Identifying key features for academic success by regression

Authors redacted

Contents

Introduction and Motivation

Identifying and cultivating attributes linked to academic success remains a persistent challenge in education [1]. In Finland this is particularly relevant given falling PISA scores [2] and cuts to education funding [3], developments that place greater emphasis on individual student traits and thus increase the incentive to distinguish and refine them. Machine learning (ML) offers a data-driven solution to identifying (and potentially cultivating) such traits [4], although it may also diminish students' critical thinking through over-reliance on artificial intelligence [5]. In this project a dataset of four academic features (200 data points) is used to identify which are most important for student success, measured by the label: exam score. The report proceeds by introducing and exploring the dataset, specifying the supervised ML methods used, presenting the results, and concluding.

Problem Formulation and Data Exploration

Dataset Presentation and Feature Selection

The aim of this project is to predict student exam scores from study-related features and to rank the most influential features; this is framed as a supervised learning problem. We use the Kaggle dataset "Analyzing Student Academic Trends" by Saad Ali Yaseen [6]. It contains 200 students (each with a unique student ID) and four academic features: hours studied, hours slept, attendance percentage, and previous exam score. The target label is the exam score. All features and the label are measured in floating-point numbers. The dataset size matches the recommended sample size (50 $\times$ features) [7]. Figure 1a shows 200-point scatter plots of each feature (x-axis) against exam score (y-axis); a linear pattern is especially evident for hours studied and previous exam score. Figure 1b, the Pearson correlation heatmap [8], shows strong linear correlations between hours studied and exam score, and between previous exam score and exam score. Figure 2 presents histograms of the features and the label: exam scores approximate a normal distribution, while the features do not follow any specific distribution.

Preprocessing of Data

The StandardScaler function [9] was used for preprocessing. This standardises features by removing the mean and scaling to unit variance ($\mu = 0$, $\sigma = 1$). Standardisation ensures all features lie on a comparable scale, which is important when the model or feature-selection method—such as Recursive Feature Elimination (RFE)—is sensitive to differences in feature magnitudes. Applying StandardScaler neutralises features measured on larger numeric ranges (e.g. attendance percentage vs. hours studied), preventing them from dominating the model solely due to scale and allowing importance to reflect each feature's relationship with the target. Consequently, the derived feature rankings are more interpretable, balanced and statistically meaningful.

ML Methods

Linear Regression

Based on the scatter plots and correlation matrix in Fig. 1, which indicate linear relationships between features and the label, multiple linear regression (MLR) is employed with the ordinary least squares loss used in sklearn.linear_model.LinearRegression [10]. This loss was chosen for its ease of use and reliability in regression tasks. Student IDs are dropped as they are not meaningful predictors; the remaining four features are used in the full regression. The data is normalised and split into training and test sets using train_test_split from sklearn.model_selection [11] with a 0.7/0.3 training/testing split (note that the same split is applied to the decision tree). This split was chosen to balance the training and testing sets, so that the testing error would be more statistically significant. A MLR is also fitted on three features—e.g. by dropping previous exam score—to test whether a feature is insignificant. The full model is specified as:

$$y_i = w_0 + w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + w_4 x_{i4} + \epsilon_i \tag{1}$$

where w_0 is the intercept, w_i are feature coefficients, x_{ij} are input features, and ϵ_i is the error term. Model performance is evaluated using mean squared error (MSE) on both training and test sets, and R^2 for the full dataset and feature importance is assessed from the magnitudes of the regression coefficients.

Tree-Based Modeling

A Multilayer perceptron was initially considered but not used due to the dataset's small size, which increases overfitting risk and reduces weight reliability. Instead, tree-based methods were adopted because they capture how feature effects vary across different student profiles: for example, distinguishing how attendance may benefit students who study less compared to those who study more. A decision tree[12] was constructed with MSE as the loss function (chosen for its reliability), and a maximum depth of 3 (Fig. 3). The first split occurred at "hours studied 6.75," confirming study time as the dominant predictor of student performance. Subsequent splits incorporated previous scores and attendance: students with previous scores 70.5 and attendance 55.75 had the lowest predicted average score of 22.3, with a mean squared error of 10.55, while higher attendance improved performance within this subgroup. For students with previous scores > 70.5 and even moderate study time (> 5.3 hours) increased predicted scores (36.5 compared to 31.3) and among those studying > 6.75 hours, additional splits at 9.05 hours and previous score thresholds (73 and 76) revealed that higher prior performance consistently amplifies the benefits of study effort. Analysis of the decision tree and random forest models highlights nonlinear interactions that linear models fail to capture. Key interactions generally suggest:

1. Hours studied $\times$ Previous exam scores: The strongest interaction. High prior scorers gain the most from extra study hours, while low scorers improve with additional study but experience diminishing returns.
2. Previous scores $\times$ Attendance: Attendance primarily affects students with lower previous scores. Those combining weak prior performance and low attendance show the lowest predicted outcomes, whereas high scorers remain relatively unaffected by attendance.
3. Low study hours - Compound disadvantages: Students who study little and previously score low with low attendance create a lowest-performing subgroup with predicted average of 22.3. This cluster suggests a compounding effect where weak engagement across multiple dimensions sharply lowers outcomes.

The decision tree suggests that study hours are the primary driver of outcomes, but their impact is amplified by prior performance and attendance. Tree-based models reveal that feature combinations, rather than single variables, determine academic results. Random forest feature importance [13] was conducted with ranked features as: hours studied (0.660), previous scores (0.172), attendance (0.095), and sleep hours (0.073), alongside the Recursive Feature Elimination (RFE) [14] (after scaling) producing the same ranking, which reinforces that study hours dominate prediction, previous scores and attendance act as secondary drivers, and sleep contributes least.

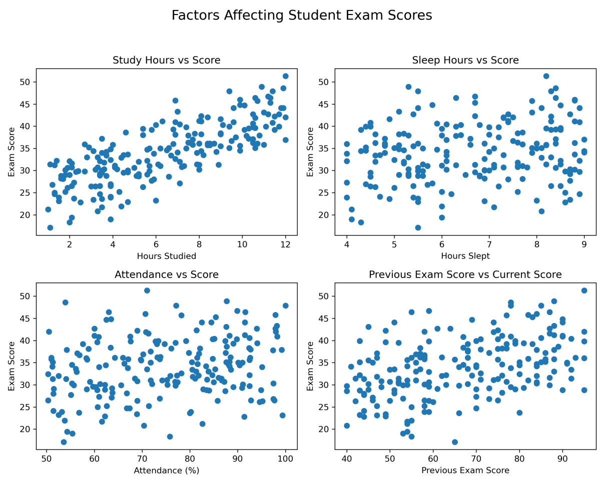
Results

The results of the MLR are shown in Table 1. Observe that hours studied is the most significant feature (MSF) and sleep hours is the least significant feature (LSF). A testing error of 6.86 indicates that the model is a robust predictor of student exam scores and an R^2 score of 0.84 indicates a good fit to the data. It suggests that on average the model is off by 2.62 points when predicting an exam score. There is a low risk of overfitting or bias since the training and testing errors are quite similar. In Table 2 you will find another MLR, but with sleep hours - the LSF - dropped. Training and testing error and R^2 indicate that the fit is still good and does not significantly differ from the full linear regression model; implying that sleep hours matter little when trying to predict student exam scores. The linear regression, decision tree, recursive feature elimination analyses consistently suggested hours studied as the dominant factor influencing future student exam performance. It exhibited the highest correlation using Pearson correlation coefficient (0.78) with exam score and the largest feature importance (0.66 based on random forest model). Previous exam scores emerged as the second most influential predictor as shown in Tables 3 and 4. The feature showed a strong linear correlation with the target variable and was the 2nd-highest in the random forest importance ranking (0.17) and RFE output. This has confirms that students' previous academic performance has a stable predictive relationship with future outcomes. Attendance percentage was moderately influential. It appeared in the middle splits of the decision tree and ranked third in both random forest and RFE importance. Attendance contributed positively to exam outcomes, particularly among students with lower prior performance (See Fig. 3). In the end, the full MLR is chosen as the final ML method. The full MLR shows low training and testing errors (7.51 and 6.86 respectively) and a strong R^2 score of 0.84, indicating a good fit and robust predictive capability. Whereas, the testing MSE of Decision tree and Random Forest models are 14.91 and 9.67 respectively with Decision tree test SSE of 894.42 5.

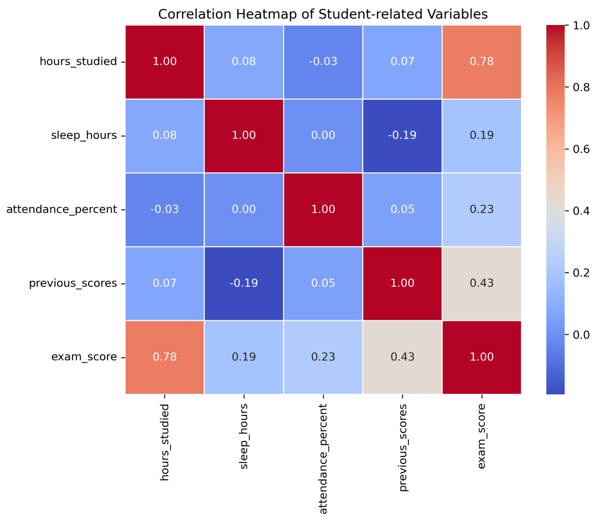
Conclusion

This project successfully identified key predictors of student exam scores using supervised learning models, including MLR and Decision Tree, on a normalized dataset. Feature importance, assessed via the mentioned models and RFE, revealed and confirmed that the most significant factors, in descending order, are hours studied, previous exam score, attendance percentage, and sleep hours. This hierarchy suggests that student effort combined with pre-existing academic interest are the primary drivers of success, reinforcing prior findings on the importance of academic motivation [15]. An analysis of feature interactions uncovered moderating effects. Although sleep hours was the LSF individually, it was found to moderate the impact of both study hours and previous exam scores, suggesting its influence on academic performance stems from its role in study efficiency, which aligns with past research [16]. Furthermore, attendance percentage proved to be a critical factor, particularly for lower-performing students, as it mitigated the negative effects of fewer study hours and poor prior results. High attendance was also shown to enhance the effectiveness of time spent studying. In summary, fostering intrinsic motivation in students is the most vital element for academic achievement. Optimal student performance arises from a combination of consistent study habits, high prior achievement, and strong engagement, with adequate rest helping to stabilize outcomes. This report is conclusive regarding the available dataset, but it is still insufficient due to the small sample size (50) and inclusion of only 4 features. Future studies should include a larger dataset and more features to further pinpoint where a student's success truly lies.

Appendix A: Figures and Tables



(a) Each of the 4 features included in the regression plotted individually against the label. All features appear to be correlated with the label, with hours studied having the strongest correlation with exam score.



	hours_studied	sleep_hours	attendance_percent	previous_scores	exam_score
hours_studied	1.00	0.08	-0.03	0.07	0.78
sleep_hours	0.08	1.00	0.00	-0.19	0.19
attendance_percent	-0.03	0.00	1.00	0.05	0.23
previous_scores	0.07	-0.19	0.05	1.00	0.43
exam_score	0.78	0.19	0.23	0.43	1.00

(b) Correlation matrix 4 features and the label (exam score). Hours studied is the most strongly correlated predictor (r = 0.78). Inter-feature correlations are moderate; the largest (negative) correlation is between previous exam score and sleep hours (r = -0.19). All features show positive correlations with exam score to varying degrees: hours studied (0.78), previous score (0.43), attendance percentage (0.23) and sleep hours (0.19).

Figure 1: Scatter plots of the features against exam score and a correlation map between features and exam score.

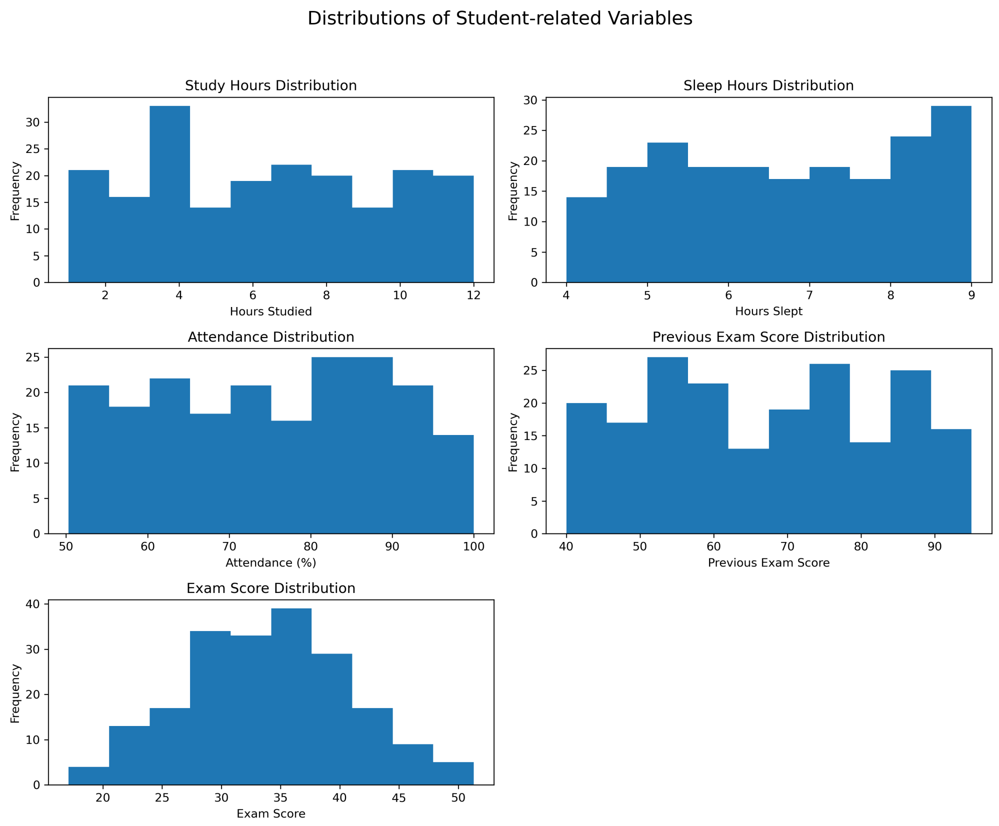


Figure 2: Histograms of all numerical variables in the dataset. Includes 4 features and the label (Exam Score).

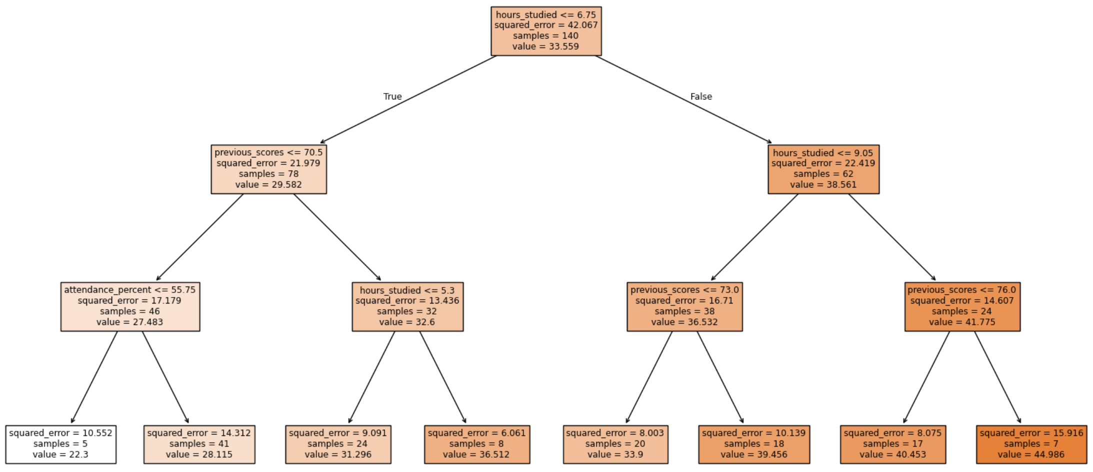


Figure 3: Regression Decision Tree [17] for predicting Student exam scores based on Study hours, Previous scores, and Attendance percent, with maximum depth of 3, and predicted exam score as value.

Table 1: Multiple linear regression summary

(a) Parameters

Parameter	**Value**
w_0 - intercept	34.04
w_1 - hours studied	5.14
w_2 - sleep hours	1.42
w_3 - attendance percent	1.73
w_4 - previous exam scores	2.77

(b) Errors

Metric	**Value**
Training error (MSE)	7.51
Training RMSE	2.74
Testing error (MSE)	6.86
Testing RMSE	2.62
R^2 score	0.84

Table 2: Multiple linear regression with sleep hours dropped summary

(a) Parameters

Parameter	**Value**
w_0 - intercept	34.05
w_1 - hours studied	5.30
w_3 - attendance percent	1.84
w_4 - previous exam scores	2.47

(b) Errors

Metric	**Value**
Training error	9.42
Testing error	8.96
R^2 score	0.80

Table 3: Random Forest Feature Importance

Feature	**Importance**
Hours studied	0.6603
Sleep hours	0.0726
Attendance percent	0.0953
Previous exam scores	0.1718

Table 4: RFE Feature Ranking (scaled)

Feature	**Rank**
Hours studied	1
Sleep hours	4
Attendance percent	3
Previous exam scores	2

Model	**Test MSE**	**Test SSE**
Decision Tree	14.9053	894.3181
Random Forest	9.6738	–

Table 5: MSE and SSE (test) for Decision Tree and Random Forest. MSE reflects average squared error, while SSE shows the total error across all samples.

Appendix B: AI usage

AI was used for cosmetic purposes: creating subplots, labeling axes, and providing improvements in text. AI was also used to create a LaTeX file skeleton, debugging problems with LaTeX and for correctly inserting plots into LaTeX, and in tasks such as condensing written text.

Appendix C: Python code

```
import numpy as np                    # import numpy package under shorthand "np"
import pandas as pd                   # import pandas package under shorthand "pd"
import matplotlib.pyplot as plt
from nose.tools import assert_equal
from numpy.testing import assert_array_equal
from sklearn.preprocessing import StandardScaler
# Regression import

%config Completer.use_jedi = False  # enable code auto-completion
from sklearn.linear_model import LinearRegression    # classes providing Linear
↪  Regression with ordinary squared error loss and Huber loss, respectively
from sklearn.metrics import mean_squared_error    # function to calculate mean squared
↪  error

from sklearn.model_selection import train_test_split
```

```
df = pd.read_csv('student_exam_scores.csv')

# print the first 5 weather recordings in the DataFrame `df`

df = df.drop(['student_id'], axis = 1)

# print the first 5 weather recordings in the DataFrame `df`
studies = df['hours_studied'].to_numpy()
sleep = df['sleep_hours'].to_numpy()
attendance = df['attendance_percent'].to_numpy()
previous = df['previous_scores'].to_numpy()
score = df['exam_score'].to_numpy()
df.head(5)
```

```python
fig, axs = plt.subplots(2, 2, figsize=(10, 8))

# Top-left
axs[0, 0].scatter(studies, score)
axs[0, 0].set_title('Study Hours vs Score')
axs[0, 0].set_xlabel('Hours Studied')
axs[0, 0].set_ylabel('Exam Score')

# Top-right
axs[0, 1].scatter(sleep, score)
axs[0, 1].set_title('Sleep Hours vs Score')
axs[0, 1].set_xlabel('Hours Slept')
axs[0, 1].set_ylabel('Exam Score')

# Bottom-left
axs[1, 0].scatter(attendance, score)
axs[1, 0].set_title('Attendance vs Score')
axs[1, 0].set_xlabel('Attendance (%)')
axs[1, 0].set_ylabel('Exam Score')

# Bottom-right
axs[1, 1].scatter(previous, score)
axs[1, 1].set_title('Previous Exam Score vs Current Score')
axs[1, 1].set_xlabel('Previous Exam Score')
axs[1, 1].set_ylabel('Exam Score')

# General title for the whole figure
fig.suptitle('Factors Affecting Student Scores', fontsize=16)

# Improve spacing so suptitle doesn't overlap subplots
fig.tight_layout(rect=[0, 0, 1, 0.95])

plt.show()
```

```python
fig, axs = plt.subplots(2, 2, figsize=(10, 8))

# Top-left
axs[0, 0].scatter(studies, score)
axs[0, 0].set_title('Study Hours vs Score')
axs[0, 0].set_xlabel('Hours Studied')
axs[0, 0].set_ylabel('Exam Score')

# Top-right
axs[0, 1].scatter(sleep, score)
axs[0, 1].set_title('Sleep Hours vs Score')
axs[0, 1].set_xlabel('Hours Slept')
axs[0, 1].set_ylabel('Exam Score')

# Bottom-left
axs[1, 0].scatter(attendance, score)
axs[1, 0].set_title('Attendance vs Score')
axs[1, 0].set_xlabel('Attendance (%)')
axs[1, 0].set_ylabel('Exam Score')

# Bottom-right
axs[1, 1].scatter(previous, score)
axs[1, 1].set_title('Previous Exam Score vs Current Score')
axs[1, 1].set_xlabel('Previous Exam Score')
axs[1, 1].set_ylabel('Exam Score')

# General title for the whole figure
fig.suptitle('Factors Affecting Student Scores', fontsize=16)

# Improve spacing so suptitle doesn't overlap subplots
fig.tight_layout(rect=[0, 0, 1, 0.95])

plt.show()
```

```python
import seaborn as sns  # add this at the top of your imports

# Compute correlation matrix
matrix = df.corr()

# Plot heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)
plt.title("Correlation Heatmap")
plt.show()
```

```python
fig, axs = plt.subplots(3, 2, figsize=(12, 10))

# Top-left histogram
axs[0, 0].hist(studies)
axs[0, 0].set_title('Study Hours Distribution')
axs[0, 0].set_xlabel('Hours Studied')
axs[0, 0].set_ylabel('Frequency')

# Top-right histogram
axs[0, 1].hist(sleep)
axs[0, 1].set_title('Sleep Hours Distribution')
axs[0, 1].set_xlabel('Hours Slept')
axs[0, 1].set_ylabel('Frequency')

# Middle-left histogram
axs[1, 0].hist(attendance)
axs[1, 0].set_title('Attendance Distribution')
axs[1, 0].set_xlabel('Attendance (%)')
axs[1, 0].set_ylabel('Frequency')

# Middle-right histogram
axs[1, 1].hist(previous)
axs[1, 1].set_title('Previous Exam Score Distribution')
axs[1, 1].set_xlabel('Previous Exam Score')
axs[1, 1].set_ylabel('Frequency')

# Bottom-left: the 5th histogram (added from your separate figure)
axs[2, 0].hist(score)
axs[2, 0].set_title('Exam Score Distribution')
axs[2, 0].set_xlabel('Exam Score')
axs[2, 0].set_ylabel('Frequency')

# Bottom-right is unused (turn it off)
axs[2, 1].axis('off')

# General title for the whole figure
fig.suptitle('Distributions of Student-related Variables', fontsize=16)

# Adjust layout so suptitle doesn't overlap subplots
fig.tight_layout(rect=[0, 0, 1, 0.95])

plt.show()
```

```python
fig, axs = plt.subplots(3, 2, figsize=(12, 10))
features = df.drop(['exam_score'], axis = 1)
X = features
y = df['exam_score']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
↪  random_state=101)
```

```python
fig, axs = plt.subplots(3, 2, figsize=(12, 10))
# Full linear regression
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=101)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_scaled = scaler.transform(X)
X_test_scaled  = scaler.transform(X_test)
regr = LinearRegression()
regr.fit(X_train_scaled, y_train)

y_pred_test = regr.predict(X_test_scaled)
y_pred_train = regr.predict(X_train_scaled)
test_error_full = mean_squared_error(y_test, y_pred_test)
tr_error_full = mean_squared_error(y_train, y_pred_train)
print("wi = ", regr.coef_)
print("w0 = ",regr.intercept_)
print("testing error:", test_error_full)
print("training error:", tr_error_full)
print("score:", regr.score(X_scaled, y))
```

```python
fig, axs = plt.subplots(3, 2, figsize=(12, 10))
# Regression without sleep hours

X_scaled_no_sleep = X_scaled[:, [i for i, c in enumerate(X.columns) if c !=
↪  'sleep_hours']]
X_train_ns, X_test_ns, y_train_ns, y_test_ns = train_test_split(
    X_scaled_no_sleep, y, test_size=0.3, random_state=101
)

regr_no_sleep = LinearRegression()
regr_no_sleep.fit(X_train_ns, y_train_ns)

y_pred_train_ns = regr_no_sleep.predict(X_train_ns)
y_pred_test_ns = regr_no_sleep.predict(X_test_ns)
train_mse_ns = mean_squared_error(y_train_ns, y_pred_train_ns)
test_mse_ns = mean_squared_error(y_test_ns, y_pred_test_ns)

print("Linear Regression without sleep_hours, scaled:")
print("wi =", regr_no_sleep.coef_)
print("w0 =", regr_no_sleep.intercept_)
print("training error:", train_mse_ns)
print("testing error:", test_mse_ns)
print("score:", regr_no_sleep.score(X_scaled_no_sleep, y))
```

```python
# Standard Scaler
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Recursive Feature Ranking
from sklearn.feature_selection import RFE

rfe = RFE(LinearRegression(), n_features_to_select=1)
rfe.fit(X_train, y_train)

rfe_scaled = RFE(LinearRegression(), n_features_to_select=1)
rfe_scaled.fit(X_train_scaled, y_train)

print("RFE Feature ranking:", rfe.ranking_)
print("RFE Feature ranking (scaled):", rfe_scaled.ranking_)

# Decision Tree
from sklearn.tree import DecisionTreeRegressor, plot_tree
import matplotlib.pyplot as plt
dt = DecisionTreeRegressor(max_depth = 3)
dt.fit(X_train, y_train)
y_test_pred  = dt.predict(X_test)
test_mse  = mean_squared_error(y_test, y_test_pred)
test_sse = test_mse * len(y_test)
print("Decision Tree MSE")
print(f"Test MSE: {test_mse:.4f}, Test SSE: {test_sse:.4f}")
plt.figure(figsize=(20,10))
plot_tree(dt, feature_names=['hours_studied', 'sleep_hours', 'attendance_percent',
↪  'previous_scores'], filled=True)
plt.show()

# Random Forest Regressor
rf = RandomForestRegressor(n_estimators = 100, random_state=42) # for reproducibility
rf.fit(X_train, y_train)

print("RF feature importance:", rf.feature_importances_)

y_pred_rf = rf.predict(X_test)
mse_rf = mean_squared_error(y_test, y_pred_rf)
print("Random Forest MSE on test set:", mse_rf)
```

Bibliography

[1] Wenwen Cao et.al, "Unraveling the factors shaping academic success: A structural equation modeling approach for college students." `https://pmc.ncbi.nlm.nih.gov/articles/PMC10875422/`. Accessed: 2025-10.

[2] Yle: Hanna Terävä, "Suomen pisa-menestys romahti: lukutaidossa suurempi muutos kuin koskaan aiemmin." `https://yle.fi/a/74-20063393`. Accessed: 2025-10.

[3] Helsingin sanomat: Veera Paananen, "Koulutuksesta leikataan." `https://www.hs.fi/politiikka/art-2000009662045.html`. Accessed: 2025-10.

[4] Microsoft Education Team, "Ai in education report: Insights to support teaching and learning." `https://www.microsoft.com/en-us/education/blog/2025/08/ai-in-education-report-insights-to-support-teaching-and-learning/`. Accessed: 2025-10.

[5] Hao-Ping Lee et.al, "The impact of generative ai on critical thinking: Self-reported reductions in cognitive effort and confidence effects from a survey of knowledge workers." `https://www.microsoft.com/en-us/research/wp-content/uploads/2025/01/lee_2025_ai_critical_thinking_survey.pdf`. Accessed: 2025-10.

[6] S. A. Yaseen, "Analyzing student academic trends." `https://www.kaggle.com/datasets/saadaliyaseen/analyzing-student-academic-trends`. Accessed: 2025-09.

[7] Google cloud, "Is my data any good? a pre-ml checklist." `https://services.google.com/fh/files/blogs/data-prep-checklist-ml-bd-wp-v2.pdf`. Accessed: 2025-09.

[8] T. pandas development team, "pandas.dataframe.corr," 2023. Computes pairwise correlation of DataFrame columns.

[9] scikit-learn developers, “sklearn.preprocessing.standardscaler — scikit-learn.” `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html`. Accessed: 2025-10.

[10] scikit-learn developers, “sklearn.linear_model.linearregression — scikit-learn.” `https://www.scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html`. Accessed: 2025-09.

[11] scikit-learn developers, “sklearn.model_selection — scikit-learn.” `https://scikit-learn.org/stable/api/sklearn.model_selection.html`. Accessed: 2025-10.

[12] scikit-learn developers, “sklearn.tree.decisiontreeregressor — scikit-learn.” `https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html`. Accessed: 2025-10.

[13] scikit-learn developers, “sklearn.ensemble.randomforestregressor — scikit-learn.” `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html`. Accessed: 2025-10.

[14] scikit-learn developers, “sklearn.feature_selection.rfe — scikit-learn.” `https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html`. Accessed: 2025-10.

[15] R. Steinmayr, A. F. Weidinger, M. Schwinger, and B. Spinath, “The importance of students’ motivation for their academic achievement - replicating and extending previous findings,” *Frontiers in Psychology*, vol. 10, p. 1730, 2019. PMCID: PMC6685139; PMID: 31417459.

[16] S. D. Hershner, “Sleep and academic performance: measuring the impact of sleep,” *Current Opinion in Behavioral Sciences*, vol. 33, pp. 51–56, 2020. ISSN: 2352-1546.

[17] scikit-learn developers, “sklearn.tree.plot_tree — scikit-learn.” `https://scikit-learn.org/stable/modules/generated/sklearn.tree.plot_tree.html`. Accessed: 2025-10.