Predicting Student Performance

Daniel Truong, Emmanuel Wang
CS 463/663 – Foundations of Machine Learning (Spring 2025)

May 20, 2025

Abstract

This paper addresses the question, "What factor is the most important for students' academic performance," by utilizing machine learning techniques like Regression and Classification models. From our results, we found that five features that most impact students' academic performance: number of failures the student has in the past, internet access, absences, education level of parents, and total alcohol consumption. Students, as well as academic institutions that are looking to provide assistance, can focus on improving the five areas.

Contents

1 Introduction

Speaking from our personal experience, we encountered many different advice (either from family members or from online sources) such as attending classes or sleeping well to achieve high academic grades, but by doing this study, we aim to find out which factors/features are related to students' academic performance, as well as discussing the following questions:

- Are there any potential relationships between different factors?
- How much does each feature impact students' academic performance?
- Which feature or group of features are the most impactful when predicting performance?

We'll touch on details about our research question and target audience in the following subsections, but our initial approach was to utilize Regression techniques to predict student grade and determine which features are important. However, due to the poor performance of Regression models, we decided to use Classification techniques and encountered much better performance after addressing the issue of class imbalance in our dataset. From our models, we found that RandomForest Classifier and Extreme Gradient Boosting Classifier performs the best, with each providing different insight into our research question.

1.1 Research Question(s)

Our main question is to answer what has the most impact on students' academic performance, and what should students focus on to achieve good academic results. We decided on this question since the significance behind academic performance is always relevant, whether it is in the eyes of parents or for college applications that can drastically change a student's future. Additionally, with lot of advice out there for students, it is hard to tell which ones are sound and effective so by conducting this study, we aim to better understand which factors in a student's life best impact academics.

1.2 Target Audience

The main target audience for this research is students of all ages, but since our dataset entails information on secondary school students' grades, it may be more relevant to that demographic. Our findings can also be utilized by academic institutions to assist students that are struggling with their coursework and help them determine what they should be doing outside of studying to improve their academic results.

2 Related Work

Our work builds upon Using Data Mining To Predict Secondary School Student Performance by Paulo Cortez and Alice Silva Cortez and Silva [2008] since we share the same dataset. Cortez et al.'s paper predicted student performance using Business Intelligence and Data Mining techniques, specifically Data Mining models such as Decision Trees. They also used binning to classify the target G3 score as well as regression. They also used different models like Decision Trees, Random Forest, Neural Networks and Support Vector Machines to predict secondary student grades of two core classes (Mathematics and Portuguese) by using past grades from first and second periods, and demographic and social information. From their work, the conclusion is that student achievement is highly affected by previous performances, but the best predictive models also shows that other features like number of absences, parent's jobs and education, and alcohol consumptions are also somewhat relevant in the prediction.

Since we shared the same dataset as Cortez et al., we decided that it was a good idea to explore it using different techniques like regression. We'll go over the details in section 3.3 but

due to poor performance on our regression models (Linear Regression, Polynomial Regression, LASSO, and Random Forest Regressor), we shifted our focus and used similar classification models/techniques that Cortez et al. used like Decision Tree Classifier, Logistic Regression, Random Forest Classifier, Gradient Boosting, and Extreme Gradient Boosting.

3 Methods

As mentioned in previous sections, we initially aimed to utilize Regression techniques, specifically the following:

- Linear Regression
- Polynomial Regression
- LASSO (Least Absolute Shrinkage and Selection Operator) Regression
- Random Forest Regressor

However, after poor performance from the models, we changed it up and used Classification techniques like:

- Decision Tree Classifier
- Logistic Regression
- Random Forest Classifier
- Gradient Boosting
- Extreme Gradient Boosting

3.1 Dataset(s) & Preprocessing

Our main dataset was pulled from Kaggle, called Student Performance by Aman Chauhan that had data on student achievement in secondary education of two Portugese schools, that included features like student grades, demographic, social and school related details. Two datasets for different core subjects (Mathematics and Portuguese) were provided, but since there were overlapping students in both of the datasets, we decided to merge the overlapping students from both datasets and get the average final grade to get our main dataset, resulting in 382 entries of students' data. The reason for this is to look at overall student performance across multiple subjects instead of focusing on just one. The following table includes details on the initial features and what they stand for.

Feature	Details
school	student's school (binary: 'GP' - Gabriel Pereira or 'MS' - Mousinho da Silveira)
sex	student's sex (binary: 'F' - female or 'M' - male)
age	student's age (numeric: from 15 to 22)
address	student's home address type (binary: 'U' - urban or 'R' - rural)
famsize	family size (binary: 'LE3' - less or equal to 3 or 'GT3' - greater than 3)
Pstatus	parent's cohabitation status (binary: 'T' - living together or 'A' - apart)
Medu	mother's education (numeric: 0 - none, 1 - primary education (4th grade), 2 - 5th to 9th grade, 3 - secondary education or 4 - higher education)
Fedu	father's education (numeric: 0 - none, 1 - primary education (4th grade), 2 - 5th to 9th grade, 3 - secondary education or 4 - higher education)
Mjob	mother's job (nominal: 'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other')
Fjob	father's job (nominal: 'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other')
reason	reason to choose this school (nominal: close to 'home', school 'reputation', 'course' preference or 'other')
guardian	student's guardian (nominal: 'mother', 'father' or 'other')
traveltime	home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)
studytime	weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)
schoolsup	extra educational support (binary: yes or no)
famsup	family educational support (binary: yes or no)
paid	extra paid classes within the course subject (Math or Portuguese) (binary: yes or no)
activities	extra-curricular activities (binary: yes or no)
nursery	attended nursery school (binary: yes or no)
higher	wants to take higher education (binary: yes or no)
internet	Internet access at home (binary: yes or no)
romantic	with a romantic relationship (binary: yes or no)
famrel	quality of family relationships (numeric: from 1 - very bad to 5 - excellent)
freetime	free time after school (numeric: from 1 - very low to 5 - very high)
goout	going out with friends (numeric: from 1 - very low to 5 - very high)
Dalc	workday alcohol consumption (numeric: from 1 - very low to 5 - very high)
Walc	weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)
health	current health status (numeric: from 1 - very bad to 5 - very good)
absences	number of school absences (numeric: from 0 to 93)
G1	first period grade (numeric: from 0 to 20)
G2	second period grade (numeric: from 0 to 20)
G3	final grade (numeric: from 0 to 20, output target)

Table 1: Initial Dataset Features and their information

3.2 EDA & Feature Engineering

During our initial EDA of the dataset, we dropped G1 and G2 (since we want to focus on predicting G3), used one-hot encoding for the Mjob, Fjob, reason, guardian features, and manually encoded the remaining categorical features where we mapped the classes to 0 and 1. More information can be seen in the code snippet in section B, but we plotted up a correlation heatmap after the initial cleanup, and the visualization can be seen in section A. The heatmap does not

provide any useful information other than the conclusion that G1 and G2 are closely correlated to G3, so we decided to drop features that are between -0.1 and 0.1 correlation to reduce the number of features. However, we also utilized Random Forest Feature Importance (after dropping G1 and G2) with the target variable as G3 for extra validation. The results for this can be seen in the following figure.

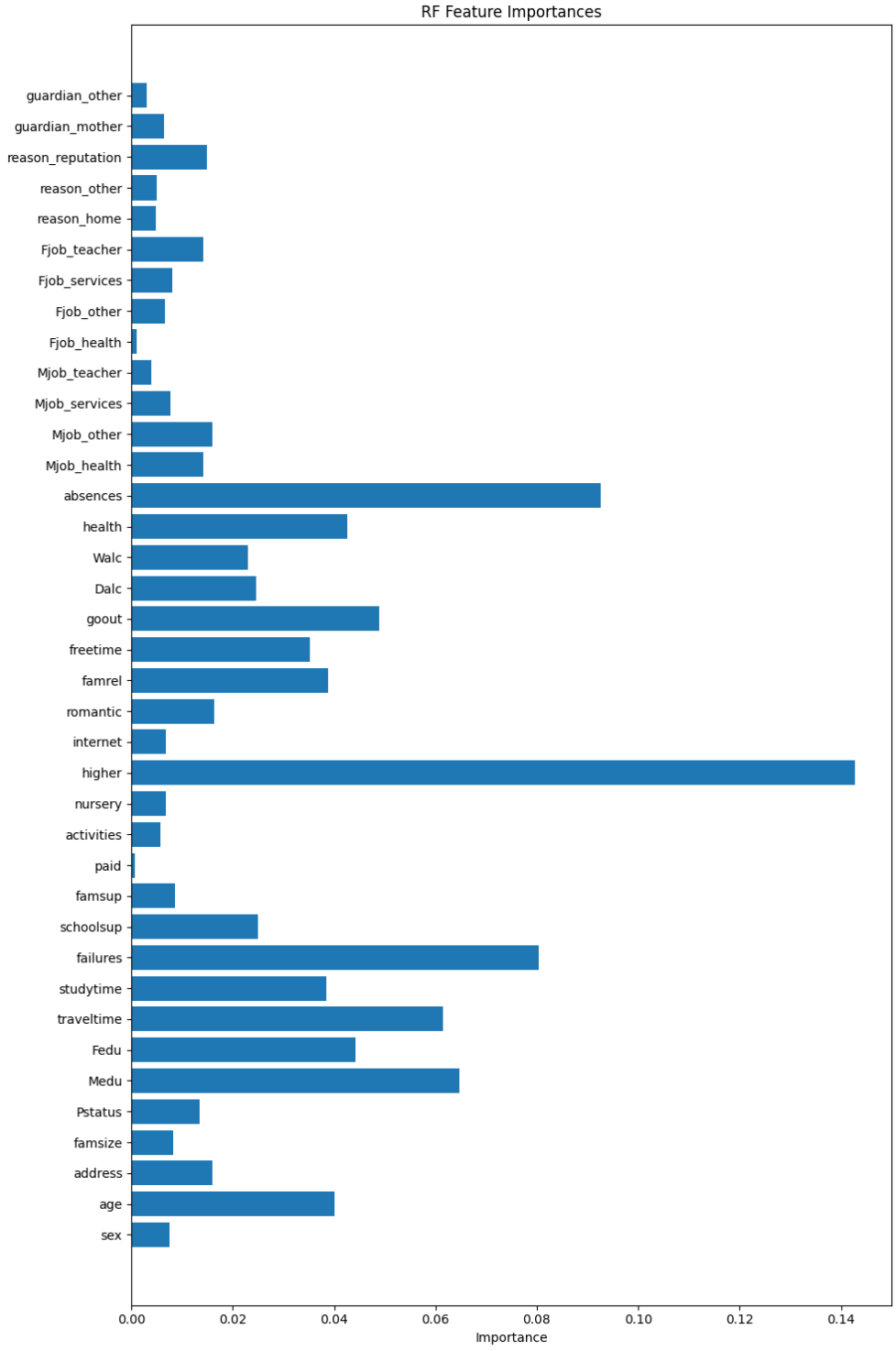

Figure 1: Initial Random Forest Feature Importance when predicting G3

From the figure, we can see that multiple features like higher, absences, and failures are important when predicting G3, but some features are also not as important. Utilizing this

observation, we decided to drop the features were below 0.025 importance, and between -0.1 and 0.1 correlation. All of the features listed below were dropped using the mentioned method:

- sex
- famsize
- Pstatus
- famsup
- paid
- activities
- nursery
- Mjob_services
- Mjob_teacher
- Fjob_health
- Fjob_other
- Fjob_services
- Fjob_teacher
- reason_home
- reason_other
- guardian_mother
- guardian_other

Building on top of this, we also conducted some feature engineering where we combined Medu and Fedu into one feature: Parental Education where we just got the mean for the two features. Other than that, we also engineered another feature called Total Alcohal Consumption where we combined Dalc and Walc with its respective weights (5/7 and 2/7). After all of this, our list of features can be seen in the table below.

Feature	Details
age	student's age (numeric: from 15 to 22)
address	student's home address type (binary: 'U' - urban or 'R' - rural)
parental_edu	Parents' education (numeric: 0 - none, 1 - primary education (4th grade), 2 - 5th to 9th grade, 3 - secondary education or 4 - higher education)
traveltime	home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)
studytime	weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)
schoolsup	extra educational support (binary: yes or no)
higher	wants to take higher education (binary: yes or no)
internet	Internet access at home (binary: yes or no)
romantic	with a romantic relationship (binary: yes or no)
famrel	quality of family relationships (numeric: from 1 - very bad to 5 - excellent)
freetime	free time after school (numeric: from 1 - very low to 5 - very high)
goout	going out with friends (numeric: from 1 - very low to 5 - very high)
total_alcohol	total alcohol consumption (numeric: from 1 - very low to 5 - very high)
health	current health status (numeric: from 1 - very bad to 5 - very good)
absences	number of school absences (numeric: from 0 to 93)
Mjob_health	Mother's job in healthcare
Mjob_other	Mother's job in other fields
reason_reputation	choosing this school due to its reputation
G3	final grade (numeric: from 0 to 20, output target)

Table 2: Final Dataset Features and their information after EDA and Feature Engineering

3.3 Models

Our results for the regression models were not great so we'll briefly touch on them in section 3.3.1, and go over each classification model in detail in the following sections.

3.3.1 Regression Models (1-4)

We tried the following models with their hyperparameters:

- Linear Regression: default hyperparameters
- Polynomial Regression: degree=1 (after utilizing cross-validation, results can be seen in figure 5)
- Decision Tree Classifier: n_estimators=100, max_depth=6
- LASSO Regression: alphas=np.logspace(-3, 3, 100), cv=5, max_iter=50000

The regression models performed terribly with R^2 scores of around 0.1-0.15 and RMSE scores of around 3. With average differences of 3 and so little of the variance being able to be captured by the models, we pivoted to using classification models.

3.3.2 Model 5: Logistic Regression

Hyperparameters - Default

Since we preprocessed our dataset for classification use by splitting G3 into Pass/Fail buckets, we decided that our first classification model should be Logistic Regression, to set a baseline for all models, providing simple and interpretable results with fast training times.

3.3.3 Model 6: Decision Tree Classifier

Hyperparameters - criterion:gini, max_depth:None, min_samples_leaf:1, min_samples_split:2

We picked Decision Tree Classifier after Logistic Regression due to its capability to handle non-linearity, as well as providing a fast training time to serve as a second baseline after Logistic Regression.

3.3.4 Model 7: Random Forest Classifier

Hyperparameters - max_depth:None, min_samples_leaf:1, min_samples_split:2, n_estimators:300

After deciding to move into more complex models with higher training times, Random Forest was the first that we tried since it can help reduce overfitting and is an improvement from single decision trees. It can also help provide feature importance which is relevant to our research question.

3.3.5 Model 8: Gradient Boosting Classifier

Hyperparameters - learning_rate:0.1, max_depth:None, min_samples_leaf:4, min_samples_split:10, n_estimators:500

We wanted to utilize boosting techniques for better predictions at the cost of training time, so the first boosting method we tried was Gradient Boosting with hopes of a higher accuracy and more details on specific patterns in the trees (even though we ended up not finding any observations on this part).

3.3.6 Model 9: Extreme Gradient Boosting Classifier

Hyperparameters - learning_rate:1, max_depth:10, n_estimators:500

As an improvement from Gradient Boosting, we looked into Extreme Gradient Boosting for its faster training time from online sources, and its capability to control overfitting as well.

3.4 Training and Evaluation Methodology

For all classification models, we preprocessed the dataset to increase model performance and convert it for classification use. First, the G3 target is transformed into knowledge_gain where a G3 score of 10 or higher is converted to a Pass and anything less is a Fail. Second, 5th and 95th percentile outliers are removed to increase boosting performance. Last, since the Pass class is oversampled from previous iterations of our models, we use RandomOverSampler to oversample Fail in turn to prevent a class imbalance. We also used scikitlearn's GridSearchCV (code snippet can be seen in section B) for Decision Tree Classifier, Random Forest Classifier, Gradient Boosting Classifier and Extreme Gradient Boosting Classifier to look for the best hyperparameters. As for evaluation, we utilized the classification report and confusion matrix (listed in section 4.1) to measure the model performance and look for any improvements we can implement.

4 Results

In this section, we'll present each classification model's accuracy, classification reports and confusion matrix, provide some analysis and observations we made, and explain the reason behind picking our final model.

4.1 Model Evaluation (for Models 5 to 9)

Some initial problems we encountered from our previous iterations were misclassification of Fail predictions due to class imbalance. We implemented some fixes but ultimately decided that oversampling Fail was the best approach to address this problem. Below are the Accuracy, Classification Reports and Confusion Matrix for all classification models we utilized, along with some observations we made for each model.

4.1.1 Model 5: Logistic Regression

Accuracy: 0.5833

	precision	**recall**	**f1-score**	**support**
0	0.57	0.69	0.62	54
1	0.60	0.48	0.54	54
accuracy			0.58	108
macro avg	0.59	0.58	0.58	108
weighted avg	0.59	0.58	0.58	108

Table 3: Decision Tree Classification Report

	Predicted Pass	Predicted Fail
Actual Pass	37	17
Actual Fail	28	26

Table 4: Decision Tree Confusion Matrix

We tried out logistic regression to compare with other decision tree and ensemble methods, which we will analyze in the following sections, but this model performed poorly compared to them in every metric with a poor accuracy of 58.33% and poor predictions as seen from the Classification Report and Confusion Matrix. It serves as a baseline to the later model's performance, and due to this, we decided to not include Logistic Regression when comparing our final models.

4.1.2 Model 6: Decision Tree Classifier

Accuracy: 0.8981

	precision	**recall**	**f1-score**	**support**
0	0.83	1.00	0.91	54
1	1.00	0.80	0.89	54
accuracy			0.90	108
macro avg	0.92	0.90	0.90	108
weighted avg	0.92	0.90	0.90	108

Table 5: Decision Tree Classification Report

	Predicted Pass	Predicted Fail
Actual Pass	54	0
Actual Fail	11	43

Table 6: Decision Tree Confusion Matrix

We can see that due to GridSearchCV, our accuracy is decently high at around 89.81%. From the confusion matrix, we can also see that our Pass prediction is fully accurate, but the model has some problems predicting Fails correctly even after oversampling it. This is also an improvement from the Logistic Regression model.

4.1.3 Model 7: Random Forest Classifier

Accuracy: 0.9444

	precision	**recall**	**f1-score**	**support**
0	0.90	1.00	0.95	54
1	1.00	0.89	0.94	54
accuracy			0.94	108
macro avg	0.95	0.94	0.94	108
weighted avg	0.95	0.94	0.94	108

Table 7: Random Forest Classification Report

	Predicted Pass	Predicted Fail
Actual Pass	54	0
Actual Fail	6	48

Table 8: Random Forest Confusion Matrix

Comparing to our Decision Tree Model, our accuracy rose to a very high 94.44% with great results from our classification report. It also only misclassified six fails as seen from the Confusion Matrix.

4.1.4 Model 8: Gradient Boosting Classifier

Accuracy: 0.9074

	precision	**recall**	**f1-score**	**support**
0	0.84	1.00	0.92	54
1	1.00	0.81	0.90	54
accuracy			0.91	108
macro avg	0.92	0.91	0.91	108
weighted avg	0.92	0.91	0.91	108

Table 9: Gradient Boosting Classification Report

	Predicted Pass	Predicted Fail
Actual Pass	54	0
Actual Fail	10	44

Table 10: Gradient Boosting Confusion Matrix

Grading Boosting proved to be more effective than Decision Tree Classifier, but still performed worse than RandomForest, with a lower accuracy of 90.74% and misclassifying 10 fails as seen from the Confusion Matrix.

4.1.5 Model 9: Extreme Gradient Boosting Classifier

Accuracy: 0.9352

	precision	**recall**	**f1-score**	**support**
0	0.89	1.00	0.94	54
1	1.00	0.87	0.93	54
accuracy			0.94	108
macro avg	0.94	0.94	0.93	108
weighted avg	0.94	0.94	0.93	108

Table 11: Extreme Gradient Boosting Classification Report

	Predicted Pass	Predicted Fail
Actual Pass	54	0
Actual Fail	7	47

Table 12: Extreme Gradient Boosting Confusion Matrix

Extreme Gradient Boosting provided similar results as our RandomForest model, with an accuracy of 93.52% and misclassifying seven fails compared to the six of RandomForest. This is very comparable to the RandomForest model in terms of performance and accuracy when it comes to predictions.

4.2 Comparative Analysis

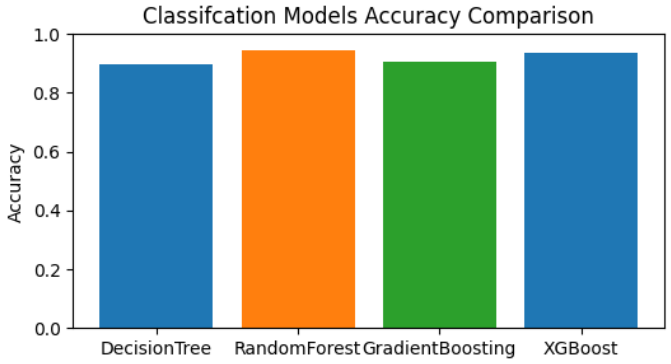

Figure 2: Accuracy Comparison of our four models after dropping Logistic Regression

From the figure above, we can see that RandomForest and XGBoost come out on top as the two highest accuracy, and from our model evaluation in section 4.1, we also observed that RandomForest and XGBoost predicts Fail better than the other two. We'll pick those two and compare the feature importance from both models in the figure below.

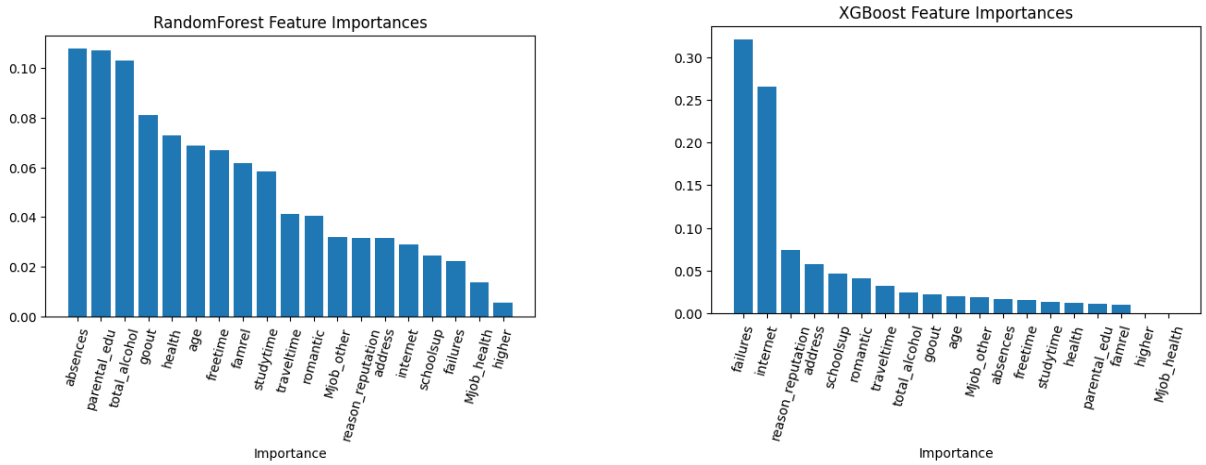

Figure 3: RandomForest and XGBoost Feature Importance Comparison

From the comparison, we can see that RandomForest highlights absences, parental_edu, and total_alcohol as the top three features that stand above the rest, and XGBoost highlights failures and internet as the top two features. With these two observations, we can conclude that both RandomForest and XGBoost give valuable insight into the most important features and the five most important features that has an impact on students' academic performance are the number of failures the student has in the past, internet access, absences, education level of parents, and total alcohol consumption.

5 Discussion

From section 4, we found that the two best performing models are RandomForest and XGBoost, with both providing insightful observations to our research question, what best predicts a student's academic performance. The final five features we decided on are the number of failures the student has in the past, internet access, absences, education level of parents, and total alcohol consumption.

The number of failures the student has in the past makes a lot of sense in this context since that shows the student's performance in the past, but this doesn't really provide much information on what should be improved from a student's perspective, but it does confirm that a student with a poor track record must put in the work to improve. Internet access and education level of parents is a bit more helpful, in the context of how the student learns. Since our dataset relies on secondary school students (where parents tend to tutor their students), it may show that having parents that are more educated can lead to more effective teaching methods that are easier for students to understand. On top of this, having internet access can help a student whether they are self-studying, or when they need information on anything. This might have some downsides especially since the internet can have many distractions and when internet usage is not supervised, using it for learning can become ineffective.

Features like absences and alcohol consumption also makes sense since classes are the primary way for a student to learn course material so missing classes can lead to gaps in knowledge, resulting in lower grades. Alcohol consumption is also a topic discussed widely, especially for college students, where alcohol abuse interferes with a student's sleep, study pattern and can cause addiction. These two negative features can be used to incentivize students to not miss class and not abuse addictive substances.

5.1 Connections to Prior Work

From Cortez et al.'s paper, they found that number of absences, parent's jobs and education, and alcohol consumption levels are important Cortez and Silva [2008]. This matches with up our findings as well, even though we took a different machine learning approach, and shows the importance of attending classes, not consuming alcohol, having access to the internet, and the level of education of parents when it comes to a student's academic performance. Of course, we understand that a student cannot do anything about their parents' education, but with dedication to learning (through classes or online/self-studying) and the willingness to stay away from addictive substances, it can help greatly with their academic performance.

5.2 Limitations

We are aware that this is just using one dataset, with a somewhat small sample size of 382. This may not represent accurate results when utilizing a dataset with different features, but the observations we made can still prove to be useful.

6 Conclusion & Future Work (0.5 page)

In this paper, we answered our research question, "What best predicts a student's academic performance", using different machine learning techniques like regression and classification models. After testing around with many models and techniques(like Cross-Validation, GridSearchCV, Boosting, and Oversampling), we concluded that our RandomForest and Extreme Gradient Boosting models were the best in terms of prediction accuracy and performance. Using the two models, we drew the conclusion(from feature importance in the predictions) that number of failures the student has in the past, internet access, absences, education level of parents, and total alcohol consumption are the most important when it comes to predicting a student's grade. The four features (apart from education level of parents) are what students should be focusing on when aiming to achieving higher grades.

For future work, we recommend taking a similar approach with multiple dataset that includes different features, and perform a comparison of observations made at the end. Comparing the most impactful features and seeing if some of them overlap can help narrow down which ones to focus on from a student's perspective, as well as provide insightful information for academic institutions to help better assist students.

Acknowledgments

We are deeply grateful to our Professor Kelsey Urgo, for all of the assistance in the project as well as Professor Alark Joshi for introducing us to Overleaf so that our report looks as good as it does. We also would like to thank Cortez et al., and Aman Chauhan for providing former research on the field and the dataset that we used for our research.

References

Paulo Cortez and Alice Maria Gonçalves Silva. *Using data mining to predict secondary school student performance.* EUROSIS-ETI, 2008.

A Appendix: Additional Visualizations

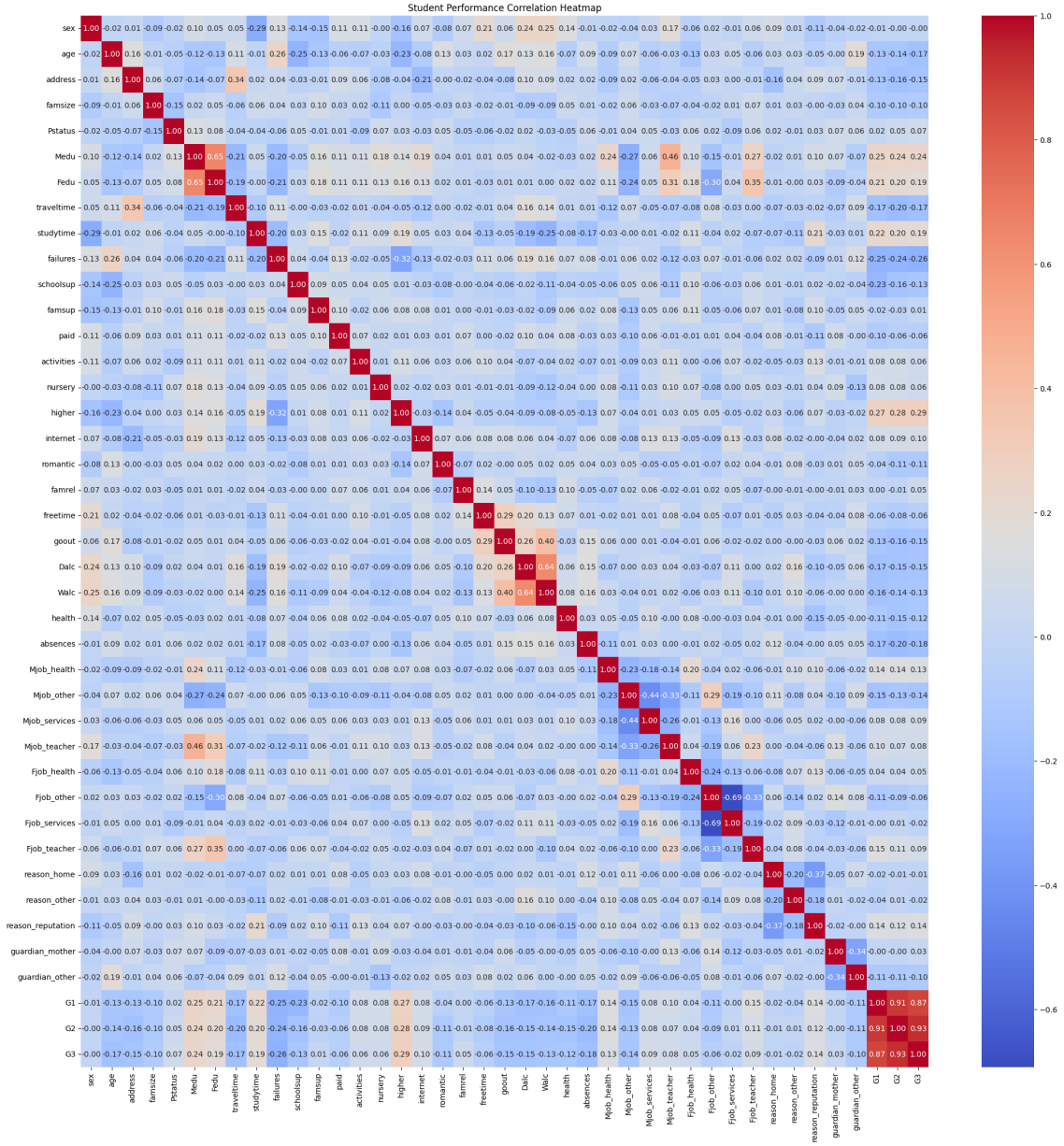

Figure 4: Initial Feature Correlation Heatmap of the dataset

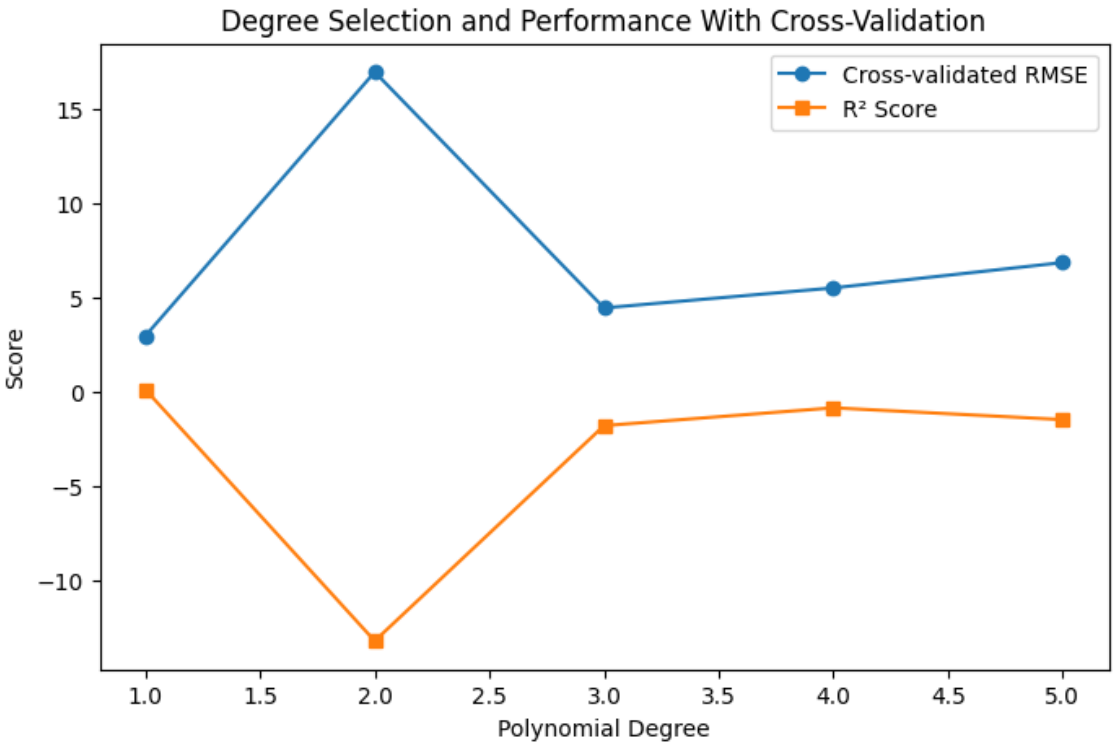

Figure 5: Using Cross-Validation to find best Polynomial degree

B Appendix: Code Implementation Details

```
# One hot encoding
df = pd.get_dummies(df, columns=['Mjob', 'Fjob', 'reason', 'guardian'],
    drop_first=True)
# Converting categorical variables into numericals + dropping
    unnecessary columns
df = df.drop(columns=['school']) # Dropping school since school location
     should not affect much in our scope
df['sex'] = df['sex'].map({'F': 0, 'M': 1}) # Female: 0, Male: 1
df['address'] = df['address'].map({'U': 0, 'R': 1}) # Urban: 0, Rural: 1
df['famsize'] = df['famsize'].map({'LE3': 0, 'GT3': 1}) # Less than or
    equal to 3: 0, Greater than 3: 1
df['Pstatus'] = df['Pstatus'].map({'T': 0, 'A': 1}) # Together: 0, Apart
    : 1
df['schoolsup'] = df['schoolsup'].map({'no': 0, 'yes': 1}) # Extra
    educational support: 0 if no, 1 if yes
df['famsup'] = df['famsup'].map({'no': 0, 'yes': 1}) # Family
    educational support: 0 if no, 1 if yes
df['paid'] = df['paid'].map({'no': 0, 'yes': 1}) # Extra paid classes: 0
     if no, 1 if yes
df['activities'] = df['activities'].map({'no': 0, 'yes': 1}) #
    Extracurricular activities: 0 if no, 1 if yes
df['nursery'] = df['nursery'].map({'no': 0, 'yes': 1}) # Attended
    nursery school: 0 if no, 1 if yes
df['higher'] = df['higher'].map({'no': 0, 'yes': 1}) # Wants to take
    higher education: 0 if no, 1 if yes
df['internet'] = df['internet'].map({'no': 0, 'yes': 1}) # Internet
    access at home: 0 if no, 1 if yes
df['romantic'] = df['romantic'].map({'no': 0, 'yes': 1}) # With a
```

```
   romantic relationship: 0 if no, 1 if yes
```

Listing 1: Encoding Code Snippet

```
param_grid_dtc = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [None, 2, 5, 10],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
}
param_grid_rfc = {
    'n_estimators': [50, 100, 300, 500],
    'max_depth': [None, 2, 5, 10],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
}
param_grid_gb = {
    'n_estimators': [50, 100, 300, 500],
    'max_depth': [None, 2, 5, 10],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'learning_rate': [1, 0.1, 0.01, 0.001],
}
param_grid_xgb = {
    'n_estimators': [50, 100, 300, 500],
    'max_depth': [None, 2, 5, 10],
    'learning_rate': [1, 0.1, 0.01, 0.001],
}
# Find the best hyperparameters for every model
gs_dtc = GridSearchCV(DecisionTreeClassifier(random_state=42),
   param_grid_dtc, cv=5, n_jobs=-1, scoring='f1')
gs_rfc = GridSearchCV(RandomForestClassifier(random_state=42),
   param_grid_rfc, cv=5, n_jobs=-1, scoring='f1')
gs_gb = GridSearchCV(GradientBoostingClassifier(random_state=42),
   param_grid_gb, cv=5, n_jobs=-1, scoring='f1')
gs_xgb = GridSearchCV(xgb.XGBClassifier(random_state=42), param_grid_xgb
   , cv=5, n_jobs=-1, scoring='f1')
```

Listing 2: Using GridSearchCV for best hyperparameters for Classification models