**Increment 1**

**Automated Essay Grading Using Natural Language Processing**

**Team Members: 1. Sai Phani Teja Chilukuri**

**2. Sai Yashwanth Reddy Gujjula**

**3. Naveen Bolla**

**4. Sumanth Dasari**

**Increment 1 Guidelines**

The project's objectives are as follows:

1. Analyze the provided student performance data.
2. Identify the factors that significantly affect student performance.
3. Build a predictive model that can forecast student performance based on the identified factors.
4. Evaluate the model's accuracy and identify areas of improvement.

For this increment, we have developed the following features:

* Data preprocessing: In this feature, we cleaned the provided student performance data and transformed it into a format suitable for analysis. We also handled any missing or invalid data.
* Feature selection: We used the TfidfVectorizer from the scikit-learn library to extract relevant features from the cleaned data.
* Model building: We built several machine learning models to predict student grades, including SVM, Random Forest, Naive Bayes, Decision Trees, and K-Nearest Neighbors.
* Model evaluation: We evaluated the models' accuracy using various performance metrics, including accuracy, precision, and recall.

**Related Work**

There are many studies in which they focused about student’s performance. From all these studies we have found out some key facts that play important role in acknowledging student’s merits/success such as teacher quality, school ambience, Environment, Parent Guidance.

Data Analytics has been the powerful idea for analyzing the performance of student data in our project. With the help of machine learning algorithms we can find out the most of the important factors that connect with student grades and develop models which helps to build the resource to find the performance accurately.

**Dataset**

The provided dataset contains the following fields:

student\_id

assignment1\_grade

assignment1\_submission

assignment2\_grade

assignment2\_submission

assignment3\_grade

assignment3\_submission

assignment4\_grade

assignment4\_submission

assignment5\_grade

assignment5\_submission

assignment6\_grade

assignment6\_submission

The dataset contains information on six assignments and their corresponding grades and submission dates for each student. There are no missing or invalid values in the provided dataset.

**Detail design of Features**

Data preprocessing:

1. Remove any duplicates from the dataset.
2. Drop any rows with missing or invalid values.
3. Transform the submission date fields into a datetime format.
4. Create a new feature, 'time\_taken,' that calculates the time taken to submit each assignment.

**Feature selection:**

We have Used TfidfVectorizer from the scikit-learn library for extracting relevant features from the cleaned data.

Set the maximum number of features to 500.

**Analysis**

Here, we have implemented machine learning models to find out student performances based on their assignment grades. In the dataset we used there are information which contains assignment grades of students and their submission dates with this information we created new features like time difference between assignment submission and final date of submission with this information we can guess the student’s performance and how efficient they are.

Our aim was to create a model that can perfectly guess the student’s performance depending upon their assignment grade’s. In here we tested various machine learning techniques like SVM, Random Forest, Naïve Bayes and Decision Tree and our goal is to find the best method among them we evaluate them using models like accuracy, precision and recall from our method we can see that Decision tree is the best among all with accuracies of Grade F as 0.52 precision, 0.47 as recall, 0.49 as f1-score and 234 as support.

Finally, we can say that it is possible to guess student performance depending on their assignment grades. The Random Forest model which we developed shown very good results and the further processes like optimization of feature engineering and machine learning models which can lead to an even more better performance.

**Implementation**

We used many machine learning and python techniques/ libraries to implement our machine learning feature and we have divided code into some other features which are given below.

Data loading and preprocessing: We loaded the student grade dataset into a pandas data frame as pd and the dataset of students gradings has been names as grades.csv and performed data preprocessing methods like deleting missing values and changing the submission date into numerical format.

Feature engineering: We created several new features such as the time difference between each assignment submission and the final assignment submission date. Average Grade is what we implemented here to calculate from all assignments and we also breaked them down into three parts F, D and C for finding grade category and by using TF\*IDF vector for submission dates of assignments.

Model training and evaluation: We divided data into training and testing sets and trained many machine learning language models like SVM, Naïve bayes, Random Forest and Decision tree with all these models we found out the accuracy, precision and recall functions to know the idea of which is performing good out of all those.

Model deployment: We found out that Decision Tree model as our best model out of all these and used it for future preference for projecting our student’s performance.

**Preliminary Results**

**SVM classification report:**

precision recall f1-score support

C 0.00 0.00 0.00 104

D 0.00 0.00 0.00 125

F 0.50 1.00 0.67 234

accuracy 0.50 463

macro avg 0.17 0.33 0.22 463

weighted avg 0.25 0.50 0.34 463

SVM confusion matrix:

[[ 0 0 104]

[ 0 0 125]

[ 0 1 233]]

The SVM classification report where we got values of precision, recall, f1-score and support where C, D and F grade categories and average grade is performed here where we found accuracy and averages of matrix and also a confusion matrix.

**Random Forest classification report:**

precision recall f1-score support

C 0.00 0.00 0.00 104

D 0.27 0.02 0.04 125

F 0.50 0.97 0.66 234

accuracy 0.50 463

macro avg 0.26 0.33 0.24 463

weighted avg 0.33 0.50 0.35 463

Random Forest confusion matrix:

[[ 0 2 102]

[ 0 3 122]

[ 0 6 228]]

Random Forest classifier is a most commonly used machine learning technique and the reports it generated are very much connective to the general valuations. Here, these values are accurate and precise and we also found confusion matrix.

**Naive Bayes classification report:**

precision recall f1-score support

C 0.00 0.00 0.00 104

D 0.00 0.00 0.00 125

F 0.51 1.00 0.67 234

accuracy 0.51 463

macro avg 0.17 0.33 0.22 463

weighted avg 0.26 0.51 0.34 463

Naive Bayes confusion matrix:

[[ 0 0 104]

[ 0 0 125]

[ 0 0 234]]

Naïve Bayes Classification report we found all the techniques as above and the results are liberate according to other features above and we found accuracy and confusion matrix.

**Decision Tree classification report:**

precision recall f1-score support

C 0.24 0.30 0.27 104

D 0.25 0.26 0.25 125

F 0.52 0.47 0.49 234

accuracy 0.37 463

macro avg 0.34 0.34 0.34 463

weighted avg 0.39 0.37 0.38 463

Decision Tree confusion matrix:

[[ 31 28 45]

[ 37 32 56]

[ 59 66 109]]

Decision Tree classifier is the best technique we used among all the above we used here C, D and F means highest grading of students where F is highest C is lowest. Then we found accuracy and confusion matrix. In this technique we got the best accuracy.

**Project Managemen**t

Implementation status report

**Work completed**:

**Responsibilities**

* Data loading and preprocessing: Sai Phani Teja Chilukuri
* Feature engineering: Sai Yashwanth Reddy Gujjula
* Model training and evaluation: Naveen Bolla
* Model deployment: Sumanth Dasari

**Work to be completed:**

* Refine feature engineering: Sai Phani Teja Chilukuri & Sai Yashwanth Reddy Gujjula
* Optimize machine learning models: Naveen Bolla
* Improve model deployment process: Sumanth Dasari

**Concerns:**

We encountered some issues with missing values in the dataset, but we were able to resolve them through data preprocessing.

We also encountered some issues with overfitting when training the machine learning models, but we were able to address them through hyperparameter tuning.

**Bibliography**

1. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825-2830.
2. Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
3. Breiman, L. (2001). Random forests. Machine learning