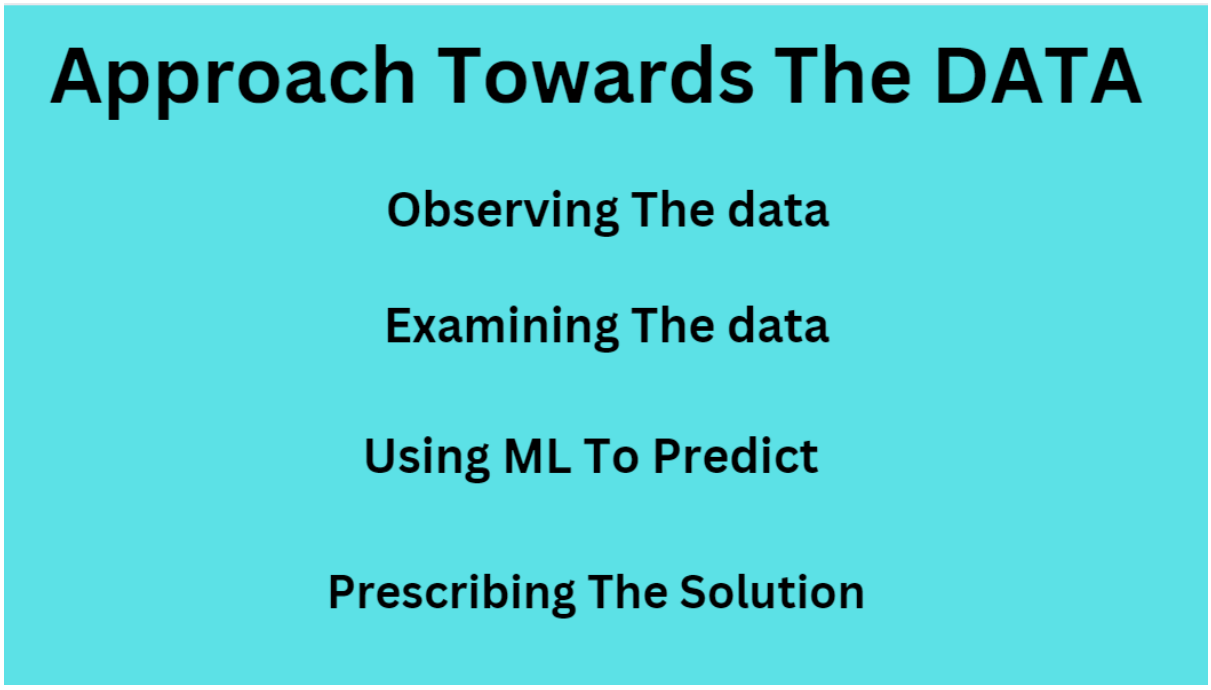
SOLUTION TO THE PROBLEM STATEMENT

Idea:

The goal of this project is to come up with a model(s) that can help us understand the performance of a student based on some attributes, such as the student's family size, alcohol consumption habits, study timings, number of failures, etc. The dataset provided contains 33 variables that could determine the students' performance and the quality of education provided (assuming that the grades received by the students represent the quality of education provided).

It's not necessary that all the variables are relevant for prediction. The idea is to figure out and focus on the important factors that can affect the student's performance.

Approach:



The initial step is to observe and perform Exploratory Data Analysis to better understand the data and the different variables. We use various visualization and statistical techniques such as the F-test and the t-test to find the important and relevant variables in the dataset. This helps us to figure out which variables are having significant impact on the students' performance.

The dataset provided is already cleaned, so there is no requirement for cleaning the data. But we perform label encoding to convert the categorical variables into numeric as they help in easier and faster calculations.

We use various Machine Learning techniques to predict the final grade of the student, which according to our assumption determines the quality of education. We use models such as Linear Regression, Lasso, Random Forest Regressor, Gradient Boosting, AdaBoost Regressor, etc. for our prediction model and try to come up with the best among the models based on the R2 score.

The UI Platform inputs the data from the user and predicts the final grade based on the input variable. This value is presented to the user, possibly with a slight suggested range (ordinal range).

The output will help the user predict the final grade for a student based on the input variable values. The UI platform aims to not only predict the final grade, but also raise awareness of quality of education provided.

With our project “Hogward Students Performance Prediction Model” we not aim in predicting the final grade of the student, but also generate insights that are easily understandable and usable by the common people, which will contribute towards the democratization of education and help us align our thoughts with the principles of Sustainable Development Goal 4: Quality Education.

With our project we aim to contribute, even the tiniest possible from our side, towards the quality of education provided and help develop better educational environment for the upcoming generation and students.

Model Architectures and Training:

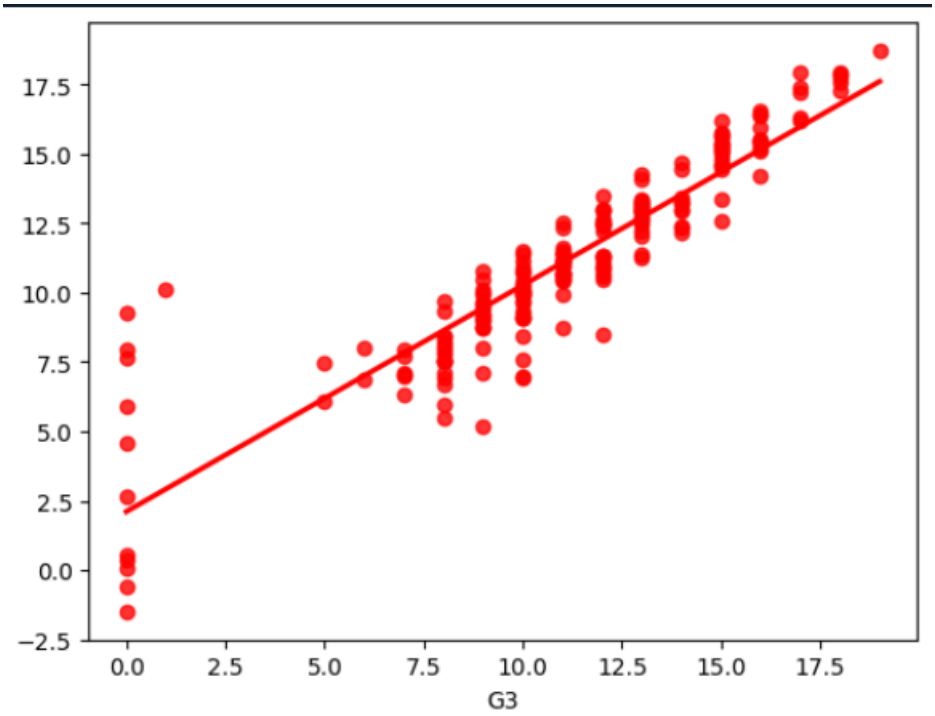
```python
def evaluate_model(true, predicted):
    mae = mean_absolute_error(true, predicted)
    mse = mean_squared_error(true, predicted)
    rmse = np.sqrt(mean_squared_error(true, predicted))
    r2_square = r2_score(true, predicted)
    return mae, mse, rmse, r2_square
```

```python
models = {
    "Linear Regression": LinearRegression(),
    "Lasso": Lasso(),
    "K-Neighbors Regressor": KNeighborsRegressor(),
    "Decision Tree": DecisionTreeRegressor(),
    "Random Forest Regressor": RandomForestRegressor(),
    "Gradient Boosting": GradientBoostingRegressor(),
    "XGBRegressor": XGBRegressor(),
    "CatBoost Regressor": CatBoostRegressor(verbose=False),
    "AdaBoost Regressor": AdaBoostRegressor()
}

model_list = []
r2_list = []

for model_name, model in models.items():
    model.fit(X_train, y_train)  # Train model

    # Make predictions
    y_train_pred = model.predict(X_train)
    y_test_pred = model.predict(X_test)

    # Evaluate Train and Test dataset
    model_train_mae, model_train_mse, model_train_rmse, model_train_r2 = evaluate_model(y_train, y_train_pred)
    model_test_mae, model_test_mse, model_test_rmse, model_test_r2 = evaluate_model(y_test, y_test_pred)

    print(model_name)
    model_list.append(model_name)

    print('Model performance for Training set')
    print("- Root Mean Squared Error: {:.4f}".format(model_train_rmse))
    print("- Mean Squared Error: {:.4f}".format(model_train_mse))
    print("- Mean Absolute Error: {:.4f}".format(model_train_mae))
    print("- R2 Score: {:.4f}".format(model_train_r2))

    print('----------------------------------')
    print('Model performance for Test set')
    print("- Root Mean Squared Error: {:.4f}".format(model_test_rmse))
    print("- Mean Squared Error: {:.4f}".format(model_test_mse))
    print("- Mean Absolute Error: {:.4f}".format(model_test_mae))
    print("- R2 Score: {:.4f}".format(model_test_r2))

    r2_list.append(model_test_r2)
    print('=' * 35)
    print('\n')
```

Model Evaluation:

```python
model_list = []
r2_list = []

for model_name, model in models.items():
    # Create a scorer object to use in grid search
    scorer = make_scorer(r2_score)

    # Perform grid search to find the best hyperparameters
    grid_search = GridSearchCV(
        model,
        param_grid[model_name],
        scoring=scorer,
        cv=5,
        n_jobs=-1
    )

    grid_search.fit(X_train, y_train)  # Make predictions
    y_train_pred = grid_search.predict(X_train)
    y_test_pred = grid_search.predict(X_test)

    # Evaluate Train and Test dataset
    model_train_mae, model_train_mse, model_train_rmse, model_train_r2 = evaluate_model(y_train, y_train_pred)
    model_test_mae, model_test_mse, model_test_rmse, model_test_r2 = evaluate_model(y_test, y_test_pred)

    print(model_name)
    model_list.append(model_name)

    print('Best hyperparameters:', grid_search.best_params_)

    print('Model performance for Training set')
    print("- Root Mean Squared Error: {:.4f}".format(model_train_rmse))
    print("- Mean Squared Error: {:.4f}".format(model_train_mse))
    print("- Mean Absolute Error: {:.4f}".format(model_train_mae))
    print("- R2 Score: {:.4f}".format(model_train_r2))

    print('----------------------------------')
    print('Model performance for Test set')
    print("- Root Mean Squared Error: {:.4f}".format(model_test_rmse))
    print("- Mean Squared Error: {:.4f}".format(model_test_rmse))
    print("- Mean Absolute Error: {:.4f}".format(model_test_mae))
    print("- R2 Score: {:.4f}".format(model_test_r2))

    r2_list.append(model_test_r2)
    print('=' * 35)
    print('\n')
```

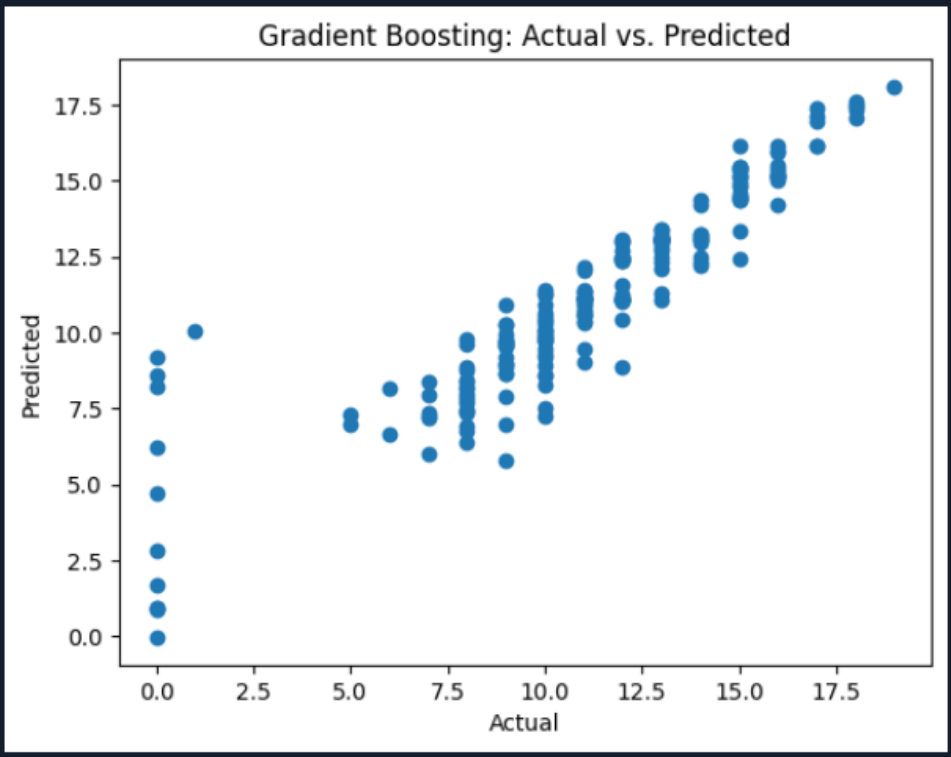
Performance:



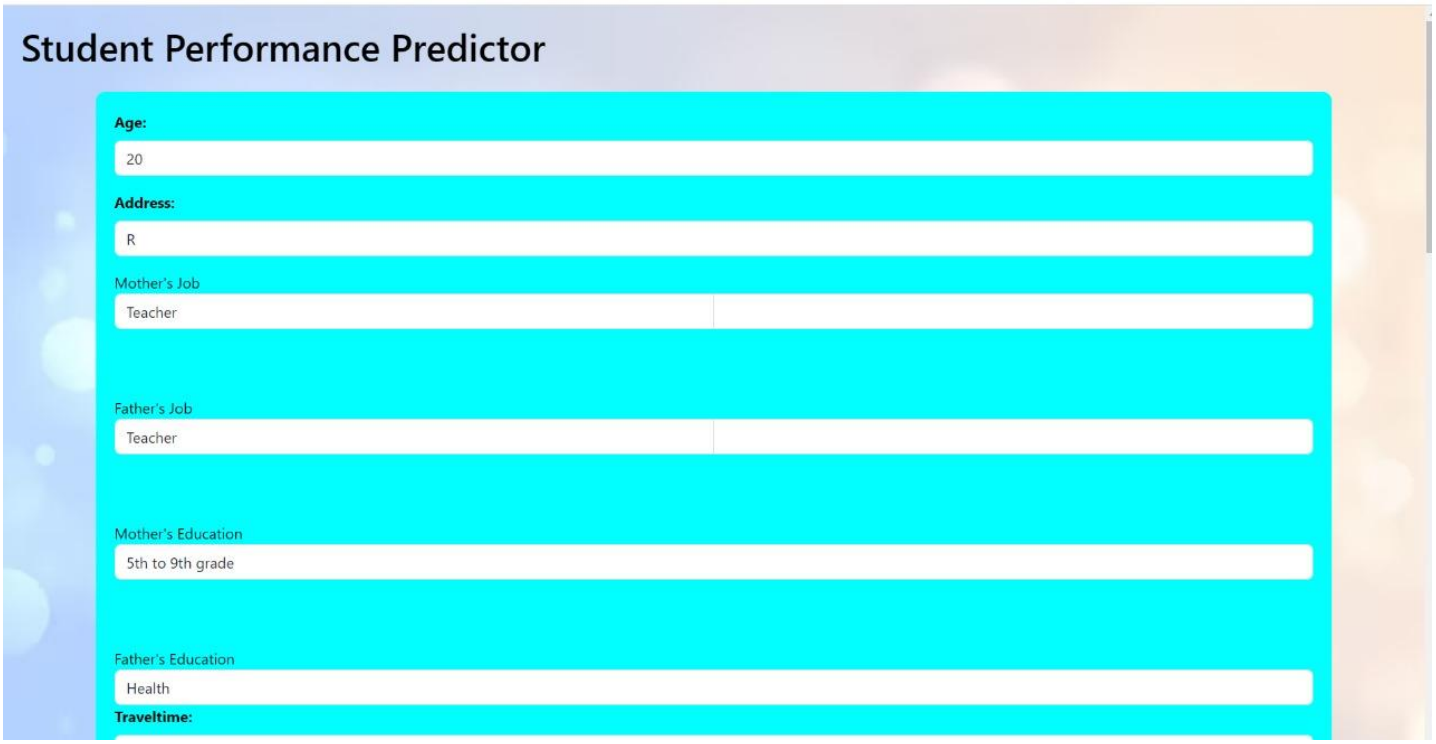
Best Model:

Gradient Boosting
Best hyperparameters: {'learning_rate': 0.05, 'max_depth': 3, 'n_estimators': 64, 'subsample': 0.7}
Model performance for Training set
- Root Mean Squared Error: 1.2169
- Mean Squared Error: 1.4809
- Mean Absolute Error: 0.7737
- R2 Score: 0.8997

Model performance for Test set
- Root Mean Squared Error: 1.6285
- Mean Squared Error: 1.6285
- Mean Absolute Error: 0.9131
- R2 Score: 0.8285



UI/UX:


Student Performance Predictor
Age:
20
Address:
R
Mother's Job
Teacher
Father's Job
Teacher
Mother's Education
5th to 9th grade
Father's Education
Health
Traveltime: