Hasnain Shahzad
Final Project Report
CS439
https://github.com/Hasnain3201/CS439-Student-Performance-Project

My project idea is to build a model that can predict a student's final grade. It does this by using the prior information already available to us in the school year. Some of which include the students' performance in class, their punctuality or how often they are present, as well as important data related to their personal and family factors. For this, the Student Performance Dataset from the UCI Machine Learning Repository was selected for its real-world application of real student data. This dataset contains real data from high school students in Portugal. In this dataset, each row is a student, and the columns for each student are: study time, absences, parents education level, did they get extra academic support, how they rate their relationship with their family, lifestyle habits, and lastly, their grades at G1 G2 and the final grade, G3. In this dataset, the primary goal is to use the information in each column to predict the final column which is G3. I also made it a goal to avoid using G2 if possible, since this would allow for early intervention before it is too late. This project will make valuable use of everything covered in CS439 such as data cleaning, exploring data and removing outliers, engineering new features and testing out a variety of models with cross-validation.

This project idea arose from something I gained a better understanding of. Which is that schools are collecting a large quantity of data about their students, such as attendance, grades and background information, however a lot of this data is sitting there without being used in any useful way. Hence, there is a real potential to make use of that data in a meaningful and systematic way, especially to support and help students ahead of time, before they fall too far behind in class. If it's possible to predict the G3 grade at the end of the year using prior information (the columns other than G3), then teachers can intervene much sooner with extra support such as private tutoring and regular performance checks. Moreover, a predictive system like the one described above is not intended to 'label' students or make unfair judgements. However, it should be objective while allowing us to prioritize betterment of the students' experience, including those who are silently struggling with their learning. Knowing that a system like this can be sensitive, it is important to not blindly accept arbitrary numbers without an explanation. For that reason, I chose to use a simpler and more transparent solution like linear regression. Linear regression shows you exactly how each factor, like study time and absences, affects the final prediction. This avoids the trap of being left in a "black box" and allows us to interpret the result using the same tools that were utilized in CS439. This enables a perfect balance between accurate model prediction and clear explanations.

Most machine learning techniques in the modern-day make use of newer and complex models such as large ensemble methods that include neural networks or even transformers. These are effective in grand-scale projects that operate with a large volume of images or text, and where the data is extremely messy and high-dimensional. But in our case, the dataset is smaller in size, and it is neatly structured with clearly labelled columns and an output column. Therefore, a high level of complexity isn't needed for this scenario, and as previously described, more complex models are prone to create a "black box" where outputs are harder to explain and justify. This is particularly important in the professional context of education, where teachers need to act quickly on results while being confident in their justification. In addition, CS439 makes it clear that a simple machine learning model used correctly with thorough data preparation is sufficient for datasets like these. As a result, rather than forcing negligible gains from more complex models, I have decided to build something that is just accurate enough to be helpful, which is why I settled with a linear model.

Before data cleaning and data preparation, I first needed to improve my understanding of the dataset before adding any changes. It had 649 students and 53 variables in total which was more than enough to work with. Fortunately, there was no missing data in the whole set, which makes data cleaning slightly easier. This means there would be no need to impute data and make guesses that could potentially affect the final output. The first task was to organize the data by splitting them into numerical and categorical. The features that were numerical included age, absences, self-rated health, alcohol consumption on weekdays or weekends, past class failures, and then finally the three grades including the final G3 grade. The rest of the features were categorical and were related to information like gender, school, whether they live in an urban or rural area etc. Before making any decisions on data preparation, I also had to look at basic statistical information to detect any important patterns. One thing I noticed is that final grades were more clustered in the 0-20 range, and most students don't miss any days. There were only a few students that had a significant number of absences if there were any absences. Another factor that was likely to affect the output is that some schools have significantly more students than the other. All of this information is vital to think about what features have the most predictive factors to them, and which features are just extra noise that don't affect the output very much.

EDA (Exploratory data analysis) is a key step for identifying patterns and outliers in the dataset before doing any processing, as well as other techniques. Which allows us to take advantage of this data for better model output. After getting comfortable and understanding what each column meant, I was able to dive straight into EDA. This involves producing useful visualizations that give deeper insight into the dataset over just looking at the numbers alone. The first visualization I made was a histogram that contained the most important numerical features such as the final grades, study time, and absences. The distribution for the final grade G3 was smooth, it was clustered around the low to mid-teens, with less students at the extremes of 0-20. Study time was variably encoded as 1 to 4, and most students reported that they studied for just one or two hours, and fewer students that were three to four hours. This made reasonable sense, since not every student is likely to log long study sessions. Likewise, the histogram of absences also stood out. The majority of students only have a few days missed, however there was a long tail of a small few students who had extremely high absences. These more extreme cases were clearly outliers which could potentially skew the results or hide important patterns in the model. Which reminded me that real-world data is messy, it's usually never perfectly balanced which makes data cleaning a highly important task.

To get a better sense of how the variables are all connected, I calculated a correlation matrix for all the numerical features and turned it into a heatmap. The strongest connections were between the grades G1, G2 and G3 themselves which was to be expected. Also, G2 and G3 were especially close, which is also to be expected. Which is because of the fact that a student's performance just before their finals is the closest indicator of their final outcome. Likewise, G1 was also directly correlated with G3, but not as much as G2. Another predictable link was the number of absences. There was a negative correlation between absences and G3, which meant that more missed days tend to go with lower scores which is similarly a natural expectation and proves that the data makes sense. However, other features like self-reported health or weekend alcohol use barely had any impact on G3, since there was almost no correlation at all. Which suggests that these less important features probably won't be major variables in the model. This helped me start forming expectations, I expect that G2 would end up as the strongest predictor in the regression model, and absences may have a smaller but still important effect.

In addition to the numbers in the correlation matrix, I made a scatter plot to produce a better visualization of how the data behaves. Looking at G1 to G3 and G2 to G3, both displayed a clear upward trend which is to be expected. Although the G2 to G3 plot stood out, specifically the plot formed a tighter band of points, meaning the relationship was stronger and less noisy than G1 to G3. Which proves that G2 may be the best possible predictor on the final grade. The relationship between absences to G3 was also checked, which produced a downward trend or a negative correlation, but the spread of data was wide. This demonstrated that some students had a high number of absences, but they still passed fine in G3. There were also students who had perfect attendance but still didn't perform as well as the data would usually predict, suggesting there were some other external factors that changed this outcome. It is obvious that absences do not give a complete picture, even if they do matter in a school setting. I also used boxplots alongside scatterplots to explore the relationship between study time and final grades. This was done by grouping G3 by each study time category (1 to 4). On average, the data suggests that grades do go up with more study time, but the boxes still overlap a lot. This could mean various things, perhaps some students were just more efficient in their learning, or they had a lot of prerequisite studying before the class, while others may be putting in the right amount of hours but without getting much done in those hours. Similarly to absences, hours studied is not enough to be a strong predictor. However, how these features interact with each other may reflect more valuable information, which indicates that a step for feature engineering may be required.

The above insights helped shape the next part which is the feature engineering part of the project, which turned out to be the most powerful step in the project as well as being a big focus in CS349. I built a few new features that attempt to combine academic and environment data in a smarter way. One of the features measured the effectiveness of study effort relative to attendance. It does this by taking the study time and dividing it by the number of absences plus one. This was because of the insight that a high amount of studying hours may not be spent effectively if the student isn't attending classes. This ratio in the calculation rewards students who both show up and put in the time, and it downplays students who have high study hours but miss a lot of classes. Another feature blends G1 and G2 into a single performance score. This basically "averages" the early term grades and helps to smoothen out noise and handle cases where students had a bad day. The third feature is G2 minus G1 which is intended to track change. A positive number obviously means the student is improving, while a negative number means they did worse in G2 and zero would suggest they are steady. And so a positive change suggests that the student is more likely to finish with a strong G3 grade.

In addition to the above engineered features, I also considered grouping absences into different categories like low, medium and high absences, as well categories for lifestyle indices which would make use of features like alcohol consumption. However, experimentation on these features proved that they did not demonstrate any noticeable improvement in the models performance. I also thought about the factor that higher parental education should affect the students' performance, such as their tendency to have higher study hours. Likewise, after experimenting with this interaction, the performance gains were also negligible which did not justify the extra complexity. This taught me that not every engineered feature will add value to the model, some ideas work while others do not. Although I was only able to discover their usefulness from trial and error, some feature engineering requires experimentation and testing while others come from logical intuition and tend to work.

After defining the above engineered features, I was ready for a preprocessing pipeline using scikit-learn's 'ColumnTransformer' and 'Pipeline'. Categorical variables were all one-hot encoded, which means they have binary indicator columns. In comparison, the numerical features need to have zero mean and unit variance through standardization. A large problem is that solvers used in linear models are sensitive to feature magnitude or the scale of a feature, so standardization is particularly necessary here to ensure that a single variable does not dominate the others after scaling. This pipeline structure ensures that a consistent transformation is applied to the training and test data, and prevents data leakage during cross-validation. It also allows for modularity, so swapping models back and forth for example would not require rewriting the whole preprocessing logic to support the other model. Which makes experimentation much easier and faster while keeping the entire workflow more maintainable, this is useful when expanding the project to test new techniques and strategies.

After the feature engineering step, I was able to isolate the rows into both training and testing sets. Whichever model is picked in the pipeline is only fitted and trained on the training set, while the testing set is held for evaluation towards the end. For this scenario, I chose to dedicate 80 percent of the data to training and 20 percent for testing. Moreover, it is normal to use various other splits like 75/25 or 70/30, however, what I chose is normal practice for general use. The training set is used for model selection and 'fitting' that model, while the test set is used at the end for final evaluation. It's important to do it this way, because training a model on the entire dataset can lead to overfitting where it is essentially 'memorizing' all the patterns rather than predicting it. Reserving unseen data is crucial to test whether the model can generalize outside of its training data. In the modelling stage, four models were trained to reflect different levels of complexity and regularization that was covered in CS439. The first being Linear Regression which serves as the "baseline" and connects directly to the closed form solution. The second is Ridge Regression, which adds L2 regularization to penalize large coefficients and deals with multicollinearity. The third is Lasso Regression which uses L1 regularization to enforce sparsity by shrinking some of the coefficients to zero. This is useful for improving interpretability, or selecting a subset of features. Finally the last model is Gradient Boosting Regression. A tree based method that is nonlinear, which is useful for highlighting interactions and patterns that linear models usually cannot capture. However, the cost of a tree-based model is that it's much harder to interpret.

Using the above models, all four were evaluated using mean absolute error (MAE) and the R^2 score. MAE was interpretable in context, meaning it reported average prediction errors in correct grade units. While R^2 indicates how much of the variance in G3 is explained relative to a baseline that predicts the mean. The most performant model was Lasso Regression with an alpha of 0.1, achieving a test MAE of approximately 0.71 and an R^2 value of 0.87. Which means that predictions are on average within 0.71 points of the true grade and that the model accounts for 87% of the outcome's variability. Moreover, to assess the stability of the model, I ran a five-fold cross-validation on the training data. Which also produced a strong performance for Lasso with a mean MAE of 0.79 and low variance across the folds. Similarly, Ridge and Linear Regression did similar in performance but underperformed slightly compared to Lasso. Gradient Boosting also showed no meaningful improvement over Lasso despite its much greater tree-based decision making complexity. This demonstrated the central lessons from CS439, which is that good feature engineering and data cleaning can go a long way when paired with a simpler linear model, and may even outperform the more complex and nonlinear models in comparison.

After compiling the results from above, Lasso was selected as the final model. In addition, I conducted diagnostic checks to assess its reliability. Firstly a scatter plot of predicted versus actual G3 values on the test set showed points tightly clustered around the diagonal line, which confirms a strong agreement across the full grade range. This aligns with the low MAE and high R^2, and visually reinforces that performance is consistent at both high and even lower grades. I also conducted a residuals vs predicted values test where the plot showed no significant pattern. Residuals were centered near zero and were evenly distributed. The residual plot didn't show any clear curvature, which suggests a linear relationship is reasonable and adding new transformations probably would not improve the performance. There are also no extreme outliers or consistent patterns in the errors themselves, which suggests that the model isn't underfitting or overfitting certain groups or ranges of students. The residual distribution is also roughly symmetric as above, no long tails or isolated points which also further proves that there's no extreme outliers.

One of the primary reasons I chose Lasso was to retain interpretability through the model's coefficients. I tested this in the training phase, where I inspected the weights to assess feature importance. G2 had the largest coefficient which was to be expected, it was approximately 1.95. A one point increase in the second-period grade is associated with a nearly two point rise in the predicted final grade G3. This strong effect also aligns with the high positive correlation that was seen earlier and the idea that recent academic performance is clearly going to be a powerful predictor. In addition, the engineered feature that combined G1 and G2 also carried a positive weight, suggesting that we could draw extra information without relying on G2 alone. The study time to attendance ratio showed a smaller but still a positive coefficient. This supports the above idea that consistent engagement with the lessons where the student both studies and shows up has a noticeable impact on the final grade. Lastly, the G2 - G1 engineered feature was positively weighted too. Which suggests that upward momentum is a good predictor of G3. However, past class failures and absences did show a negative coefficient. Many of the one-hot encoded categorical features resulted in the coefficients being near zero because of Lasso's built-in feature selection. This informs us that once things like academic performance and effort is accounted for, the demographic and other categorical factors do not hold as much predictive power.

Given the sensitivity in the context of education decision-making. I also conducted an analysis by evaluating the model's performance across different student subgroups, especially by sex and by school. For each group, I calculated both the MAE and MSE (mean signed error). The MSE is used to detect potential directional biases. The results from both of these computations showed slightly lower MAE for female students compared to male students, though both of them were below 1.0. Signed errors were small in magnitude, which indicates that there was no significant tendency to over predict or under predict for either group. Overall, the differences were relatively small compared to the scale of the final outcome. A similar comparison across the schools showed there was no meaningful disparity in the rate of error. The model performed consistently for both. This type of analysis is simple but still useful, it provides an initial check for unequal performance among different groups which is essential to check before deploying the model. In addition to these checks, it could be extended for other things like demographic detail or parental education or other socioeconomic factors. The largest benefit of using a linear model is full transparency, since the coefficients in a linear model are explicit, it can immediately reveal information about that group (like school or sex).

Looking back at the original project proposal, the path that was set out was followed quite consistently. I used the student performance dataset and conducted EDA, feature engineering while discarding feature interactions that didn't work. I focused particularly on linear models and made use of good coefficients like academic progress and engagement. The final result of the project did go beyond the initial scope. In particular, there was additional work done in fairness checks and residual diagnostics, while they were not fully detailed, they went beyond the initial plan. I also had originally considered implementing gradient descent to understand the theory behind it, but chose to select well-tested libraries like scikit-learn instead. This was so more time could be dedicated to comparing models and different cross validation strategies to save time since this approach was more well-trusted and reliable. Although nothing was coded from scratch, the Lasso model still operated very well on the same core principles covered in CS439 where the theoretical foundation was still central to the whole project. In the end, this final product was a natural extension to CS439 as it applied the same decision-making concepts but on real-world data.

In conclusion, this project demonstrated how the core ideas from CS439 translate into real-world application and practice. One of the primary goals was to build a useful tool that is interpretable and contextual, while also being accurate in its output. I started with just raw data and worked through multiple essential steps like cleaning the data, producing useful visualizations and engineering features. This allowed me to identify useful patterns like academic momentum and consistent effort and showing up to class. This useful insight allowed me to train and compare a wide variety of models that fit this particular scenario. Ultimately I chose to use Lasso Regression for its strong performance and built-in sparsity. The experimentation with Lasso allowed us to achieve a low MAE of approximately 0.71 and a high R^2 with approximately 0.87. Moreover, the use of a linear model allowed for full transparency and clear interpretation of how each feature influenced the final prediction. From analyzing feature importance, we found that G2 dominated in importance, confirming that recent performance was the strongest indicator of final outcome. In addition, features like study time to attendance ratio had a smaller but still useful signal. I conducted additional diagnostic analysis to validate the model. Residual analysis proved there was no systematic bias across predicted values. Fairness checks between different schools and different sex didn't show any significant disparities in rate of error, which is vital to consider in the context of education where equity matters. While the project proposal focused primarily on aligning theory to code, the overall process did evolve towards choosing robust evaluation methods and meaningful features from intuition, as well using transparent and well-trusted methods over needlessly going for algorithmic complexity. The biggest take away from this project was that real-world applications require more than just accuracy, it is also about doing responsible data science, making fair and informed decisions based on the context of work.