tokens will contain quotes ('). Meanwhile wordpunct_tokenize will separate into 3 tokens, where the quotation marks (') become separate tokens. In this study, there are 8 steps related to text analysis in essays.

1. def sent2word

```
def sent2word(x):
    x=re.sub("[^A-Za-z0-9]"," ",x)
    words=nltk.word_tokenize(x)
    return words
```

Figure 10 Sent2word Fucntion

This function is used to change sentences into words. The steps are removing characters other than letters and numbers, then using the tokenizer from the nltk library to break up the tokens into words.

2. def sent2word

```
def essay2word(essay):
    essay = essay.strip()
    tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
    raw = tokenizer.tokenize(essay)
    final_words=[]
    for i in raw:
        if(len(i)>0):
            final_words.append(sent2word(i))
    return final_words
```

Figure 11 essay2word function

This function is performed to convert essays into words by removing spaces at the beginning and end of the essay text, using the tokenizer from the nltk library to separate text into sentences and using the send2word function to convert each sentence into a list of words.

3. def sent2word

```
def noOfChar(essay):
    count=0
    for i in essay2word(essay):
        for j in i:
            count=count+len(j)
    return count
```

Figure 12 No Of Char Function

This function counts the number of characters in the essay. And the '2 word essay' function gets a list of words, then iterates over each word and counts the number of characters.

4. def avg_word_len(essay)

```
def avg_word_len(essay):
    return noOfChar(essay)/noOfWords(essay)
```

Figure 13 Avg_word Fuction

This function is used to calculate the average word length in an essay by dividing the number of words.

5. def noOfSent(essay)

```
def noOfSent(essay):
    return len(essay2word(essay))
```

Figure 14 noOfsent Function

This function counts the number of sentences in the essay by using the function 'essay2word' to get a list of words, and calculates the length of the list.

6. Def check_spell_error(essay)

```
def check_spell_error(essay):
    essay=essay.lower()
    new_essay = re.sub("[^A-Za-z0-9]"," ",essay)
    new_essay = re.sub("[0-9]","",new_essay)
    count=0
    all_words = new_essay.split()
    for i in all_words:
        if i not in words:
            count+=1
    return count
```

Figure 15 spell Error Check Fucntion

This function is used to count the number of words in the essay that are not in the word 'words' dictionary. It uses the re.sub() function to remove unnecessary characters from the essay text, and then separates the text into words. This function will count the number of words that are not in the 'words' dictionary.

3.5. Vectorizing

This function is used to count the number of words in the essay that are not in the word 'words' dictionary. It uses the re.sub() function to remove unnecessary characters from the essay text, and then separates the text into words. This function will count the number of words that are not in the 'words' dictionary. The next stage is to convert the text into a numerical representation that can be used for modeling. CountVectorizer performs feature extraction from text by counting the frequency of occurrence of words in the text.

```
vectorizer = CountVectorizer(max_features = 10000, ngram_range=(1, 3), stop_words='english')
count_vectors = vectorizer.fit_transform(df['clean_essay'])
feature_names = vectorizer.get_feature_names_out()
data = df[['essay_set','clean_essay','final_score']].copy()
X = count_vectors.toarray()
y = data['final_score'].to_numpy()
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)
```

Figure 16 Vectorizing Function

The first line is used to create an object for countVectorizer with several parameters such as max_features which functions to determine the number of features or words taken for vector formation. In this case, it's limited to the top 10,000 features according to the frequency of occurrence. In addition, ngram_range is used to determine the range 1-3 which is used to consider unigrams, bigrams and trigrams in vector formation.

The next line using the countvectorizer functions to convert the cleaned essay text (clean_essay) into a word frequency vector. fit_transform is used to learn a dictionary of words from data and effectively transform text into a vector representation. feature_names = vectorizer.get_feature_names_out() functions to retrieve a list of features or words used in vector formation from the CountVectorizer object. This will be used later to analyze the model results. data = df[['essay_set','clean_essay','final_score']].copy(): Makes a copy of the data from the essay_set, clean_essay, and final_score fields in the DataFrame df. It is used to store data that has been processed and will be used in modeling. X = count_vectors.toarray() is used to convert the word frequency vector matrix (count_vectors) into an array matrix that can be used in modeling. Each row in the matrix represents a feature vector for a single document or text. y = data['final_score'].to_numpy() is used to retrieve the final_score column from the DataFrame data and convert it to a numpy array. These will be used as targets or labels in modeling. X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3) functions to separate data into training sets (X_train, y_train) and test sets (X_test, y_test) using the train_test_split function from the scikit-learn library. In this study, we will try 3 times with a varied amount of training and testing data, namely 50:50, 70:30 and 90:10.

3.6. Vectorizing

```
#Import svm model
#from sklearn import svm
from sklearn.svm import LinearSVC

#Create a svm Classifier
#clf = svm.SVC(kernel='linear') # Linear Kernel
lsvm= LinearSVC()

#fitting
lsvm.fit(X_train, y_train)

#Train the model using the training sets
#clf.fit(X_train, Y_train)

#Predict the response for test dataset
y_preds = lsvm.predict(X_test)
```

Figure 17 SVM Algorithm Initiation

```
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics

# Model Accuracy: how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_preds))
```

```
Accuracy: 0.39275622912920627
```

Figure 4 Sample Accurary Output

Table 1 Result Table Comparation

Training	Testing	SVM Algorithm	Naïve Bayes
50%	***50%***	***39%***	***36%***
70%	30%	39%	37%
90%	10%	42%	36%

Based on the provided data, we can evaluate the performance of the SVM algorithm and Naïve Bayes classifier for the given training and testing splits. SVM Algorithm. With a 50% training and 50% testing split, the SVM algorithm achieved an accuracy of 39%. With a 70% training and 30% testing split, the SVM algorithm achieved an accuracy of 39%.
With a 90% training and 10% testing split, the SVM algorithm achieved an accuracy of 42%.

In this research naïve also with a 50% training and 50% testing split, the Naïve Bayes algorithm achieved an accuracy of 36%. This means that when trained on 50% of the data and tested on the other 50%, the Naïve Bayes classifier correctly classified 36% of the instances. With a 70% training and 30% testing split, the Naïve Bayes algorithm achieved an accuracy of 37%. In this case, it correctly classified 37% of the instances when trained on 70% of the data and tested on the remaining 30%. With a 90% training and 10% testing split, the Naïve Bayes algorithm achieved an accuracy of 36%. This indicates that when trained on 90% of the data and tested on the remaining 10%, it correctly classified 36% of the instances.

Based on the given data, the SVM algorithm performed slightly better than the Naïve Bayes classifier in terms of accuracy. It's recommended to evaluate the models using multiple metrics and consider the specific requirements and characteristics of the problem at hand when selecting the most appropriate algorithm.

4. CONCLUSION

In conclusion, this research aims to develop an AES system using the SVM algorithm to predict recipients of the Youth Break the Boundaries scholarship based on essay quality. Through the collection and preprocessing of a dataset consisting of essays from previous scholarship applications, relevant features were identified and used as input in the SVM algorithm. The developed AES model was evaluated using appropriate metrics, and cross-validation ensured good generalization. The analysis of the evaluation results provided insights into the performance of the model and identified influential factors in essay evaluation. The developed AES system showed potential in accurately predicting scholarship recipients based on essay quality. The research contributes to the field of automated essay scoring and highlights the effectiveness of the SVM algorithm in this context. The findings provide a deeper understanding of the factors that contribute to successful essay quality in the Youth Break the Boundaries scholarship program. The interpretability of the model enhances the understanding of the essay evaluation process. The implications of this research include recommendations for further development to improve the quality of the AES system and its potential application in other scholarship programs. The findings can be used to enhance the scholarship selection process, providing a more efficient and objective evaluation method. Overall, this research demonstrates the feasibility and potential of using automated essay scoring with the SVM algorithm in predicting scholarship recipients based on essay quality. It opens up avenues for further research and advancements in the field of educational assessment and scholarship selection processes.

REFERENCES

[1] K. D. ANDRIADI, E. T. W. ASIH, A. A. W. DEWI, K. NUGRAHA, and M. D. SAMADHINATA, "Efektifitas Penyelenggaraan Program Beasiswa Bidikmisi Di Universitas Pendidikan Ganesha," *J. Ilm. Akunt. dan Humanika*, vol. 8, no. 3, pp. 206–212, 2019, doi: 10.23887/jinah.v8i3.20015.

[2] A. Rusli, "Analisis Penggunaan Dana Beasiswa Berprestasi Pada Mahasiswa Pendidikan Ekonomi FKIP UNTAN," *Artik. Penelit.*, 2017.

[3] "Beasiswa – Penerimaan Mahasiswa Baru ITERA." https://pmb.itera.ac.id/biaya-pendidikan/beasiswa/ (accessed Apr. 07, 2023).

[4] M. A. Rizaty, "Jumlah Mahasiswa Indonesia Sebanyak 9,32 Juta Orang pada 2022," Feb. 15, 2023. https://dataindonesia.id/Ragam/detail/jumlah-mahasiswa-indonesia-sebanyak-932-juta-orang-pada-2022 (accessed Apr. 07, 2023).

[5] C. Anam and H. B. Santoso, "Perbandingan Kinerja Algoritma C4.5 dan Naive Bayes untuk Klasifikasi Penerima Beasiswa," *Energy - J. Ilm. Ilmu-Ilmu Tek.*, vol. 8, no. 1, pp. 13–19, 2018, [Online]. Available: https://ejournal.upm.ac.id/index.php/energy/article/view/111

[6] Modul Kuliah Data Mining, "Bab Iv Klasfikasi," *Modul Kuliah- Data Min.*, pp. 63–87, 2021.

[7] B. Lewis Sevcikova, "Human versus Automated Essay Scoring: A Critical Review," *Arab World English J.*, vol. 9, no. 2, pp. 157–174, 2018, doi: 10.24093/awej/vol9no2.11.

[8] S. Essays and S. Andr, "Sammanfattning," 2012.

[9] P. with Code, "Automated Essay Scoring | Papers With Code." https://paperswithcode.com/task/automated-essay-scoring (accessed Jun. 02, 2023).

[10] "All You Need to Know About Automated Essay Scoring Systems | AgreeYa Solutions," Apr. 30, 2021. https://agreeya.com/article/all-you-need-to-know-about-automated-essay-scoring-systems/ (accessed Apr. 13, 2023).

[11] K. Kumari and S. Yadav, "Linear regression analysis study," *J. Pract. Cardiovasc. Sci.*, vol. 4, no. 1, p. 33, 2018, doi: 10.4103/jpcs.jpcs_8_18.

[12] J. Korstanje, "Assumptions of Linear Regression | Towards Data Science." https://towardsdatascience.com/assumptions-of-linear-regression-fdb71ebeaa8b (accessed Jun. 02, 2023).

[13] B. Santosa, "No Title," in *Data Mining Teknik Pemanfaatan Data untuk Keperluan Bisnis*, Yogyakarta: Graha Ilmu, 2007.