Phase 2 – Ansh Patel

REPORT

In this project we use a dataset of student performance to determine whether a student is at risk academically, meaning they will achieve a final grade below 10. The data in our set includes demographic information, behavioral factors, family background, school support indicators, and previous grades. In Phase 1, I debugged and completed a baseline machine learning pipeline, fix data preprocessing and a Gradient Boosting pipeline, implemented a Random Forest from scratch, and evaluated my model using real metrics such as F1 and ROC-AUC. In Phase 2, I was working off of phase 1 to improve the performance and accuracy through testing different scope items. Here I was able to use the full dataset, reduced dataset dimensionality, and included stacking to improve results from the dataset to better accurately predict at risk students.

Phase 1 started with creating a data loading function that would be able to read the student dataset from different file locations. To ensure that the model could access the file from different areas, the function I created try to load through the file name, through the path, and with a backup to the full dataset if the mini dataset cannot be found. It then looks for separator punctuation such as commas and semicolons. The purpose of the preprocessing is to turn the raw data the model is fed from the csv into numbers with values for which the model will learn from. It does this by creating a binary target variable “at_risk”, it also removes grade columns to prevent label leakage. In this we also separate numeric and categorical variables, takes the median for the numeric columns, does min-max scaling and more. This portion of my project is one of the most vital as it turns the raw data into useful information for the model.

I also implemented two additional helpers, one which gives summary statistics which returns basic metric such as mean and median absences, and a correlation computation function that returns a numeric correlation matrix. I also implemented a simple Gradient Boosting pipeline using sklearn. This pipeline contained a passthrough ColumnTransformer for flexibility and a GradientBoostingClassifier as the estimator. This baseline was the simplest model in the project which allowed validation of correctness of preprocessing and data loading.

Next I also implemented a Random Forest classifier from scratch. To create this I stated off with building a DecisionTree class with Gini impurity calculations, best split selection over various features, recursive tree construction, and majority vote leaf creation. From this the RandomForest class was created with bootstrap sampling, training different instances of DecisionTree and a final predication that is created through majority vote. After implementing all the required functionality, Phase 1 after being passed through the auto grader met all requirements and passed the 18 tests.

Phase 2 is where I was able to further improve upon the project to increase accuracy by choosing different scopes to add. The scope items that I decided to add were to use the full dataset, perform feature selection, and use stacking.

For the phase 2 experiments, I switched from the 39 row mini dataset to the full 395 row student dataset. The larger dataset makes sense because it would provide better results and support better feature selection simply because there are more items for the model to work and learn from. I used the same preprocessing steps from phase 1 for this portion of the project. The baseline gradient boosting performance with the full dataset was used to compare with my improved results later on. The baseline accuracy is 0.681, F1 score is 0.406, and ROC AUC is 0.675.

The second enhancement used SelectKBest with F-scores to select the top 20 features before training the Gradient boosting model. Doing this helped reduce noise, lower dimensionality and highlight the most predictive variables. With this enhancement my results were as follows: accuracy is 0.739, F1 score is 0.523, and ROC AUC is 0.727. This was the strongest model across all tests as it improved the predictions and reduced dummy features. Feature selection improved the performance because many of the encoded variables that just added noise without contributing to the prediction were limited. Thus reducing the dimensionality helping the Gradient Boosting focus on the core signals in the dataset.

The third enhancement I include in this project was a stacking mechanism that combines the base models from GradientBoostingClassifer and RandomForestClassifier with the meta learner of LogisticRegression to create an overall stronger and more accurate model. This allows the model to combine the strength of multiple learning methods to overall improve the results. With stacking my results were: accuracy is 0.723, F1 score of 0.377, and ROC AUC of 0.686. Compared to the base line numbers we have, my stacking model was able to improve on the ROC AUC score. However, the F1 score was lower than the feature selected model meaning the meta learner may have smoothed the decision boundary in a way that reduces minority class recall.

Model Comparison

Model	Accuracy	F1 Score	ROC AUC
Mini Dataset Phase 1	0.583	0.286	0.469
Gradient Boosting on Full Dataset	0.681	0.406	0.675
Gradient Boosting with Top 20 Features	0.739	0.523	0.727
Stacking Ensemble	0.723	0.377	0.686

When comparing the different accuracies with the different updates from phase 2, we can see that the feature selection gave us the strongest results with the highest scores across accuracy, F1 score, and ROC-AUC. The Stacking ensemble enhancement also did improve accuracy and ROC-AUC but falls short because of its lower F1 score. Using the full dataset also significantly improves our results as compared to the baseline stats we have from the mini dataset.

Phase 2 expanded the analysis beyond the basic modeling done in phase 1. When I used the full dataset compared to using the mini version, the model had better training and evaluation because of a greater number of samples. Feature selection gave me the best results which shows the value of reducing the categorical variables in the dataset.

Reflection

What was your favorite part of the project?

My favorite part of the project was building everything out from start to finish and being able to see the improvement of performance from the models as I added new scope items and changes in the project. I was able to build every part of the pipeline myself. After implanting feature selection, I was able to actually see the improvement in performance compared to the baseline and I was able to understand why it did so. I also enjoyed how hands on the Random Forest implementation were in phase 1. Implementing things such as the Gini impurity, recursive splitting, and bagging gave me a deeper appreciation for what occurs behind the scenes in many algorithms. It was great to see my own implantation succeed and pass.

What was your least favorite part of the project?

My least favorite part of this project was debugging during the early stages of Phase 1. It took me a decent amount of time to figure out the preprocessing function to get it work as intended. I also had a few mismatches in data types and missing values which was hard to track down, as they were easy to overlook. Part of this seemed very tedious at times as sometimes the test does not always tell the user exactly what the issue is. So it took me a good amount of time to proofread through the pipeline and figure out what corrections were required.

Similarly in the Random Forest, the implementation for every tree operation manually was difficult to debug. The concept itself was very interesting, but when there were slight issues it was hard tell where it was coming from. However, overall I really enjoyed this project and learned quite a deal from this, but for me the debugging of phase 1 was the most frustrating portion if I had to pick out something.

On what topic from the course did your perspective change the most from before to after the project? Did your opinion of the usefulness of this topic go up or down after working with it within the context of the project?

The biggest change in perspective for me from before and after was with the feature preprocessing and feature selection. Before the project, my general understanding of the concept was decent at best, but I was able to learn much more about it through this project and what it has to offer. I thought preprocessing was a normal step that you just have to do. However, I did not realize how much the quality of preprocessing influences the rest of the results. After working through it and seeing the difference from the top 20 feature model I made, I was able to understand how big of an impact good preprocessing and dimensionality reduction have on performance.

My opinion of the usefulness of feature selection definitely went up. At the start I thought using all available features would give the model more information and thus better performance since there is more content for the model to work with. After seeing that selecting feature led to my best performing model in my experiments, I realized that extra feature can introduce noise and extra complexity.

If you had more time, which scope item (or item from phase 1) that you implemented would you wish to improve the implementation of further and how might you try to do this?

If I had more time, the scope item that I implemented that I would like to improve the further would be the stacking ensemble. Even though in my project it increased the ROC AUC, the F1 score dropped compared to the feature selected model. I believe that the meta learner did not have enough information or did not have the best choice for the data size, so I would want to mess around with a different meta model or tune the base learners before stacking. It would be interesting to see whether a tuned version of stacking would have been able to outperform the top 20 feature model instead of slightly improving the performance. My implantation definitely showed promise, so it would be great if I could improve the ensemble to have greater accuracy overall.

If you had more time, what new scope item/ techniques would you like to try to apply and why?

If I had more time I would like to try to apply Bayesian networking learning scope. Bayes networks are really interesting conceptually and would be cool to see how implementing this would influence the results. The student dataset we are working with has many different factors and a Bayes net would be able to show the different linkages between each to see how each piece fits together and what variables would more likely influence the at-risk category. Instead of seeing whether a model predicts at-risk or not, a Bayes net could help explain why a student is likely to be at risk by showing the probabilistic connections in the graph. This would be able to add a realistic layer for the student performance prediction. This approach definitely would have taken a considerably more time, but it would definitely be a cool and valuable addition to the project that adds to both the insights and a different modeling perspective.

GIT Repo

https://github.com/Fall25-CS5100-JesseAStern/final-project-fall-2025-ansh