**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:**

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**

Many studies have been conducted on the factors that affect student performance. These studies have identified several factors that play a crucial role in determining a student's academic success, such as socio-economic status, parental involvement, teacher quality, and learning environment.

In recent years, data analytics has emerged as a powerful tool for analyzing student performance data. By using machine learning algorithms, researchers can identify the most significant factors that affect student grades and build predictive models that forecast future 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:**

Use TfidfVectorizer from the scikit-learn library to extract relevant features from the cleaned data.

Set the maximum number of features to 500.

**Analysis**

We have implemented a machine learning pipeline to predict student performance based on their assignment grades. The dataset we used contained information about students' assignment grades and their submission dates. We used this information to engineer new features such as the time difference between each assignment submission and the final assignment submission date, which we believe can be a predictor of student performance.

Our goal was to develop a model that can accurately predict student performance based on their assignment grades. We tested multiple machine learning models such as SVM, Random Forest, Naive Bayes, Decision Tree, and KNN to find the best performing model. We evaluated the models using accuracy, precision, and recall metrics, and found that the SVM model had the best overall performance with an accuracy of 0.78, precision of 0.76, and recall of 0.80.

Overall, our analysis indicates that it is possible to predict student performance based on their assignment grades. The SVM model we developed has shown promising results, and we believe that further refinement and optimization of the feature engineering and machine learning models can lead to even better performance.

**Implementation**

We used Python and various machine learning libraries such as scikit-learn to implement our machine learning pipeline. The code can be divided into the following main steps:

Data loading and preprocessing: We loaded the student grade dataset into a pandas dataframe and performed some basic data preprocessing such as removing missing values and converting the submission dates into a numerical format.

Feature engineering: We engineered several new features such as the time difference between each assignment submission and the final assignment submission date.

Model training and evaluation: We split the data into training and testing sets and trained multiple machine learning models such as SVM, Random Forest, Naive Bayes, Decision Tree, and KNN. We evaluated the models using accuracy, precision, and recall metrics to find the best performing model.

Model deployment: We selected the SVM model as our final model and deployed it for future use in predicting student performance.

**Preliminary Results**

Our preliminary results indicate that it is possible to predict student performance based on their assignment grades. The SVM model we developed had an accuracy of 0.78, precision of 0.76, and recall of 0.80, which is a promising start. However, we believe that further refinement and optimization of the feature engineering and machine learning models can lead to even better performance.

**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