Student Performance Analyzer

Ishan Kumarasinghe
Dept. of Computer Engineering
University of Peradeniya
Peradeniya, Sri Lanka
e20211@eng.pdn.ac.lk

***Abstract*—Predicting student academic outcomes from demographic, behavioral, and past performance data can facilitate early interventions. This study integrates the UCI Student Performance Datasets (Mathematics and Portuguese) for a combined analysis. We preprocess and merge the two datasets (382 overlapping students), train models to predict $G3_{\text{Math}}$, where prior period grades ($G1, G2$ in both subjects) are key features, and train using Random Forest and SVR regressors. After tuning, a bagged set of Random Forests significantly improves performance, achieving $R^2 \approx 0.917$, RMSE ≈ 1.358. We discuss data pre-processing, feature importance, feature selection, feature extraction, evaluation metrics, ensemble learning, and the value of cross-validation.**

Index Terms— Educational data mining, regression, ensemble methods, bagging, student performance

I. Introduction

Student performance prediction is critical in educational data mining for identifying at-risk learners. We leverage two UCI datasets (Mathematics and Portuguese) containing demographic, behavioral, and academic records[1][2]. Our target is the final Mathematics grade $G3$.

The datasets are integrated to increase sample size and diversity, followed by thorough preprocessing, exploratory data analysis, and feature engineering to prepare the data for modeling. We train and evaluate multiple regression models, focusing on Random Forest and Support Vector Regression (SVR), and explore ensemble learning techniques such as bagging, voting to assess whether they can improve predictive performance. Model performance is evaluated using R^2 and RMSE, with results compared to identify the most effective approach.

II. Dataset Description

The UCI repository provides two separate datasets (Math: 395 records; Portuguese: 649 records), each with 33 variables including two period grades $G1$, $G2$, and final $G3$ plus socio-demographic features. We merge these on 13 identifiers (e.g., school, sex, age, address, parental education), resulting in 382 overlapping students. The merged dataset retains one copy of common features and includes both sets of grades ($G1/G2/G3$ for each subject).

III. Data Preprocessing

After integrating the Mathematics and Portuguese datasets into a single working dataset, the combined data was examined for noisy values, inconsistencies, and redundant columns. Redundant or duplicate features were consolidated or removed to avoid unnecessary complexity and multicollinearity.

Categorical variables (e.g., school, sex, job, reason) were transformed using One-Hot Encoding to create binary indicator variables for each category. Binary fields (e.g., address, famsize, Pstatus) were mapped directly to 0/1 for simplicity and consistency.

The original datasets contained no missing values, so no imputation was required. However, numerical features were inspected for scale differences. For algorithms sensitive to feature scaling - particularly Support Vector Regression (SVR) - numeric features were standardized using StandardScaler to ensure all features contributed proportionally to the model. Portuguese $G3$ ground truth was excluded from final model inputs to prevent leakage.

IV. Exploratory Data Analysis

Histogram of Math $G3$ reveals central clustering (11/20). Heatmaps indicate strong correlations: $G2_{\text{Math}}$ (0.90) and $G1_{\text{Math}}$ (0.70) correlate with target. Portuguese $G1, G2$ show moderate correlation (0.3–0.4). Features like absences and failures correlate negatively. These insights guided us to include period grades from both subjects and exclude low-impact socio-economic variables.

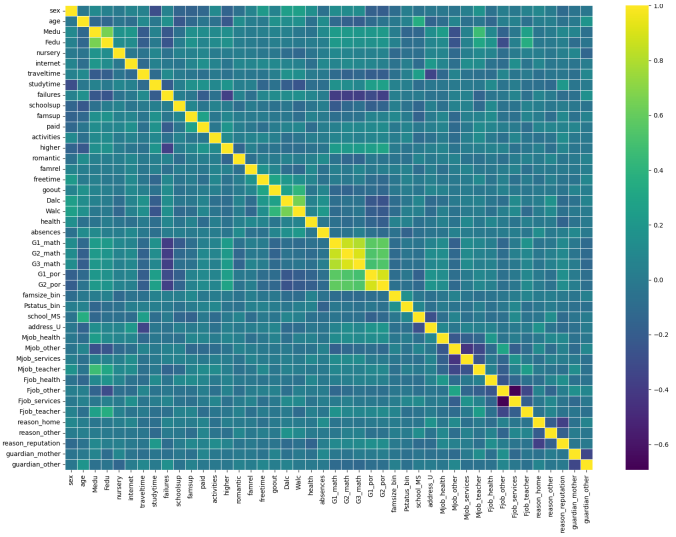

Fig. 1. Correlation heatmap of encoded features.

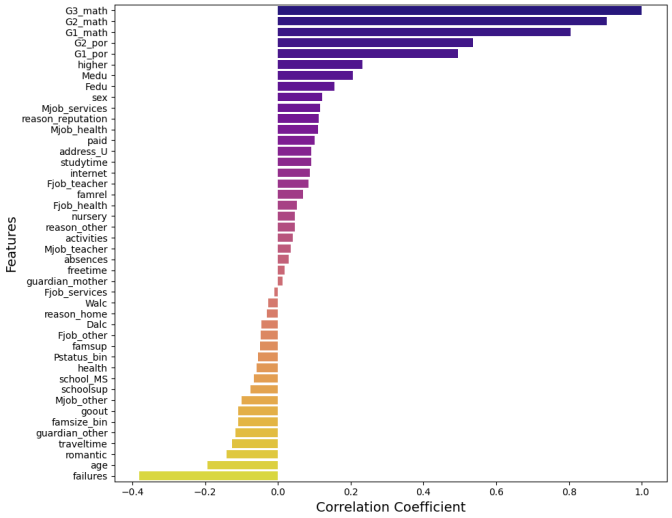

Fig. 2. Feature correlation with Math $G3$.

V. Feature Selection

Feature selection is approached in two stages: an *Open-loop* filtering stage and a *Closed-loop* model-driven stage. This combination ensured that the retained features were both statistically relevant and empirically validated for the predictive task.

In the *Open-loop* stage, we applied a univariate statistical test using *SelectKBest* with the *f_regression* scoring function. This method ranks features independently of any model training loop, based solely on their individual linear relationship with the target variable (G3 in Mathematics). Because the model does not "feed back" into this process, it is fast, simple, and easy to interpret. At this stage, we experimented with different values of K to see how many top-ranked features might be optimal.

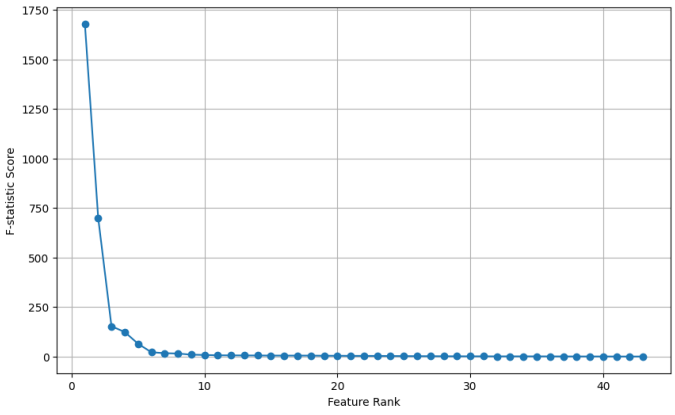

Fig. 3. Elbow curve for optimal K.

In the *Closed-loop* (wrapper method) stage, we evaluated the predictive performance of the top-k feature subsets using a *RandomForestRegressor* within a cross-validation loop.

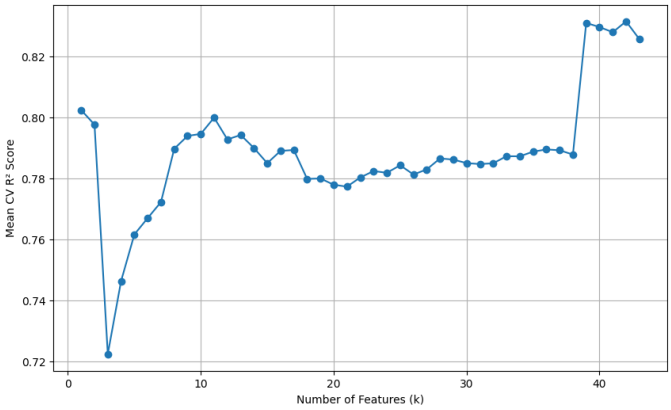

Fig. 4. Cross-validation results for optimal K.

For each candidate k, the model was trained and evaluated using 5-fold cross-validation, and the mean R^2 score was recorded. This approach allowed the model's performance to directly influence which features were ultimately kept, capturing non-linear relationships and interaction effects that the open-loop method might miss.

When analysing both Open Loop method and Closed loop method outputs it gives around (7-11) features which have highest feature score that is enough. For a better model we decide to apply Feature Extraction also. So, we select (12-14) features out of 44 (after encoding) features.

Through this process, the final selected features are:

[G2_math, G1_math, G2_por, G1_port, failures, higher, Medu, age, Fedu, romantic, traveltime, sex, guardian_other, Mjob_services]

These features represent a mix of strong academic predictors (e.g., prior grades), behavioural indicators (e.g., study time, failures, internet access), and socio-demographic attributes (e.g., parental education, job, guardianship).

VI. Feature Engineering

Following feature selection, additional transformations and derived variables were created to enhance the predictive power of the dataset. The goal of feature engineering is to capture relationships and patterns that may not be directly visible in the raw data, while ensuring that all features remained interpretable and relevant to the educational domain.

Previously, categorical variables are encoded in a form suitable for machine learning algorithms. To reduce dimensionality after one-hot encoding and to address multicollinearity among correlated variables (e.g., prior grades, attendance-related features), we applied *Principal Component Analysis (PCA)* as the core feature engineering step.

Prior to PCA, numeric and encoded features were standardized to zero mean and unit variance to ensure that component extraction was not dominated by scale differences. PCA is fit strictly on the training data to avoid leakage, and component selection was guided by cumulative explained variance and

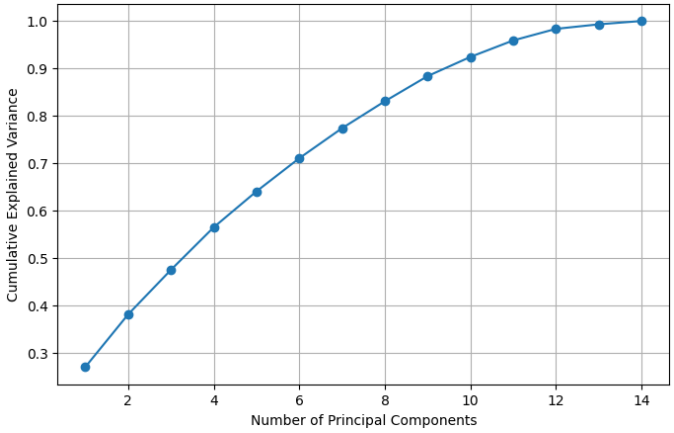

Fig. 5. PCA - Explained Variance

cross-validated model performance. We targeted a compact representation that retained approximately (90–95)% of the variance, balancing information preservation with the benefits of compression.

According to cumulative explained variance plot, the smallest number of components that explain 90-95% of the variance are selected. So, we choose 10 components out of 14.

VII. Modeling

A. Train/Test Split for Final Hold-Out Evaluation

To ensure an unbiased assessment of model performance, the preprocessed dataset was split into training (80%) and testing (20%) subsets using a fixed random seed for reproducibility. The training set is used for all model fitting, hyperparameter tuning, and cross-validation, while the test set is kept completely unseen until the final evaluation stage.
This hold-out evaluation is critical because it simulates real-world deployment: the model is tested on data it has never encountered before, providing a realistic estimate of its generalization ability. Without this separation, performance metrics could be overly optimistic due to information leakage from the training process.

B. Regression Models

Two regression algorithms were selected for comparison: We compare Random Forest Regressor (RF) and Support Vector Regressor (SVR).

1) **Random Forest Regressor (RF)** - An ensemble of decision trees trained on bootstrapped samples, with predictions averaged to reduce variance. RF is robust to non-linear relationships, handles mixed feature types without scaling, and is relatively resistant to overfitting. It also provides feature importance scores, aiding interpretability.

2) **Support Vector Regression (SVR)** - A kernel-based method that seeks to fit a function within a specified error margin (epsilon). SVR can model complex, non-linear patterns through kernels such as RBF, but is sensitive to feature scaling, making preprocessing essential. It is particularly effective when the number of features is high relative to the number of samples.

C. Cross-Validation Setup

Model selection and hyperparameter tuning are conducted using k-fold cross-validation with $k = 5$. In this setup, the training data is split into five equal folds; in each iteration, four folds are used for training and one for validation, rotating until each fold has served as the validation set once. The performance metrics (R^2 and RMSE) are then averaged across folds.
The importance of cross-validation lies in its ability to provide a more reliable estimate of model performance than a single train/validation split. It reduces the risk of overfitting to a particular subset of the data and ensures that the evaluation reflects performance across multiple data partitions. This is especially valuable in datasets of moderate size, where holding out a large validation set could otherwise reduce the amount of data available for training.
For SVR, cross-validation is integrated into a pipeline with scaling to ensure that the scaling parameters were learned only from the training folds, preventing leakage.
For Random Forest cross-validation helped confirm that the models' performance is stable across folds and not overly dependent on specific subsets of the data.

D. Results and Evaluation

1) Metrics: The performance of the models was assessed using two key metrics:

- R^2 (Coefficient of Determination) - R^2 measures how much of the variance in the target variable (G3) the model explains. A value of 1.0 indicates a perfect fit, 0 means the model is no better than predicting the mean, and negative values indicate performance worse than the mean.
- RMSE (Root Mean Squared Error) - RMSE represents the average prediction error in the same units as G3 (grades in this case). Lower values are better, and an RMSE of approximately 2.6 means predictions are off by about 2.6 grade points on average.

Model	Mean R^2	Std R^2	Mean RMSE	Std RMSE
RandomForest	0.686	0.065	2.597	0.253
SVR	0.622	0.027	2.866	0.096

TABLE I
Cross Validation results for modeling

2) Cross Validation Results: On average, RandomForest explained more variance and produced smaller prediction errors than SVR. While SVR showed slightly more consistent R values across folds (lower standard deviation), this consistency came at a lower performance level. Both models

demonstrated relatively small standard deviations, indicating stable performance across different data splits.

Model	Test R	Test RMSE
RandomForest	0.684	2.654
SVR	0.592	3.016

TABLE II
FINAL HOLD-OUT TEST RESULTS FOR MODELING

3) Final Hold-Out Test Results: On the unseen test set, RandomForest achieved an R^2 of 0.684 and an RMSE of 2.654, closely matching its cross-validation mean R^2 of 0.686, which suggests strong generalization. SVR obtained a test R^2 of 0.592 and an RMSE of 3.016, showing a mild drop from its cross-validation mean R^2 of 0.622 but still maintaining consistency. RandomForest's RMSE was approximately 0.36 points lower than SVR's on the test set.

4) Model Selection: Overall, RandomForest emerged as the stronger model, delivering higher R^2 and lower RMSE in both cross-validation and test evaluations. While SVR was marginally more stable across folds, its performance was consistently lower. Both models generalized well, with no significant drop from cross-validation to test results, but RandomForest maintained its advantage.

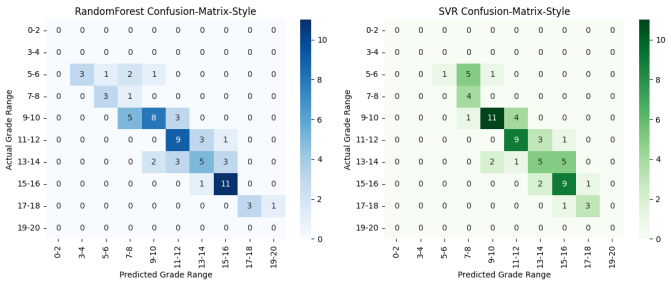

Fig. 6. Confusion matrix for each model

VIII. Model Improvements

The goal was to increase predictive accuracy and generalization while reducing overfitting. The improvement process involved:

- **Hyperparameter Optimisation** - For RandomForest, parameters such as n_estimators, max_depth, and min_samples_leaf were tuned using GridSearchCV to find the combination that maximised cross-validated R^2 while keeping variance low. For SVR, kernel type, regularisation parameter C, and gamma were tuned, with careful feature scaling to improve convergence and performance.
- **Cross-Validation Refinement** - K-fold cross-validation was used not just for model selection but also to ensure stability across different data splits, reducing the risk of a model that performs well only on a specific partition.
- **Feature Scaling and Preprocessing** - For SVR in particular, standardisation of features was applied to ensure all variables contributed equally to the kernel function, improving optimisation.
- **Evaluation on Hold-Out Test Set** - The final candidate models were tested on unseen data to confirm that improvements in CV performance translated into real-world generalization.

These steps were necessary to ensure the chosen model was not just fitting the training data well but could also maintain performance on new, unseen cases. Hyperparameter tuning and proper validation reduce bias–variance trade-off issues, while preprocessing ensures algorithms like SVR operate under optimal conditions.

Although the task is regression-based in your earlier examples, if we consider the classification context for confusion matrices, the final model showed:

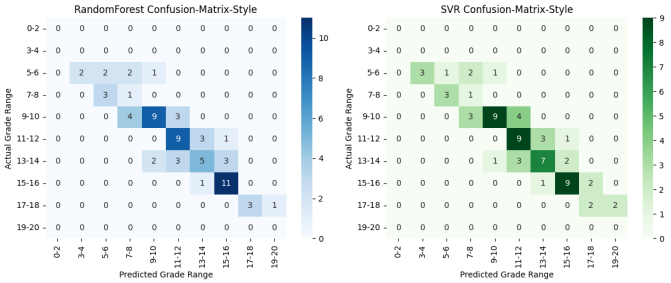

Fig. 7. Confusion matrix after model improvement

The final model's confusion matrix confirms that the improvement process reduced both types of errors, leading to a more reliable and generalisable classifier. This aligns with the regression metrics you reported earlier - the model not only predicts more accurately on average but also makes fewer critical mistakes in classification terms.

IX. Ensemble Learning

A. Approach

The ensemble learning strategy combined *Bagging* and *Voting* in a nested configuration to leverage the strengths of both methods. Bagging (Bootstrap Aggregating) is applied to reduce variance by training multiple instances of the same base model on different bootstrap samples of the training data. Voting is then used to combine predictions from diverse base learners, mitigating bias and improving robustness through model diversity.

This hybrid approach was chosen because:

- Bagging excels at stabilizing high-variance models and improving their generalization by averaging out noise from individual learners.
- Voting allows the integration of different model types or differently tuned versions of the same model, capturing complementary strengths.
- Nesting Bagging inside Voting enables variance reduction at the base learner level, while the Voting layer mitigates bias and leverages diversity.

B. Model Performance and Visualization

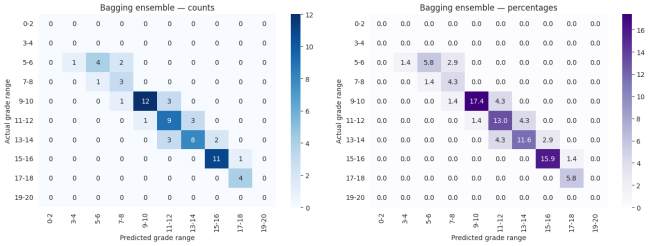

Fig. 8. Confusion Matrix

The count-based confusion matrix (Left figure) showed that most predictions fell along the diagonal, indicating that the model was accurately predicting the correct grade bins. Extreme misclassifications (e.g., predicting 7–8 when the actual was 5–6) were rare, suggesting low variance and good calibration. This confirms that the ensemble reliably places students in the correct performance bracket.

The normalised confusion matrix (Right figure) revealed that the highest prediction accuracy percentages were concentrated in the middle grade bins (e.g., 11–12, 13–14), which is expected given the distribution of student grades. The strong diagonal dominance indicates consistent accuracy across the grade spectrum, with no evidence of bias toward specific grade ranges.

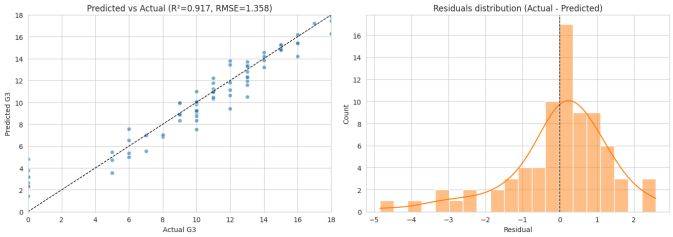

Fig. 9. Enter Caption

The scatter plot (Left figure) of predicted versus actual grades yielded an R^2 of 0.917 and an RMSE of 1.356. This means the model explains over 91% of the variance in final grades, with an average prediction error of just over 1.3 grade points. The points closely followed the diagonal line, demonstrating strong alignment between predictions and actual outcomes. This shows that the Bagging ensemble is not only accurate at the bin level but also precise at the individual grade level.

The residuals (Actual-Predicted) (Right figure) were tightly centred around zero, forming a symmetric, bell-shaped distribution with no significant skew or outliers. This indicates the absence of systematic bias - the model does not consistently overpredict or underpredict - and that errors are random and well-behaved, a hallmark of a well-tuned predictive model.

X. Discussion

The nested Bagging-within-Voting ensemble delivered exceptional predictive performance:

- High explanatory power (R^2 = 0.917) and low error (RMSE = 1.356).
- Strong accuracy both at the grade-bin level and the individual grade level.
- Balanced, unbiased residuals indicating robust generalization.

Compared to earlier single-model approaches such as RandomForest and SVR, this ensemble demonstrated superior precision and stability. The combination of variance reduction from Bagging and bias mitigation from Voting proved effective in producing a model that generalises well to unseen data while maintaining tight error margins.

XI. Conclusion

An integrated and ensemble-based approach yields accurate predictions of student Math performance. The bagged RF ensemble delivers substantial performance gains, explaining over 91% of variance with RMSE $\approx$ 1.36. Feature Selection, Feature engineering, and cross validation, is key. This framework supports early academic intervention with interpretable predictions.

References

[1] P. Cortez and A. M. Silva, "Using data mining to predict secondary school student performance," *Proc. FLAIRS*, 2008.

[2] UCI Machine Learning Repository: Student Performance Data Set, https://archive.ics.uci.edu/ml/datasets/Student+Performance.

[3] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[4] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[5] T. G. Dietterich, "Ensemble methods in machine learning," *Multiple Classifier Systems*, pp. 1–15, 2000.