AUTOMATED GRADING SYSTEM

A PROJECT REPORT

Submitted by

ABINA S (TKM21MCA-2001)

to

The APJ Abdul Kalam Technological University

In partial fulfillment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATION

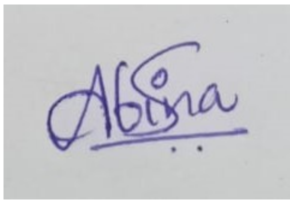
Thangal Kunju Musaliar College of Engineering
Kerala

DEPARTMENT OF COMPUTER APPLICATION

MAY 2023

DECLARATION

I undersigned hereby declare that the project report on **AUTOMATED GRADING SYSTEM USING MACHINE LEARNING** , submitted for partial fulfillment of the requirements for the award of the degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under the supervision of Prof. Jasmin M R. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma, or similar title of any other University.



Kollam

16-05-2023

ABINA S

DEPARTMENT OF COMPUTER APPLICATION

TKM COLLEGE OF ENGINEERING, KOLLAM

2021-23

CERTIFICATE

This is to certify that the report entitled **AUTOMATED GRADING SYSTEM USING MACHINE LEARNING** submitted by **ABINA S (TKM21MCA-2001)** to the APJ Abdul Kalam Technological University in partial fulfillment of the Masters degree in Computer Application is a bonafide record of the project work carried out by her under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Internal Supervisor | Head of the Department | External Examiner

ACKNOWLEDGEMENT

First and foremost I thank GOD almighty and my parents for the success of this project. I owe sincere gratitude and heart-full thanks to everyone who shared their precious time and knowledge for the successful completion of my project.

I am extremely grateful to **Dr. Fousia M Shamsudeen**, Head of the Department, Department of Computer Application, for providing me with the best facilities.

I would like to express my sincere gratitude to the project coordinator **Prof. Vaheetha Salam**, Department of Computer Application, for their valuable guidance, support, and encouragement throughout the entire project.

I would like to thank my project guide **Prof. Jasmin M R** , Department of Computer Application, motivated me throughout the project.

I would like to thank my coordinator and project guide **Prof. Natheera Beevi M** , Department of Computer Application, for the unwavering support throughout this project.

I profusely thank all other faculty members in the department and all other members of TKM College of Engineering, for their guidance and inspiration throughout my course of study.

I owe my thanks to my friends and all others who have directly or indirectly helped me in the successful completion of this project.

ABINA S

ABSTRACT

AUTOMATED GRADING SYSTEM for essays refers to the utilization of computer programs to evaluate and score essays written in response to specific prompts. It involves automating the assessment process, which offers benefits to both educators and learners by facilitating iterative improvements in students' writing skills. In traditional grading methods, evaluators need to manually read and evaluate each paper, which can be a time-consuming process, especially when dealing with a large number of papers. Automated grading systems leverage the power of machine learning algorithms to analyze essays and provide accurate grades. By implementing these systems, institutions can significantly reduce the time required for grading papers, allowing teachers to focus on other important tasks such as providing feedback to students. The proposed grading system mentioned in the project aims to use machine learning algorithms such as Linear Regression, support vector regression (SVR), and Random Forest (RF) to automate the grading process. By analyzing various features extracted from essays and incorporating natural language processing techniques, the system aims to accurately predict scores for essays in a timely and efficient manner. The effectiveness of the system is evaluated using mean squared error as a performance metric. The results demonstrate the potential of machine learning models in automating the grading process and providing reliable feedback to students.

Contents

List of Figures

List of Abbreviations

LR Linear Regression

RFR Random Forest Regressor

SVR Support Vector Regression

Chapter 1

Introduction

AUTOMATED GRADING SYSTEM is a tool for evaluating and scoring essays written in response to specific prompts. It can be defined as the process of scoring written essays using computer programs. The process of automating the assessment process could be useful for both educators and learners since it encourages iterative improvements in students' writing. The assessment of essays plays a vital role in evaluating students' writing skills and providing valuable feedback. However, manual grading of a large number of essays can be time-consuming, subjective, and prone to human errors. To address these challenges, automated grading systems have emerged as a promising solution, leveraging the power of machine learning and natural language processing.

The goal of this project is to develop an automated grading system that can accurately predict scores for essays based on various linguistic and structural features. The proposed system uses machine learning models, namely Linear Regression, Support Vector Regression (SVR), and Random Forest, to train on a dataset of essays that are already graded by human raters. These models learn the underlying patterns and relationships between the features and the corresponding scores, enabling them to make predictions on new, unseen essays.

The system employs a series of preprocessing techniques to clean and prepare the essays for analysis. This involves removing irrelevant symbols, punctuation, and stop words, as well as tokenizing the text to extract meaningful words and sentences. Also extract additional features from the essays, such as word count, character count, sentence count, average word length, and counts of different parts of speech. These features provide valuable insights into the structure, complexity, and language usage within the essays.

To represent the textual data in a machine-readable format, the CountVectorizer technique from sci-kit-learn, which converts the essays into a matrix of token counts. This representation enables the machine learning models to work with the transformed features and make predictions based on the learned patterns from the training data. Then evaluate the performance of the system using mean squared error as a measure of the prediction accuracy. By comparing the predicted scores with the actual scores assigned by human raters, we can assess the system's effectiveness in grading essays. The results demonstrate the potential of machine learning models in automating the grading process and providing reliable feedback to students promptly.

The proposed automated grading system combines the power of machine learning models, natural language processing techniques, and feature extraction to predict scores for essays. By automating the grading process, this system offers an efficient and objective approach to evaluating student essays and providing constructive feedback.

1.1 Problem Statement

The project focuses on data preprocessing and machine learning modeling. Data preprocessing involves applying NLP techniques to clean and prepare text data for further analysis. The system also extracts additional features from essays, such as character count, word count, sentence count, average word length, and part-of-speech counts. These features, along with a Bag-of-Words representation of the essays, create a comprehensive set of input features for the machine learning models.

The system employs three regression models: Support Vector Regression (SVR), Random Forest Regression, and Linear Regression. These models are trained using the extracted features and Bag-of-Words representation. The objective is to predict a target variable based on the provided data. Each regression model possesses distinct strengths and characteristics, and their integration into the system aims to enhance prediction accuracy and performance.

By utilizing NLP techniques, extracting informative features, and leveraging multiple regression models, the system strives to provide precise predictions and valuable insights for the given problem. This combination allows for a thorough analysis of the data and improves the system's understanding and predictive capabilities.

1.2 Objective

The goal is to accomplish the following:

- To reduce the workload of teachers by automating the grading process.
- To provide immediate feedback to students, allowing them to learn from their mistakes and make improvements in their writing skills.
- To provide consistent and objective evaluations without any biases.
- To improve the efficiency, consistency, and accuracy of the grading process.

Chapter 2

Literature Survey

A literature review is the comprehensive study and interpretation of literature that relates to a selected topic. When doing a literature review, research questions are defined, and then relevant literature is sought and analyzed to address these issues. By reanalyzing the study's data, it is possible to acquire fresh insights, which is an advantage of literature reviews. A literature review is both a summary and an explanation of the complete and current state of information on a topic as contained in academic books and journal articles. There are two types of literature reviews you may be required to write in college: one is written as a stand-alone assignment in a course, while the other is done as an introduction to or preparation for a longer piece of writing, typically a thesis or research report. The primary objective and perspective of the review, as well as the hypothesis or thesis argument developed, depend on the type of review are writing. It can learn the distinctions between these two types by reading published literature reviews or the introductory chapters of theses and dissertations in the subject area. Note the framework of their arguments and how they approach the issues.

2.1 Purpose of the Literature Review

1. It chooses top-notch research papers or studies that are pertinent, significant, important, and valid and summarises them into a single comprehensive report to provide readers with quick access to information on a certain issue.

2. By requiring them to describe, assess, and compare original research in this particular field, it gives researchers who are starting their research in a new area a great place to start.

3. It makes sure that researchers don't repeat already completed studies.

4. It can indicate potential directions for future research or suggest topics to concentrate on.

5. It emphasizes the important findings.

6. It points up gaps, discrepancies, and inconsistencies in the literature.

7. It offers a helpful critique of the methods and strategies used by other researchers.

2.2 Related Works

Muhammad Farrukh et. al[1] propose a novel approach that incorporates various machine learning and natural language processing techniques such as Wordnet, Word2vec, word mover's distance (WMD), cosine similarity, multinomial naive Bayes (MNB), and term frequency-inverse document frequency (TF-IDF). The approach utilizes solution statements and keywords to automatically evaluate descriptive answers and predict their grades. Experimental results show that the word mover's distance (WMD) technique outperforms cosine similarity in overall evaluation performance. With sufficient training, the machine learning model can be used independently for grading subjective answers. The proposed approach achieves accuracy without the MNB model being 88%, and the incorporation of the MNB model further reduces the error rate of 1.3%. Overall, this research contributes to the advancement of automatic subjective answer evaluation by combining machine learning and natural language processing techniques. It addresses the limitations of traditional counting-based methods and provides valuable insights for educational institutions and organizations involved in subjective answer evaluation. The findings offer potential opportunities to automate and streamline the grading process, improving efficiency and accuracy.

Masashi Okano et. al[2] propose a method to improve the training of automated essay scoring (AES) models by considering the effects of rater biases. Traditional AES models rely on handcrafted features, while more recent approaches use deep neural networks to eliminate the need for manual feature engineering. However, the assigned grades in training datasets often exhibit biases due to the characteristics of the raters. When only a few raters are assigned to grade each essay, their individual biases can impact the scores, leading to reduced performance of AES models trained on such biased data. To address this issue, the paper introduces a new approach that leverages item response theory (IRT) models. These models can estimate essay scores while accounting for the effects of rater biases. By training AES models using IRT-based scores, the proposed method aims to mitigate the negative impact of rater biases in the training data. The incorporation of IRT-based scores in the training process enhances the fairness and accuracy of AES models by considering the biases introduced by raters. This approach contributes to the advancement of AES techniques by addressing the challenges associated with biased data. It offers a promising solution to improve the reliability and effectiveness of automated essay scoring systems.

Prasad et. al[3] introduce an automated essay grading system that utilizes deep learning techniques, specifically LSTM and dense layers. The goal of the system is to reduce the effort and time required for manual evaluation of essays in educational institutions. By leveraging deep learning models, the system can grade a large number of papers within a stipulated time frame. The use of LSTM allows the system to capture long-term dependencies and analyze essays with varying lengths and complex structures. Dense layers further process the extracted features from LSTM and make predictions regarding the essay's grade. The proposed system has potential applications in smart schools and educational institutions, where it can streamline the grading process and provide more realistic scores when compared to human evaluations.

Salim et. al[4] system aims to create an automated English digital essay grader using machine learning. The objective is to develop a program that can automatically assess and grade essays without human intervention. The study utilizes techniques such as string kernel, word embedding, and reinforced learning to build the grading system. The system focuses on grading argumentative and narrative essays written by junior high school students. It analyzes various score features, which are indicators of the essay's quality and characteristics. These features encompass aspects such as coherence, grammar, structure, and overall effectiveness. To train the essay grader, the researchers employ XGBoost, a popular machine-learning algorithm known for its ability to handle structured data efficiently. By leveraging XGBoost as the classifier, the system learns patterns and relationships within the essays to make accurate grading predictions. To evaluate the system's performance, the researchers employ the 5-fold cross-validation method, which ensures that the model's accuracy is robust and applicable to new data. The average accuracy achieved by the automated essay grader is reported to be 66.87. Overall, this research demonstrates the potential of using machine learning techniques to automate the grading process for English digital essays. By reducing the need for manual grading, the system offers a more efficient and objective evaluation method, benefiting both teachers and students.

Singh et. al[5] develop an automated paper evaluation system for subjective handwritten answers. The objective is to create a mobile application that can read handwritten answers and grade them automatically. To achieve this, the researchers utilize the RAKE (Rapid Automatic Keyword Extraction) algorithm to extract key phrases from the answers. These key phrases are then embedded using the Word2vec model, which captures the semantic meaning of words. The comparison of key phrases is performed using Word Mover's Distance, a technique that measures the similarity between two sets of words based on their distributional representations. The created model demonstrates high accuracy in evaluating the handwritten answers, approaching the level of human evaluation. This indicates the effectiveness of the automated system in accurately assessing subjective responses. In addition to grading functionality, the mobile application also incorporates accessibility features, aiming to provide a comprehensive solution for the automatic evaluation of handwritten subjective answers. These features enhance the usability and inclusivity of the application, making it accessible to a wider range of users. Overall, this project presents an innovative approach to automate the evaluation of subjective handwritten answers using advanced algorithms and embedding techniques. The developed mobile application serves as a convenient and efficient tool for educators, enabling them to streamline the grading process and provide timely feedback to students.

Zhiguo Wu et .al[6] introduce a new algorithm for the automated assessment of Chinese subjective answers. It addresses the challenge of grading subjective answers in the Chinese language, which typically require a deep understanding of the context, keywords, and sentence structure. The algorithm proposed in this paper incorporates factors considered by teachers during manual grading and utilizes advanced techniques such as sentence similarity analysis based on How-Net and dependency syntax. By analyzing these factors and applying the algorithm, the system aims to provide accurate and reliable assessments of student answers. The experiment conducted in this study evaluates the performance of the algorithm by comparing its results with an existing assessment method. The goal is to assess the algorithm's effectiveness, reliability, and potential for practical application in automating the assessment of Chinese subjective answers. Overall, this research contributes to the field of automated assessment by introducing a new algorithm specifically designed for Chinese subjective answers. The findings of the experiment provide insights into the algorithm's capabilities and its potential to improve the efficiency and accuracy of grading subjective answers in the Chinese language.

Chitrita Chaudhuri et. al[7] introduce a framework for the semi-automated evaluation of subjective papers in examinations. While existing research has primarily focused on automating online examinations with multiple-choice or short descriptive answers, this framework specifically addresses the evaluation of textual papers with subjective questions. The framework proposes the use of model answer points, which serve as reference points for evaluating the answers provided by students. By incorporating these model answer points into the evaluation process, the framework aims to improve efficiency and consistency in grading subjective papers. Additionally, the framework includes reward and penalty schemes. In the reward scheme, students who provide additional valid points beyond the model answer are rewarded with bonus marks. This encourages students to provide comprehensive and insightful answers. On the other hand, the penalty scheme helps detect and penalize unfair practices among neighboring examinees. By maintaining seat plans in the form of a neighborhood graph and computing the similarity between adjoining answer scripts, the framework can identify and address instances of unfair means. The main question-bank and model answer points are managed using Case-Based Reasoning strategies, which allow for efficient retrieval and utilization of relevant cases during the evaluation process. Overall, this research contributes to the automation of examination systems by proposing a framework that enhances the evaluation of subjective papers. By incorporating model answer points, reward and penalty schemes, and Case-Based Reasoning strategies, the framework aims to expedite the evaluation process, ensure fairness, and improve the overall assessment of subjective answers in examinations.

Aboul Ella Hassanien et. al[8] introduce a novel approach to the Automated Essay Grading System (AEGS) that combines natural language processing (NLP) techniques and a neural network grading engine. The system aims to automate the evaluation of student essays and provide constructive feedback to enhance their writing skills. The proposed system consists of two main components: Writing Features Analysis tools and a neural network grading engine. The Writing Features Analysis tools leverage NLP techniques to analyze various aspects of the essays, such as grammar, sentence structure, and vocabulary. These tools extract meaningful information to assess the quality of the writing. The neural network grading engine is trained using pre-graded essays, enabling it to develop a grading model. This model evaluates student answers and assigns grades based on the knowledge acquired from the pre-graded essays.

The system aims to provide accurate and consistent grading results. To assess the system's performance, datasets of essays from computer and information sciences college students at Mansoura University were utilized. These essays were collected during mid-term exams for specific courses. The evaluation results demonstrate a substantial agreement between the system's grades and the grades given by teachers, ranging from 70 to nearly 90. This suggests that the proposed system has the potential to serve as an effective tool for automating essay assessment, leading to significant cost reductions.

ASM Latiful Hoque et. al[9] propose an automated Essay Grading (AEG) as an important research area within educational technology. One common technique used in AEG is Latent Semantic Analysis (LSA), which involves creating a word-by-document matrix and applying Singular Value Decomposition (SVD) for analysis. However, existing AEG systems based on LSA have limitations and struggle to match the grading accuracy of human evaluators. To address these limitations, this paper proposes a new approach called Generalized Latent Semantic Analysis (GLSA) for AEG. GLSA differs from LSA by creating an n-gram-by-document matrix instead of a word-by-document matrix, allowing for a more comprehensive analysis of linguistic patterns. The goal is to improve the performance and accuracy of automated essay grading. The proposed AEG system using GLSA is evaluated using detailed representations, and its performance is compared to existing systems. The experimental results demonstrate that the GLSA-based system outperforms the traditional LSA-based systems, indicating its effectiveness in automated essay scoring. This research contributes to the advancement of automated essay grading techniques, offering a promising alternative to traditional approaches. By improving the accuracy and efficiency of essay grading, the proposed system has the potential to reduce grading costs and provide valuable feedback to students, enhancing their writing skills.

Zainab Binti Abubakar et. al[10] research is to automatically generate a domain-specific concept ontology for essays using OntoGen, a tool specifically designed for ontology creation. By building an ontology that captures the key concepts and relationships within the essay domain, the system can better understand the content and structure of the essays. To enhance the accuracy and efficiency of the automated grading process, the study utilizes natural language processing algorithms implemented through the Natural Language Tool Kit (NLTK). NLTK provides a comprehensive set of tools and libraries for text mining and linguistic analysis, allowing for the extraction of relevant information from the essays. By combining the ontology-based approach with NLTK's text-mining capabilities, the proposed system aims to improve the grading process and provide more objective evaluations of student essays. This advancement has the potential to address the issues of time consumption and subjectivity in essay grading, while also enabling prompt feedback for students. Overall, this research contributes to the field of AES by introducing a novel approach that leverages ontology and natural language processing techniques. The application of this approach has the potential to revolutionize the essay grading process, providing educators with a more efficient and objective tool for evaluating student essays.

Hoblos et. al[11] used two popular techniques, Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA), for assessing the semantic similarity between essays and answer keys. The researchers develop an essay grading system using an open-source Python software called Gensim. The system compares an essay to an answer key and assigns a grade based on the level of semantic similarity. To evaluate the system's performance, experiments are conducted using essays of different lengths. The grades assigned by the automated system are compared to those given by a human grader, typically a professor. The results indicate a strong correlation between the grades assigned by the human grader and those assigned by both LSA and LDA modeling techniques. However, the LSA-based approach shows more promising results compared to the LDA-based method. This research contributes to the advancement of automated essay grading by demonstrating the efficacy of LSA and LDA in assessing semantic similarity. The developed system offers potential benefits in terms of efficiency and consistency, allowing for more objective and reliable assessment of student essays.

Harneet Kaur et. al[12] system proposes rule-based grammar checks and surface-level coherence analysis to assess the syntactic quality of the essays. Moreover, it goes beyond traditional approaches by considering the semantic similarity between sentences using graph-based relationships derived from the essay's content. Additionally, the system takes into account the sentiment expressed in the essay by analyzing the polarity of opinion expressions. By combining these different analyses, the system aims to capture a broader range of essay characteristics and provide a more holistic evaluation. To achieve this, the algorithm utilizes 23 salient features that have demonstrated high predictive power. By focusing on fewer but more representative features, the system enhances its ability to make accurate predictions and handle noisy data effectively. To validate the system's performance, the evaluation model employs neural networks and utilizes data from the ASAP competition hosted by Kaggle. The agreement between the scores predicted by the system and those assigned by human graders is assessed using Quadratic Weighted Kappa (QWK). The system achieves a QWK score of 0.793, indicating a strong correlation with human graders' evaluations. This research advances the field of automated essay evaluation by emphasizing the joint effect of syntactic, semantic, and sentiment analysis. By considering these aspects together, the proposed system provides a more comprehensive and reliable evaluation of essays. Its potential to reduce the time and effort required for manual grading makes it a valuable tool in educational settings.

Kavinder Singh et. al[13] propose an architecture that combines pre-trained language models with adapter modules, which employ a bottleneck architecture to reduce the number of trainable parameters. This parameter-efficient design ensures excellent performance while optimizing computational efficiency. Additionally, the paper introduces a novel model that repurposes the bidirectional attention flow model to identify adversarial essays, enhancing the system's robustness. Experimental results demonstrate that the proposed model achieves state-of-the-art performance on various essay prompts, as evaluated using the Automated Student Assessment Prize dataset. The paper also provides a comprehensive overview of previous approaches attempted in this domain and demonstrates how the proposed model outperforms them. By incorporating advanced language models and innovative techniques, this research offers a significant advancement in the field of automated essay scoring. The model's ability to effectively capture long-term dependencies and detect adversarial essays contributes to improved performance and highlights its superiority over existing methods.

Chapter 3

Methodology

AUTOMATED GRADING SYSTEM are software systems developed to evaluate and score essays automatically, without the need for manual grading by human instructors. The model is trained with three machine-learning regression algorithms using an AES dataset.

3.1 Algorithm

The algorithm includes:-

- Step 1:Data Preprocessing.
- Step 2:Feature Extraction.
- Step 3:Model Training and Evaluation.
- Step 4:Model Comparison and Selection.
- Step 5:Model Deployment.

3.2 System Architecture

An automated grading system using machine learning involves several steps, including data cleaning using Natural Language Processing (NLP) techniques and feature extraction using BOW (Bag-Of-Words). The proposed system uses Support Vector Regression (SVR), Linear Regression, and Random Forest Regressor to train the model. The model's performance is evaluated using metrics like accuracy or mean squared error. The trained model is then used to make predictions on test essays, and feedback is provided to students.

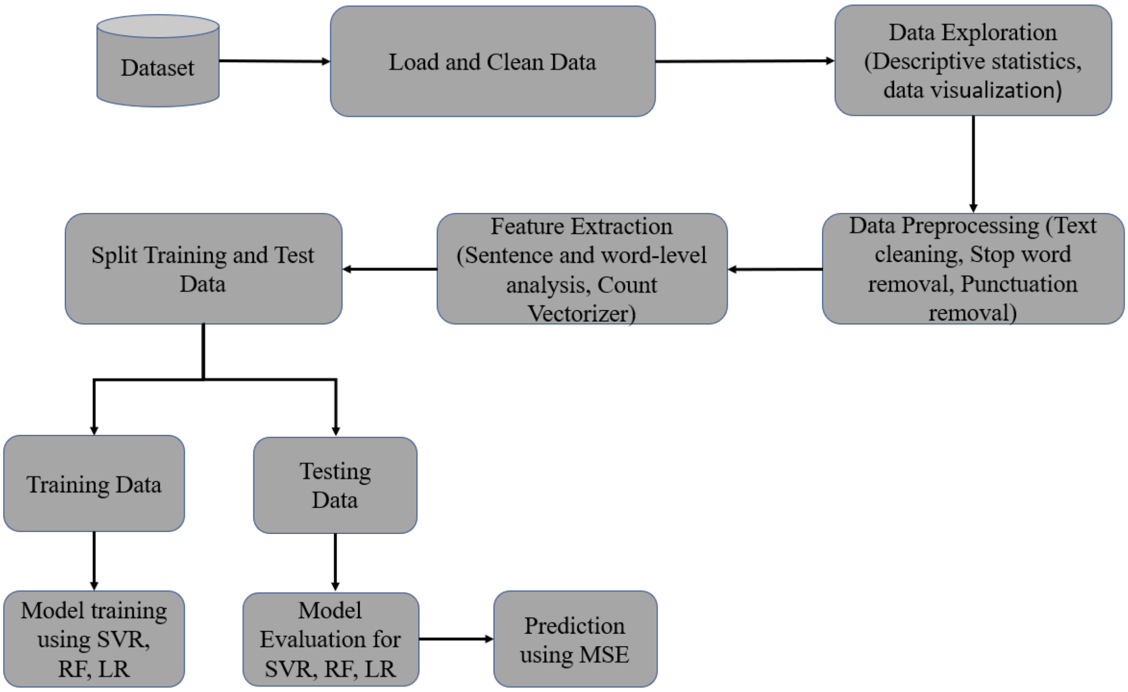


Figure 3.1: System Architecture

3.2.1 Dataset

The dataset used for the project is ”Hewlett Foundation: Automated Essay Scoring” which is a collection of essays written by students in response to specific prompts, along with corresponding scores given by human raters. The dataset includes over 12,000 essays written by students in grades 7-10 from 8 different states in the United States.

essay_id	essay_set	essay	rater1	rater2	rater3	domain1	rater1_	rater2_	domain2	rater1_tra	rater1_tra	rater1_tra	rater1_tra	rater:
19237	7	One time I was going to see my frie	11	12		23				3	3	2	3	
19238	7	One @CAPS1, I was very patient wh	9	11		20				2	2	2	3	
19239	7	Being patient! Patience is a good le	8	11		19				2	2	2	2	
19240	7	Being patient is a very hand thing fo	9	7		16				2	2	2	3	
19241	7	At one @CAPS2 in left you would h	8	8		16				2	2	2	2	
19242	7	I remember one time I was impatie	9	8		17				2	3	2	2	
19243	7	Patience in @CAPS1: Understandin	12	12		24				3	3	3	3	
19244	7	It was my first game. I was pumped	10	9		19				2	3	2	3	
19245	7	Hunting when you are turkey hunti	8	8		16				2	2	2	2	
19246	7	One day, two years ago I was going	7	8		15				2	2	2	1	
19247	7	In my opinion being patient is whe	5	8		13				0	1	2	2	
19248	7	Have you ever had to be patient be	9	12		21				2	3	2	2	
19249	7	You will say that you are board very	6	9		15				0	2	2	2	
19250	7	One time I was patient was when I	11	12		23				3	3	2	3	
19251	7	one time I was in patient. We was a	7	4		11				2	2	1	2	
19253	7	One time I had to use patience whe	8	8		16				2	2	2	2	
19254	7	Being patient is hard. I hate waiting	4	6		10				0	1	1	2	
19255	7	When I was @NUM1 yrs old my mo	9	6		15				2	2	3	2	
19256	7	One time when I was patient was w	8	8		16				2	2	2	2	
19257	7	When I was patient I was sitting at a	8	8		16				2	2	2	2	
19258	7	We were going to @CAPS1. Me and	9	12		21				2	2	2	3	
19259	7	Patience is a hard thing to have, bu	7	8		15				2	2	2	1	

Figure 3.2: Dataset

3.2.2 Data Pre-processing

Data pre-processing for text involves preparing text data for analysis or modeling by applying various techniques to clean, transform, and extract features from the text. Here, the first step is Cleaning the Essays. It involves removing unnecessary symbols, stop words (commonly used words like ”the,” ”and,” and ”is”), and punctuation from the essays. The cleaning process helps in eliminating noise and reducing the complexity of the data. To enhance the accuracy further, additional features such as the number of sentences, number of words, number of characters, and average word length are incorporated. These features provide insights into the structure and length of the essays, which can be valuable for subsequent analysis.

Techniques such as parts of speech tagging were applied to determine the counts of different parts of speech (nouns, verbs, adjectives, adverbs) in the essays. This analysis provides a deeper understanding of the language used in the essays. The essays were compared against a reference corpus to identify misspelled words. By checking the presence of each word in the corpus, the total number of misspelled words in an essay was determined.

3.2.3 Feature Extraction

Feature extraction is an important step because it transforms the raw text data into a numerical representation that can be understood by machine learning algorithms. In the context of text data, feature extraction is crucial because most machine learning algorithms operate on numerical data rather than raw text.

The main feature extraction technique used is the Bag-of-Words (BoW) model. It is implemented using the CountVectorizer class from the sci-kit-learn library. The CountVectorizer converts a collection of text documents into a matrix of token counts. It represents each essay as a vector where each element represents the count of a specific word or token in the essay. This approach allows the machine learning models to work with numerical feature vectors rather than raw text data. The CountVectorizer in the code creates a feature matrix by extracting features from the cleaned essays. It tokenizes the text, builds a vocabulary of unique words, and assigns a numerical count to each word in each essay. The resulting feature matrix represents the occurrence of words in the essays and serves as input to the machine learning models.

3.2.4 Building and training the model

Training an automated grading system is possible with the use of NLP techniques and machine learning algorithms. Machine learning regressors such as Support Vector Regression (SVR), Linear Regression, and Random Forest Regressors are used to train the model. Using the 'train test split' the dataset into two separate sets- a training set and a testing set. The system focuses on building and training a machine-learning model to predict scores for essays.

The support Vector Regression (SVR) algorithm is used to create a regression model that can predict continuous numerical values. In the given code, the SVR model is built using the sci-kit-learn library. The code first prepares the data by preprocessing and splitting it into training and testing sets. Then, an instance of the SVR class is created, allowing for the customization of hyperparameters such as the kernel type and regularization parameter. The SVR model is trained using the fit method, which adjusts the internal parameters of the model to find the optimal hyperplane that fits the training data. Once trained, the model can be used to make predictions on new data using the prediction method.

The second model is Random Forest Regressor, the RandomForestRegressor class is created from sci-kit-learn and is imported. The RandomForestRegressor class allows for the creation of a Random Forest model specifically designed for regression tasks. The Random Forest Regressor is instantiated with desired parameters. In the code, the n estimator's parameter is set to 1000, which determines the number of decision trees in the random forest. Other parameters, such as max depth, min samples split, and min samples leaf, can be specified to control the tree growth and prevent overfitting. Once the Random Forest Regressor is instantiated, it is trained on the training data using the fit() function. The fit() function takes the input features (X train) and the corresponding target values (y train) as arguments. During training, the Random Forest Regressor builds a collection of decision trees, each trained on a random subset of the training data. After training, the Random Forest Regressor can be used to make predictions on new data using the predict() function. The predict() function takes the input features (X test) and returns the predicted target values (y pred).

The third model is Linear Regression is a statistical algorithm used for predicting numerical continuous values based on a linear relationship between input features and the target variable. The Linear Regression model is built using the sci-kit-learn library. To train the Linear Regression model, the fit() function is used, which takes the input features (X train) and the corresponding target values (y train) as arguments. During training, the model estimates the coefficients (weights) for each feature and the intercept term. Once the model is trained, it can be used to make predictions on new data using the predict() function. The Linear Regression model calculates the predicted values (y pred) based on the input features (X test).

The model with the lowest MSE(Mean Squared Error) rate is selected and saved to use later to test and deploy the system. The pickle module from the Joblib library is used to serialize the models and save them as binary files

SVR(Support Vector Regression)

A Support Vector Machine (SVM) is a powerful machine learning model used for classification and regression tasks. It belongs to the family of supervised learning algorithms and is widely used in various domains, including image recognition, text categorization, and bioinformatics. At its core, an SVM aims to find an optimal hyperplane that separates data points of different classes in the feature space. The hyperplane is chosen in such a way that it maximizes the margin, which is the distance between the hyperplane and the nearest data points of each class. This concept is called ”maximum margin” and forms the foundation of SVMs. In a binary classification scenario, where there are two classes, the SVM algorithm finds the hyperplane that best separates the data points. However, in cases where a linear separation is not possible, SVMs can employ a technique called the kernel trick. The kernel trick allows SVMs to map the input data into a higher-dimensional feature space, where a linear separation is possible. Commonly used kernels include linear, polynomial, radial basis function (RBF), and sigmoid. The SVM training process involves solving an optimization problem to find the optimal hyperplane. This problem is formulated as a quadratic programming task, and various optimization algorithms, such as Sequential Minimal Optimization (SMO) and the interior point method, can be used to solve it efficiently. Once the SVM model is trained, it can classify new, unseen data points by determining which side of the hyperplane they fall on. Data points on one side are classified as belonging to one class, while those on the other side are classified as belonging to the other class. The decision boundary created by the SVM is defined by a subset of the training data points called support vectors, which are the data points closest to the hyperplane.

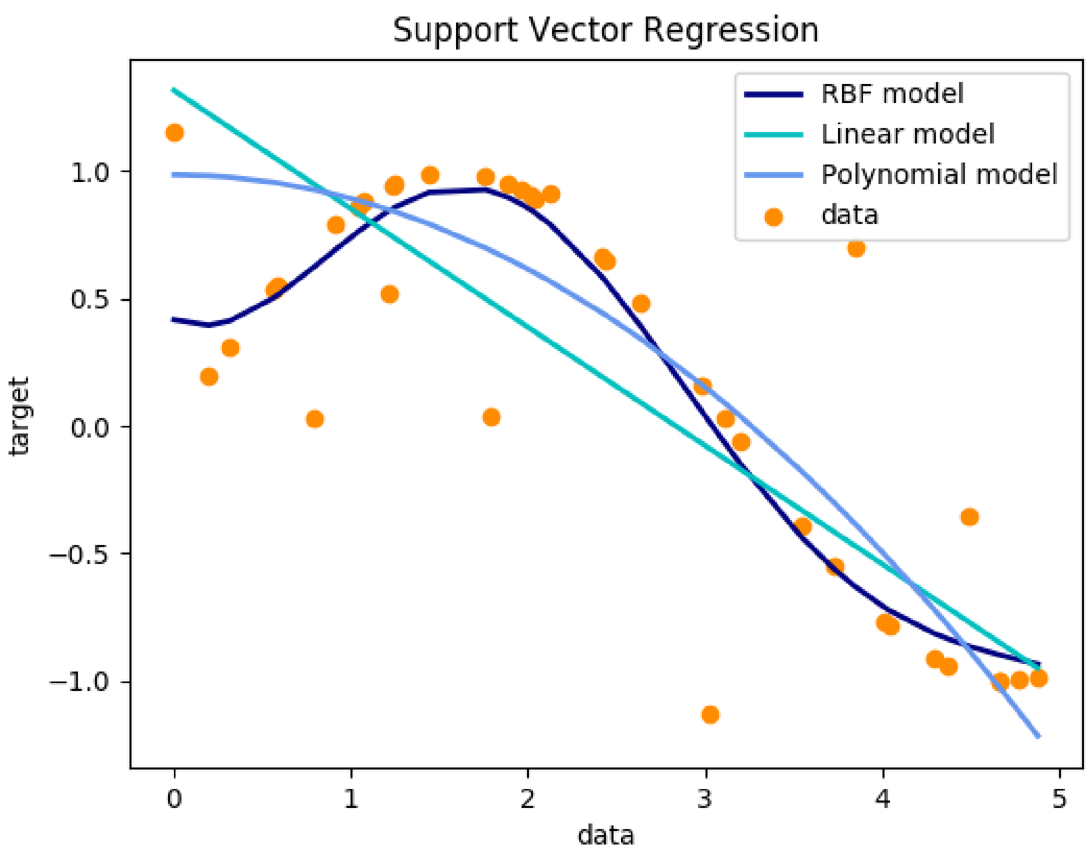


Figure 3.3: SVR

Linear Regression

Linear regression is a statistical technique used to analyze and predict the relationship between a dependent variable and one or more independent variables. It assumes a linear relationship, represented by a straight line, between the variables. In simple terms, linear regression aims to find the best-fitting line that represents the relationship between the dependent variable (often denoted as ”Y”) and the independent variables (often denoted as ”X”). The line is determined by estimating the coefficients associated with each independent variable. These coefficients indicate how the dependent variable changes when the corresponding independent variable changes, assuming other variables remain constant. The goal of linear regression is to minimize the difference between the observed values of the dependent variable and the predicted values based on the regression line. This is typically done using a method called ordinary least squares (OLS), which calculates the sum of squared differences between the observed and predicted values.

The accuracy of a linear regression model is often assessed using statistical measures such as the coefficient of determination (R-squared). R-squared quantifies the proportion of the variance in the dependent variable that can be explained by the independent variables. Higher R-squared values indicate a better fit of the model.

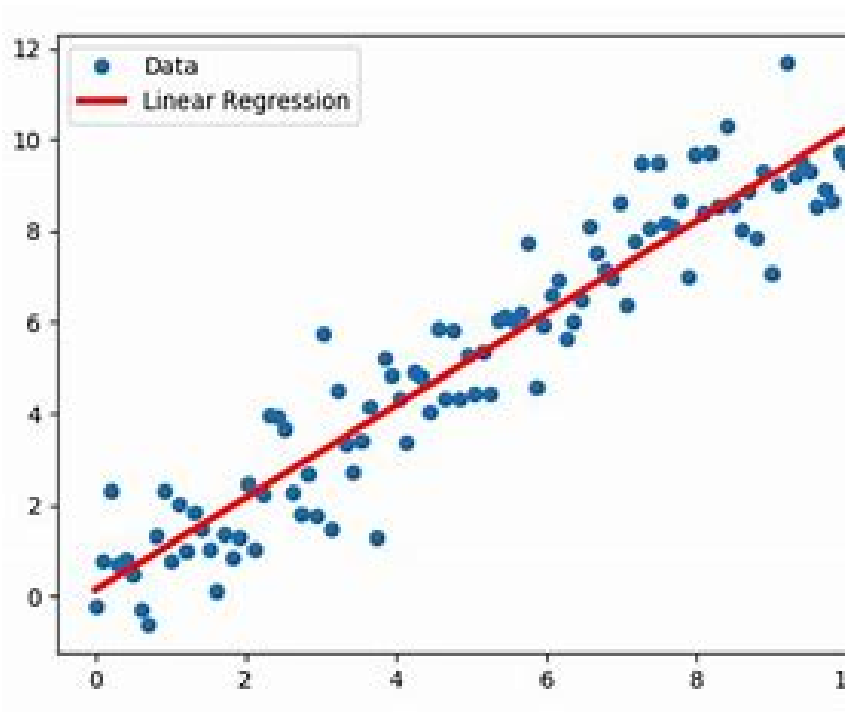


Figure 3.4: Linear Regression

Random Forest

Random Forest Regressor is a powerful machine learning algorithm used for regression tasks. It is particularly popular due to its ability to handle complex datasets and provide accurate predictions. The algorithm is based on an ensemble learning approach, which means it combines the predictions of multiple decision trees to make more robust and accurate predictions. Each decision tree is trained on a random subset of the training data, known as a bootstrap sample, and a random subset of features is considered at each node for splitting.

The training process of a Random Forest Regressor involves several steps. Firstly, random subsets of the training data are selected through a process called bagging. Each subset is used to train a separate decision tree. This introduces randomness and diversity into the model, which helps to reduce overfitting and improve generalization. During the construction of each decision tree, the algorithm randomly selects a subset of features to consider at each split. The decision trees are built by recursively partitioning the data based on the selected features. The splitting continues until a stopping criterion is met, such as reaching a maximum depth or minimum number of samples at a leaf node. This process creates a tree structure that can represent complex relationships between the features and the target variable. Once all decision trees are built, the Random Forest Regressor combines their predictions to make the final prediction. In the case of regression, this is typically done by averaging the predictions of individual trees. The ensemble nature of the algorithm helps to reduce the impact of outliers and noisy data, resulting in more robust predictions.

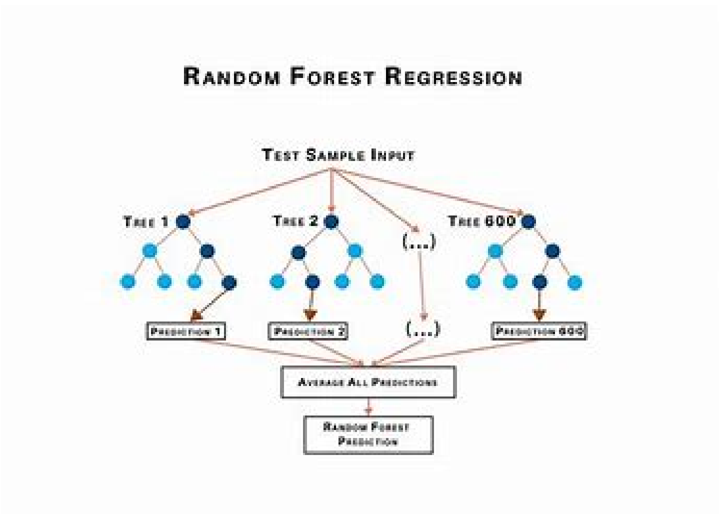


Figure 3.5: Random Forest

3.2.5 Testing and Deployment

1. **Input**:
   In the first step, the user provides the essay text through the web interface. This can be done by typing the essay directly into a text field or by uploading a file containing the essay. The input essay serves as the basis for the subsequent analysis and scoring. The web interface should provide a user-friendly way to enter the essay and ensure that the text is correctly captured.

2. **Preprocessing**:
   Once the essay text is received, it undergoes preprocessing to clean and prepare it for analysis. This step involves various operations to remove unwanted elements and standardize the text. Common preprocessing techniques include converting the text to lowercase to ensure consistency, removing punctuation marks, and handling any special characters. Additionally, stopwords, which are common words that do not contribute much to the meaning of the essay, can be eliminated to focus on more relevant content. Preprocessing is crucial to ensure that the subsequent analysis is accurate and meaningful.

3. **Feature Extraction**:
   After preprocessing, relevant features are extracted from the essay. Features capture different aspects of the essay's content and structure, providing valuable insights for scoring. Feature extraction can be based on various techniques and considerations. For example, basic features such as word count (total number of words in the essay) and sentence count (total number of sentences) give an overview of the essay's length and complexity. Other features, like average word length or the presence of specific keywords or phrases, can provide more nuanced information. Advanced feature extraction techniques, such as sentiment analysis or topic modeling, can also be applied to capture the emotional tone or thematic composition of the essay.

4. **Scoring Model**:
   Once the features are extracted, a scoring model is applied to evaluate the essay. The scoring model can be based on different approaches, depending on the available data and requirements. One common approach is to use a machine learning model trained on a dataset of essays and their corresponding scores. The model learns patterns and

relationships between the extracted features and the essay scores, allowing it to make predictions on new essays. Another approach is to employ a predefined rule-based system that applies specific criteria to assess the essay's quality. In this case, the extracted features serve as input to the rule-based system, which applies predetermined rules or scoring rubrics to generate the final score. The choice of the scoring model depends on factors such as the availability of training data, the desired accuracy, and the complexity of the evaluation criteria.

5. **Presentation**:

   The resulting score is presented to the user through the web interface. The score can be displayed as a numerical value, a grade or rating, or any other format that effectively conveys the evaluation. The web interface should provide clear and understandable feedback to the user, indicating the quality or performance of the essay. In addition to the score, the interface can offer further insights and suggestions for improvement. For example, it can highlight specific areas where the essay excelled or provide recommendations for enhancing specific aspects, such as grammar, organization, or coherence. This feedback can help the user understand the strengths and weaknesses of their essay and guide them in developing their writing skills

6. **Deploying the model**:

   The model is deployed using a flask. A web interface is valuable in automated essay grading systems as it allows for easy and convenient interaction between users (such as students, teachers, or administrators) and the grading system.

3.3 System Specifications

The application development architecture recognized for this project is specified in this section based on requirements.

3.3.1 Software Specification

The software used for the project includes:

1. Python

2. Google Colaboratory

3. Flask

3.3.2 Software Description

1. Python

   Python is a high-level, interpreted programming language known for its simple syntax and readability. It was first released in 1991 by Guido van Rossum and has since become one of the most popular programming languages in the world. Python has a wide range of applications, including web development, scientific computing, data analysis, artificial intelligence, machine learning, and automation. It is also known for its large standard library, which includes modules for a variety of tasks such as string processing, regular expressions, and file I/O. When a Python script is run, the interpreter evaluates the code and converts it into machine-readable bytecode. Python is an object-oriented programming language, meaning it allows users to handle and manipulate objects or data structures to create and run programs. Python is a reliable and popular language, constantly evolving to meet the needs of developers.

   Features:

   - Interpreted: Python is an interpreted language, meaning it does not require a compilation step before execution.

   - Dynamically-typed: Python is dynamically-typed, which means that variable types are inferred at runtime, rather than being specified in the code.

- Object-oriented: Python is an object-oriented language, meaning it supports encapsulation, inheritance, and polymorphism.
- Easy-to-learn: Python has a simple and intuitive syntax, making it easy to learn for beginners.

2. Google Colaboratory

   Google Colaboratory, also known as ”Colab,” is a cloud-based platform developed by Google that allows users to write, run, and share Python code via a Jupyter notebook interface. It is designed to provide a free and easy-to-use environment for data analysis, machine learning, and artificial intelligence tasks, with access to powerful hardware resources such as GPUs and TPUs.Colab provides a range of pre-installed libraries and frameworks commonly used in data science and machine learning, such as NumPy, Pandas, TensorFlow, and PyTorch, and also supports the use of external libraries and packages. Users can store and access their code and data files on Google Drive, and collaborate with others by sharing notebooks and commenting on code. Additionally, Colab allows users to run code on remote servers, making it useful for tasks that require large amounts of computing power or memory.

   Features:

   - Free access to a virtual machine with a pre-installed environment for data analysis and machine learning.
   - Ability to write and execute Python code in a Jupyter Notebook-style environment, which allows for interactive coding, documentation, and visualization.
   - Access to Google’s high-performance hardware, such as GPUs and TPUs, which can significantly speed up machine learning tasks and reduce training time.
   - Integration with Google Drive, which allows for easy sharing and collaboration on projects with others.

3. Flask

   Flask is a micro web framework written in Python. It is designed to be lightweight and modular, providing developers with the flexibility to customize and extend it to meet their specific needs. Flask includes built-in support for creating web applications, including handling HTTP requests and responses, routing, and templating. Its simplicity and ease of use make it a popular choice for building small to medium-sized web applications, APIs, and prototypes. Flask can also be easily integrated with other Python libraries and frameworks, making it a versatile tool for web development.

Features:

- Easy to get started: Flask is designed to be easy to learn and use, making it a great choice for beginners who want to build web applications quickly.
- Lightweight and flexible: Flask is a micro-framework, which means it is lightweight and flexible, allowing developers to add only the features they need for their applications.
- Built-in development server: Flask comes with a built-in development server that makes it easy to test and debug your application.
- Modular design: Flask is designed to be modular, with a core framework that provides basic functionality, and extensions that can be used to add additional features and functionality.

3.4 Hardware and experimental environment

The hardware used for the experiments includes Windows 11 Home Single Language,64-bit operating system, x64-based processor,11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz,8 GB RAM. The experimental environment was prepared by using Python 3.10 programming language. The Web framework is used Flask

Chapter 4

RESULT AND DISCUSSION

An automated grading system is a tool for evaluating and scoring essays written in response to specific prompts. It can be defined as the process of scoring written essays using computer programs. The process of automating the assessment process could be useful for both educators and learners since it encourages iterative improvements of students' writing. The project aims to develop an automated essay assessment system by use of machine learning techniques by classifying a corpus of textual entities into a small number of discrete categories, corresponding to possible grades.

The dataset is preprocessed and cleaned by removing unnecessary characters, punctuation, and stopwords. Various features are extracted from the essays, such as character count, word count, sentence count, average word length, spelling error count, and part-of-speech tag counts.

Three machine learning algorithms are employed: Support Vector Regression (SVR), Random Forest, and Linear Regression. For SVR, the dataset is split into training and testing sets. The CountVectorizer is used to convert the essays into numerical features. The SVR model is then trained using the training data and tested on the testing data. The mean squared error is calculated as an evaluation metric. Similarly, for Random Forest, the dataset is preprocessed, and features are extracted. The Random Forest model is trained using the training data and evaluated using the mean squared error. For Linear Regression, the CountVectorizer is used to transform the essays into numerical features. The Linear Regression model is trained on the training data and evaluated using the mean squared error. The trained models can be saved for future use, and the saved models can be loaded to make predictions on new data.

4.1 Training and Validation Results

The training and validation results of machine learning algorithms refer to the performance of the models during the training phase and their ability to generalize to unseen data during the validation phase. The project involves the training and validation of three machine learning models: Support Vector Regression (SVR), Random Forest, and Linear Regression.

For SVR, the model is trained using the training set, and the trained model is then used to predict the scores for the validation set. The mean squared error (MSE) is calculated as the evaluation metric. The MSE value for SVR is 2.83. For Random Forest, the model is trained using the training set, and the trained model is used to predict the scores for the validation set. The MSE value for Random Forest is 0.88. For Linear Regression, the model is trained using the training set, and the trained model is used to predict the scores for the validation set. The MSE value for Linear Regression is 8.63, which is significantly higher than the other two models.

These MSE values indicate the performance of each model in terms of prediction accuracy. Lower values of MSE indicate better performance, as it represents the average squared difference between the predicted scores and the actual scores. Therefore, the Random Forest model performs the best among the three models in this project, followed by SVR. Linear Regression has the highest MSE, indicating relatively poor performance in predicting the scores. The following table shows the MSE of different machine learning models used in the respective study.

SL.NO	Model	MSE
1	Linear Regression	8.36
2	Random Forest Regressor	0.88
3	SVR	2.83

Figure 4.1: Model Comparison

4.1.1 Training and Validation Graphs

Bar plots can be used to visually represent training and testing MSE. The training MSE graph shows the performance of the model on the training data, while the testing MSE graph indicates the model's performance on a separate set of testing data.

The following plots provide a concise representation of the performance of different algorithms used.

Figure 4.2: Training and Testing MSE

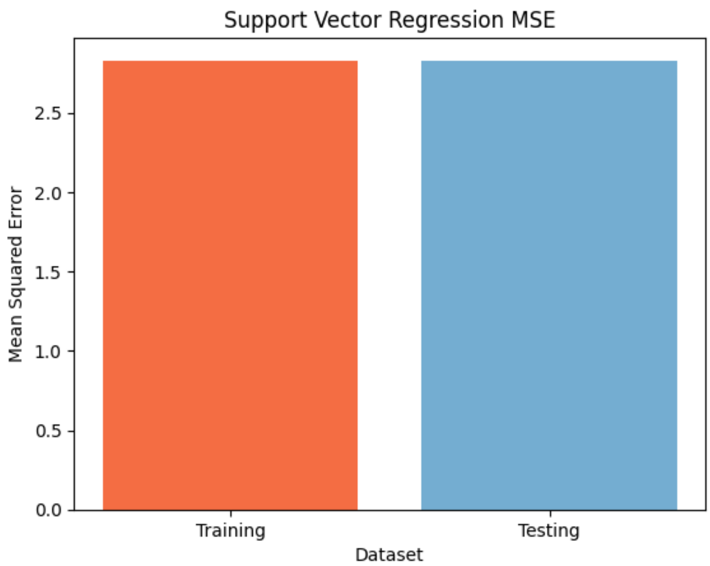


Figure 4.3: Training and Testing MSE

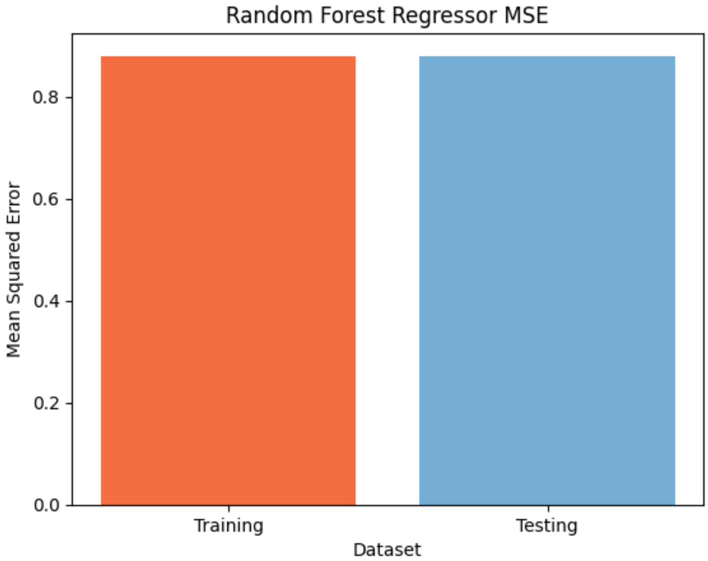


Figure 4.4: Training and Testing MSE

4.1.2 Graph of Comparison of the models used.

1. Bar Plot

   A bar plot, also known as a bar chart or bar graph, is a visual representation of data using rectangular bars of varying lengths. It is commonly used to display and compare categorical or discrete data. The length of each bar corresponds to the magnitude or frequency of the data it represents. In a bar plot, the x-axis represents the categories or groups being compared, while the y-axis represents the measurement or value associated with each category. Each category is assigned a separate bar, and the height or length of the bar represents the value of the corresponding data.

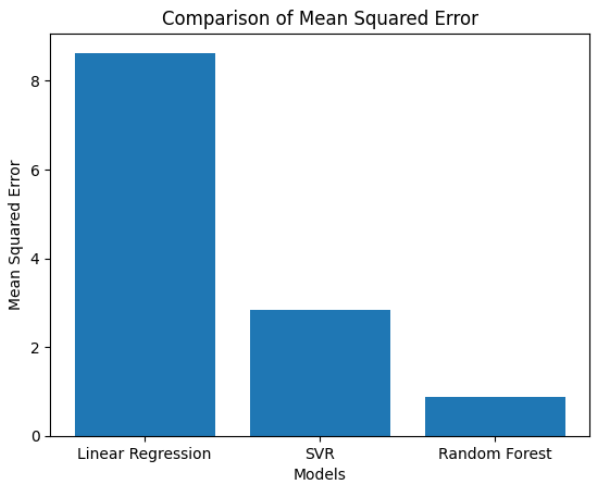


Figure 4.5: Comparison of models

2. Line Plot

   A line plot, also known as a line chart or line graph, is a graphical representation of data where data points are connected by straight lines. It is commonly used to display trends or patterns over time or to show the relationship between two continuous variables. In a line plot, the x-axis represents the independent variable (such as time or a range of values) and the y-axis represents the dependent variable (the variable being measured or observed). Each data point is plotted as a dot on the graph, and a line is drawn to connect these points in the order they occur or based on the underlying order of the data. Line plots are particularly useful for visualizing and understanding trends, changes, or patterns in data over continuous intervals. They can reveal the overall direction of the data, identify fluctuations or seasonal variations, and provide insights into the relationship between variables.

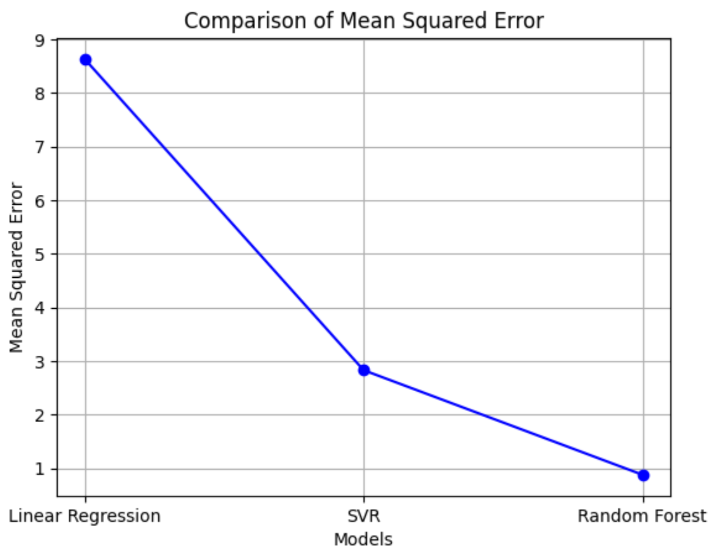


Figure 4.6: Comparison of models

3. Heatmap

   A heatmap is a graphical representation of data where values in a matrix are displayed as colors to visually represent patterns, relationships, or distributions. It is often used to analyze and present complex datasets, especially when dealing with large amounts of numerical data. In a heatmap, the data matrix is represented by a grid of cells, with each cell typically filled with a color corresponding to the value it represents. The color scale can be chosen to reflect different gradients, such as a spectrum of colors from low to high values or a diverging color scheme to highlight positive and negative values.

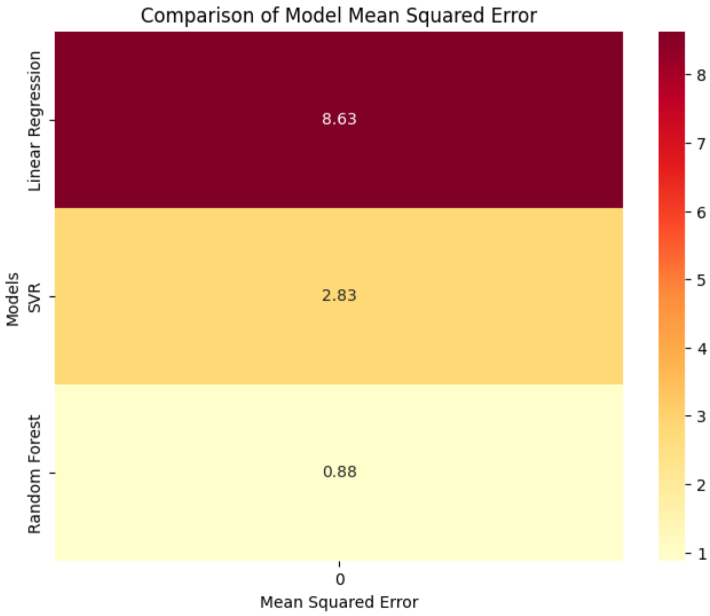


Figure 4.7: Comparison of models

Chapter 5

CONCLUSION

The project aims to develop an automated grading system using machine learning algorithms. The dataset consists of essays along with their scores provided by multiple raters. The main objective is to train a model that can accurately predict the scores of essays based on their textual features.

The initial step of the project involves data preprocessing. The dataset is explored by checking its shape, missing values, and basic statistical measures. Skewness and kurtosis of the target variable (domain1 score) are calculated to understand the distribution of scores. A histogram is plotted to visualize the distribution. Data cleaning and feature engineering are performed to prepare the dataset for modeling. Columns that are not required for the grading task, such as 'rater1 domain1' and 'rater2 domain1', are dropped. The 'domain1 score' column is normalized to a scale of 0-10 based on the score ranges for each essay set. Textual preprocessing techniques are applied to the 'essay' column, including removing special characters, stop words, and punctuation. Additional features are derived from the essays, such as word count, character count, sentence count, average word length, and part-of-speech counts.

Next, machine learning algorithms are applied to train models for score prediction. The dataset is split into training and testing sets using the train test split function. Three algorithms are used in the project: Support Vector Regression (SVR), Random Forest Regression, and Linear Regression. For SVR, a trained model is saved or loaded from a file. The model is used to predict scores on the testing set, and the mean squared error (MSE) is calculated as an evaluation metric. Similarly, for Random Forest Regression, a trained model is saved or loaded. The model is then used to predict scores on the testing set, and the MSE is calculated. For Linear Regression, a model is trained using the linear regression class. The trained model is saved or loaded, and predictions are made on the testing set. The MSE is calculated as an evaluation metric.

The project concludes by summarizing the MSE values obtained from the three models. These metrics provide an indication of the performance of each algorithm in predicting the scores of the essays. Based on the MSE values, the Linear Regression model seems to perform the worst, with a high MSE value. On the other hand, the SVR model shows better performance, with a lower MSE value. The Random Forest Regressor model achieves the lowest MSE value, indicating the best performance among the three models.

5.1 Future Enhancement

Two future enhancements in the user interface of automated grading systems are the development of an interactive dashboard and the incorporation of gamification elements.

The interactive dashboard provides users, such as educators or administrators, with a comprehensive overview of grading statistics, trends, and performance metrics. It allows users to access and analyze data in a visual and intuitive manner, using graphs, charts, and customizable displays. The dashboard empowers users to monitor and track grading progress, identify patterns and trends, and make data-driven decisions. It enhances the user experience by providing a centralized platform for accessing and managing grading-related information efficiently.

On the other hand, incorporating gamification elements in the user interface introduces game-like elements to the grading process, making it more engaging and motivating for users. By introducing features like badges, leaderboards, or rewards, the system encourages users to actively participate and excel in their grading tasks. Users can earn badges or achievements for achieving specific milestones, such as completing a certain number of grading tasks or providing high-quality feedback. Leaderboards display rankings based on grading performance, fostering healthy competition and a sense of achievement. These gamification elements add an element of fun and motivation to the grading process, increasing user engagement and satisfaction.

REFERENCE

[1] Bashir, Muhammad Farrukh, et al. *"Subjective answers evaluation using machine learning and natural language processing." IEEE Access 9 (2021): 158972-158983.*

[2] Uto, Masaki, and Masashi Okano. *"Learning automated essay scoring models using item-response-theory-based scores to decrease effects of rater biases." IEEE Transactions on Learning Technologies 14.6 (2021): 763-776.*

[3] Prasad, PV Hari, et al. *"Automated essay grading system using deep learning." International Research Journal of Engineering and Technology 7.03 (2020).*

[4] Salim, Yafet, et al. *"Automated English digital essay grader using machine learning." 2019 IEEE International Conference on Engineering, Technology, and Education (TALE). IEEE, 2019.*

[5] Singh, Sarika, et al. *"Automated Paper Evaluation System for Subjective Handwritten Answers." 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT). IEEE, 2021.*

[6] Li, Runhua, Yonghua Zhu, and Zhiguo Wu. *"A new algorithm to the automated assessment of the Chinese subjective answer." 2013 International Conference on Information Technology and Applications. IEEE, 2013.*

[7] Roy, Chhanda, and Chitrita Chaudhuri. *"Case-based modeling of answer points to expedite the semi-automated evaluation of subjective papers." 2018 IEEE 8th International advance computing Conference (IACC). IEEE, 2018.*

[8] Shehab, Abdulaziz, Mohamed Elhoseny, and Aboul Ella Hassanien. *"A hybrid scheme for automated essay grading based on LVQ and NLP techniques." 2016 12th International Computer Engineering Conference (ICENCO). IEEE, 2016.*

[9] Islam, Md Monjurul, and ASM Latiful Hoque. *"Automated essay scoring using generalized latent semantic analysis." 2010 13th International Conference on Computer and Information Technology (ICCIT). IEEE, 2010.*

[10] Contreras, Jennifer O., Shadi Hilles, and Zainab Binti Abubakar. *"Automated essay scoring with ontology based on text mining and nltk tools." 2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE). IEEE, 2018.*

[11] Hoblos, Jalaa. *"Experimenting with latent semantic analysis and latent Dirichlet allocation on automated essay grading." 2020 Seventh International Conference on Social Networks Analysis, Management and Security (SNAMS). IEEE, 2020.*

[12] Janda, Harneet Kaur, et al. *"Syntactic, semantic and sentiment analysis: The joint effect on automated essay evaluation." IEEE Access 7 (2019): 108486-108503.*

[13] Sethi, Angad, and Kavinder Singh. *"Natural Language Processing based Automated Essay Scoring with Parameter-Efficient Transformer Approach." 2022 6th International Conference on Computing Methodologies and Communication (ICCMC). IEEE, 2022.*

APPENDIX

Screenshots

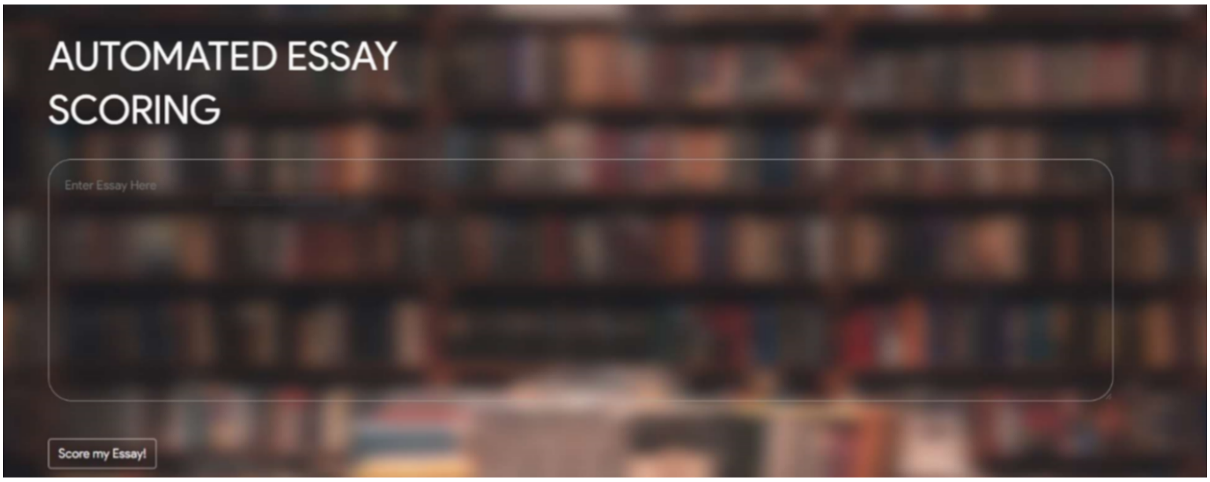


Figure A.1: Step 1

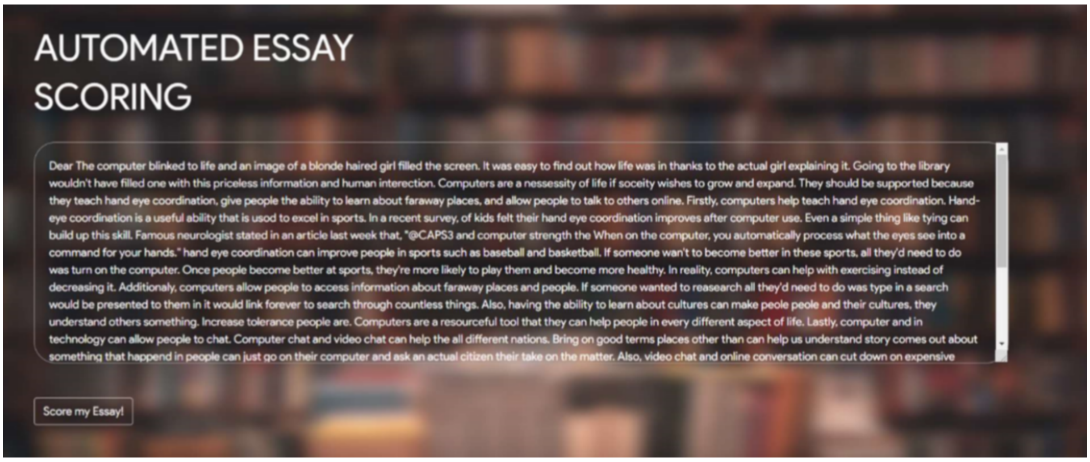


Figure A.2: Step 2

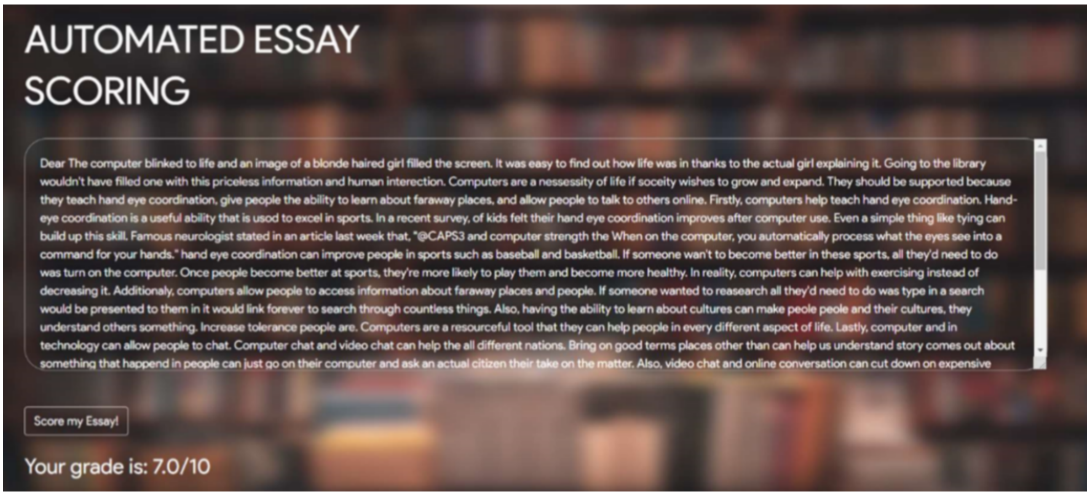


Figure A.3: Step 3