SC1015 Mini-Project

Predicting Student Test Scores

FCED Group 2
Wang Xinping (U2321151E)
Goh Qing Wen (U2322556F)
Glynis Looi Xin Lin (U2321198L)

Table of contents

01

Introduction

Motivation and Data Set

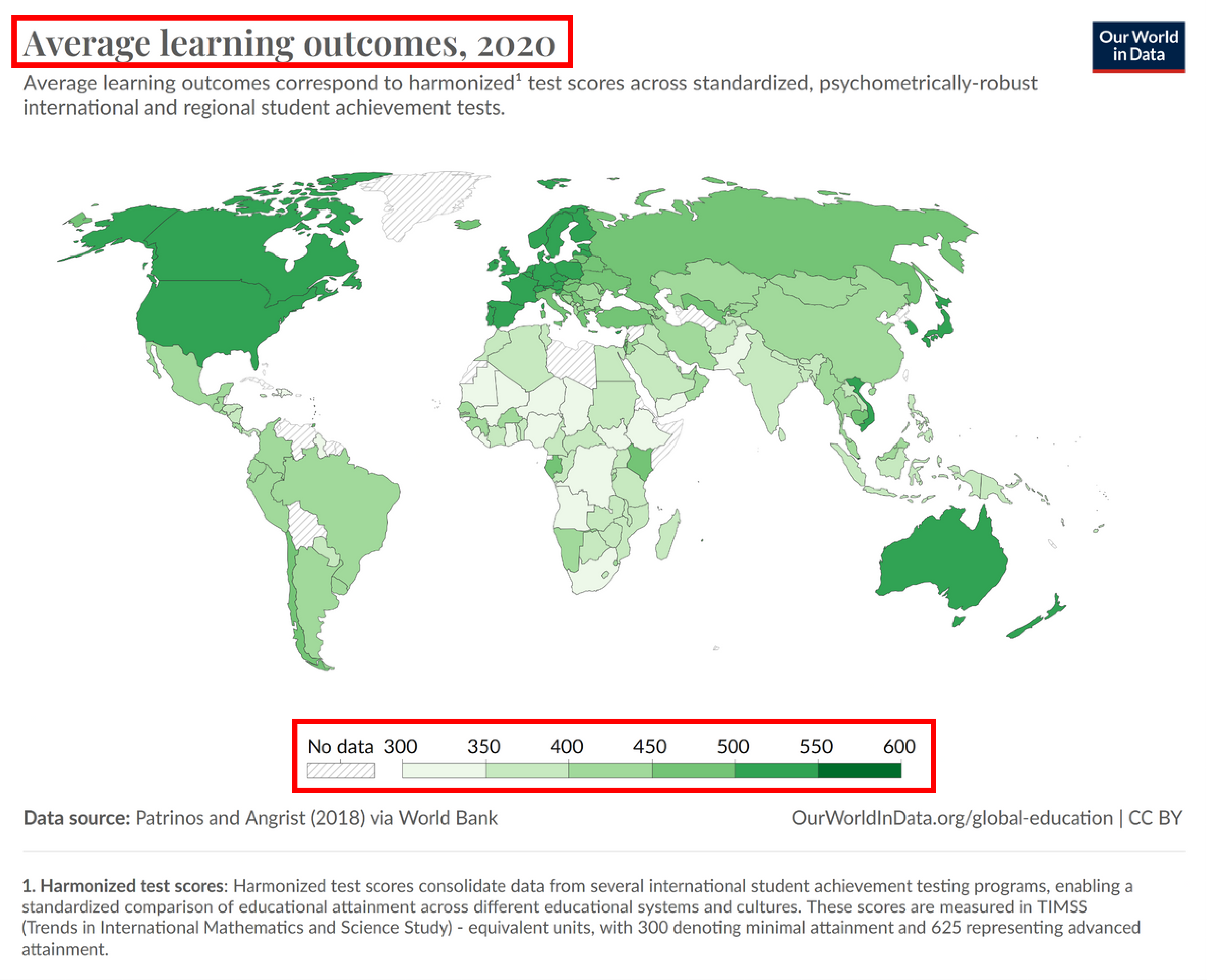
Introduction

- “Education is the most powerful weapon which you can use to change the world.” - Nelson Mandela

- Benefits of Education:
    - Employment Opportunities
    - Career Advancement
    - Development of Critical Thinking Skills
    - Many more……

Introduction

- Education outcomes are most commonly measured by test scores
- Some students tend to do better, some students tend to do worse
- Based on many different factors
- Knowledge of these factors can lead to better allocation of resources and intervention to help students

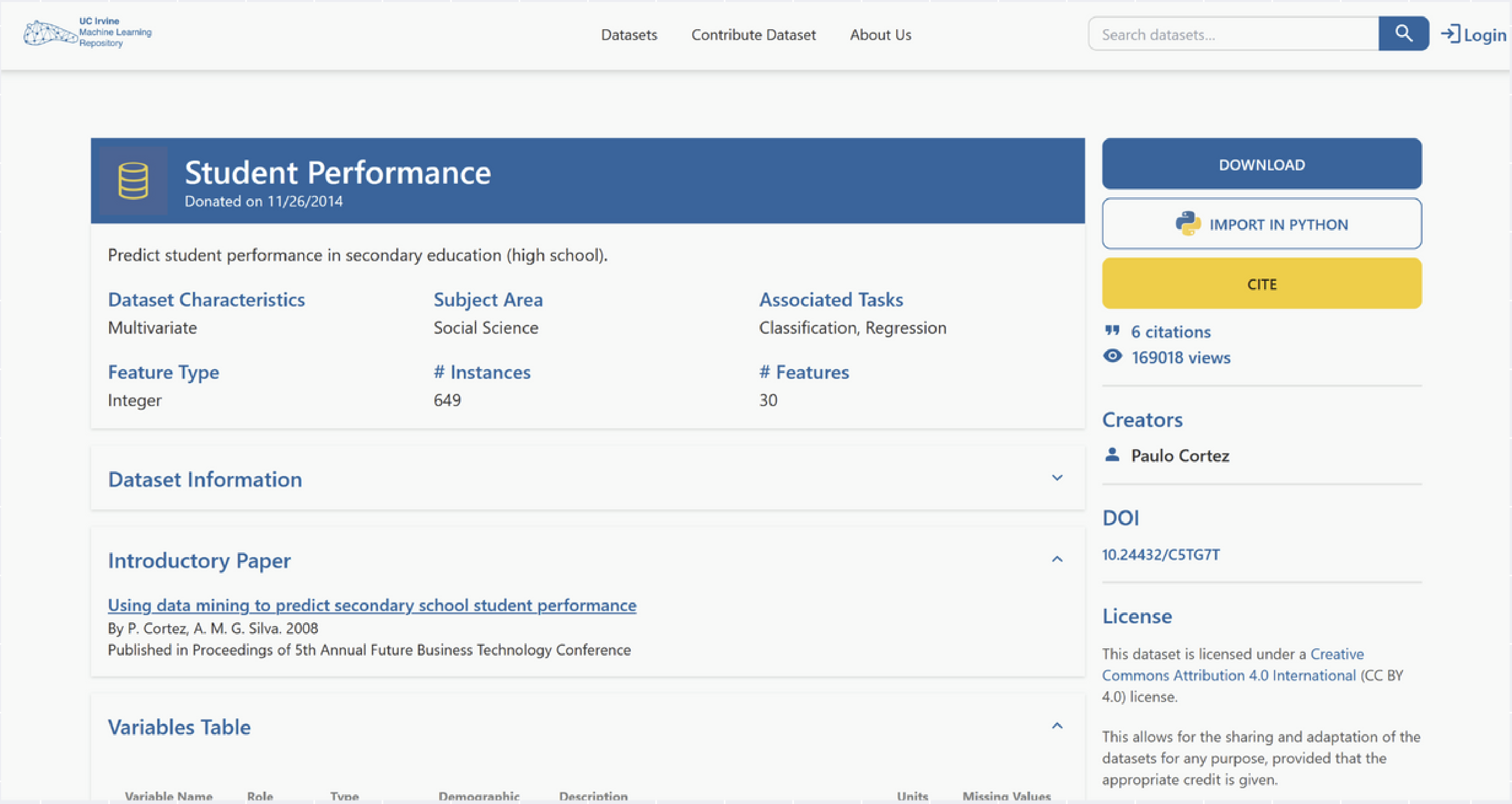


Problem Definition

What variables are the most reliable in determining the future test scores of students?

Data Set

- “Student Performance” by Paulo Cortez
- From the UCI Machine Learning Repository
- Data collected from 2 Portuguese Secondary Schools
- Data Set shows the performance of students in the Portuguese Language, based on demographic, social and school-related variables



02

Exploratory Data Analysis

EDA and Data Preparation

Variable Overview

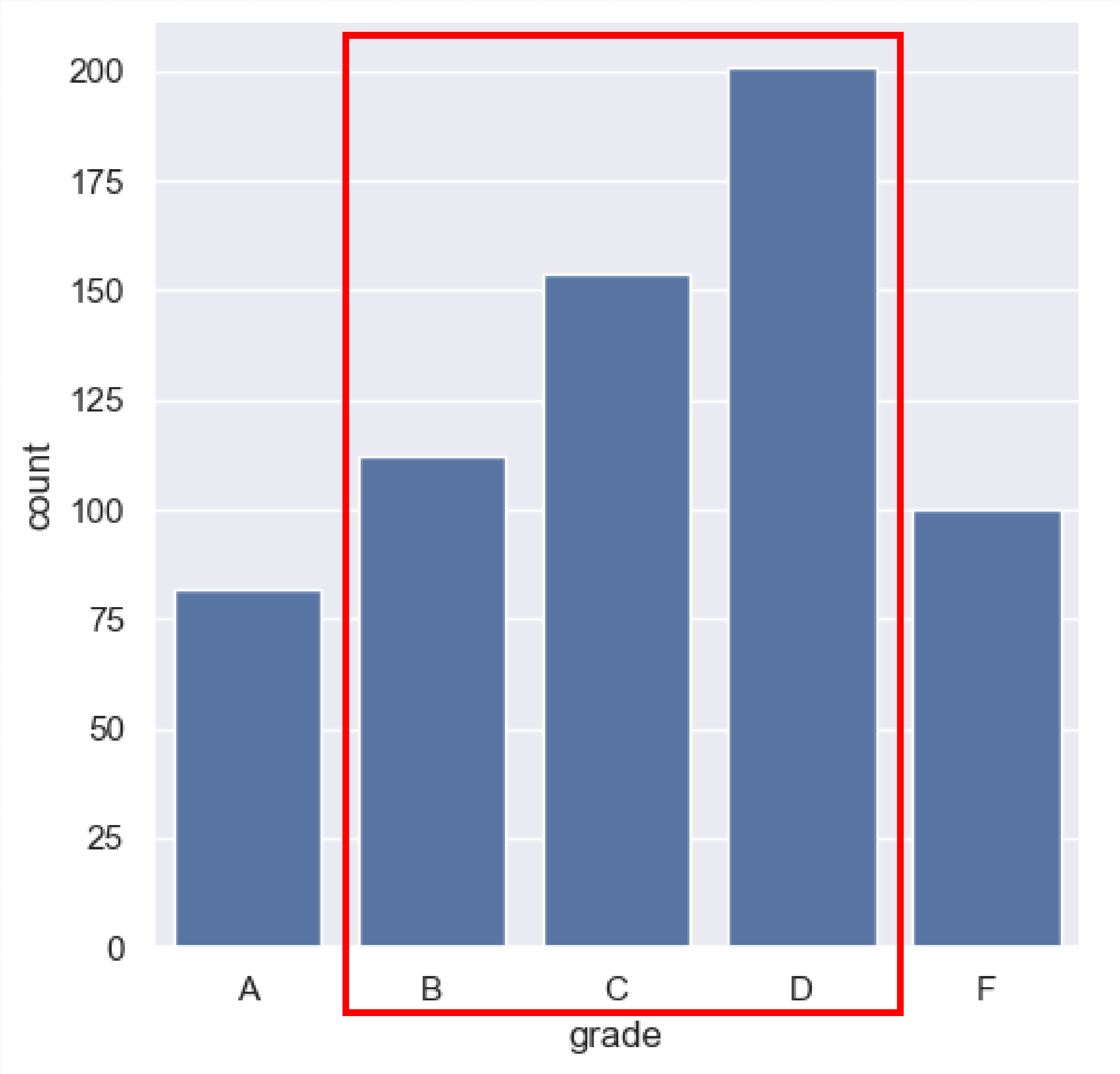
- 33 Different Variables, 650 data rows
- 16 Numeric, 17 Categorical
- G3 represents the final scores of the students, our response variable
- 8 other variables from both data types as predictors

0	school	649 non-null	object
1	sex	649 non-null	object
2	age	649 non-null	int64
3	address	649 non-null	object
4	famsize	649 non-null	object
5	Pstatus	649 non-null	object
6	Medu	649 non-null	int64
7	Fedu	649 non-null	int64
8	Mjob	649 non-null	object
9	Fjob	649 non-null	object
10	reason	649 non-null	object
11	guardian	649 non-null	object
12	traveltime	649 non-null	int64
13	studytime	649 non-null	int64
14	failures	649 non-null	int64
15	schoolsup	649 non-null	object
16	famsup	649 non-null	object
17	paid	649 non-null	object
18	activities	649 non-null	object
19	nursery	649 non-null	object
20	higher	649 non-null	object
21	internet	649 non-null	object
22	romantic	649 non-null	object
23	famrel	649 non-null	int64
24	freetime	649 non-null	int64
25	goout	649 non-null	int64
26	Dalc	649 non-null	int64
27	Walc	649 non-null	int64
28	health	649 non-null	int64
29	absences	649 non-null	int64
30	G1	649 non-null	int64
31	G2	649 non-null	int64
32	G3	649 non-null	int64

Response Variable (G3)

- Student's Final scores
- Converted Numerical to Categorical for better visualisation
- Most scored B to D
- Models trained on this data may be inaccurate for other data sets

Country	I (excellent/very good)	II (good)	III (satisfactory)	IV (sufficient)	V (fail)
Portugal/France	16-20	14-15	12-13	10-11	0-9
Ireland	A	B	C	D	F



Predictor Variables

3 Numeric Variables

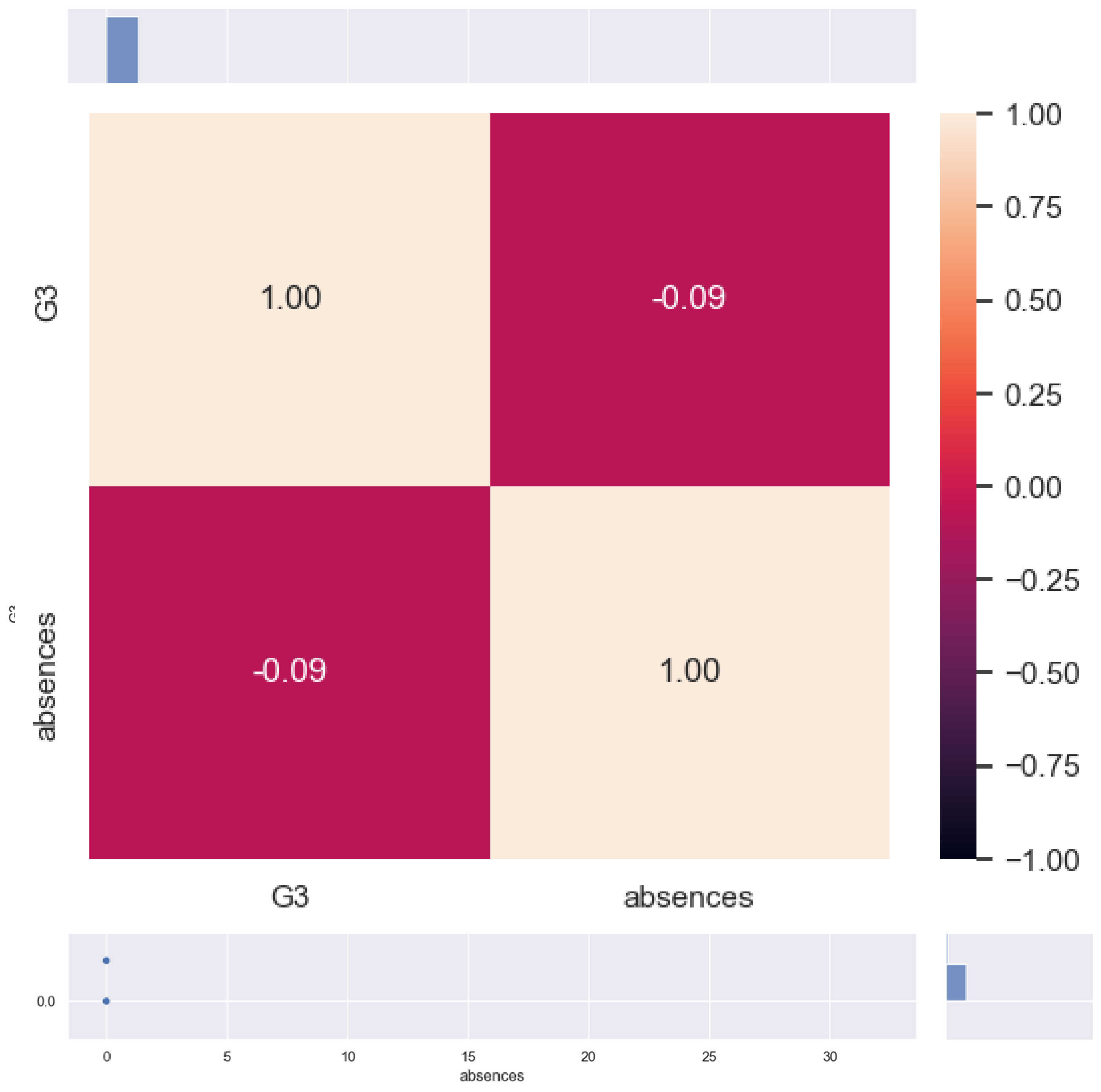
- **Absences**: Number of School Absences
- **Health**: Current Health Status (The higher the better)
- **Study Time**: Weekly Study Time

5 Categorical Variables

- **Address**: Rural/Urban
- **Paid**: Extra Tuition classes (Yes/No)
- **Activities**: Extracurriculars (Yes/No)
- **Higher**: Wants higher education (Yes/No)
- **Reason**: Reason for choice of school:
  home - close to home
  reputation - reputation of school
  course - preference of course
  other - other reasons

Numerical Variables

- **Absences**: Number of School Absences
- Data clustered towards the top left
- Very slight Negative Correlation

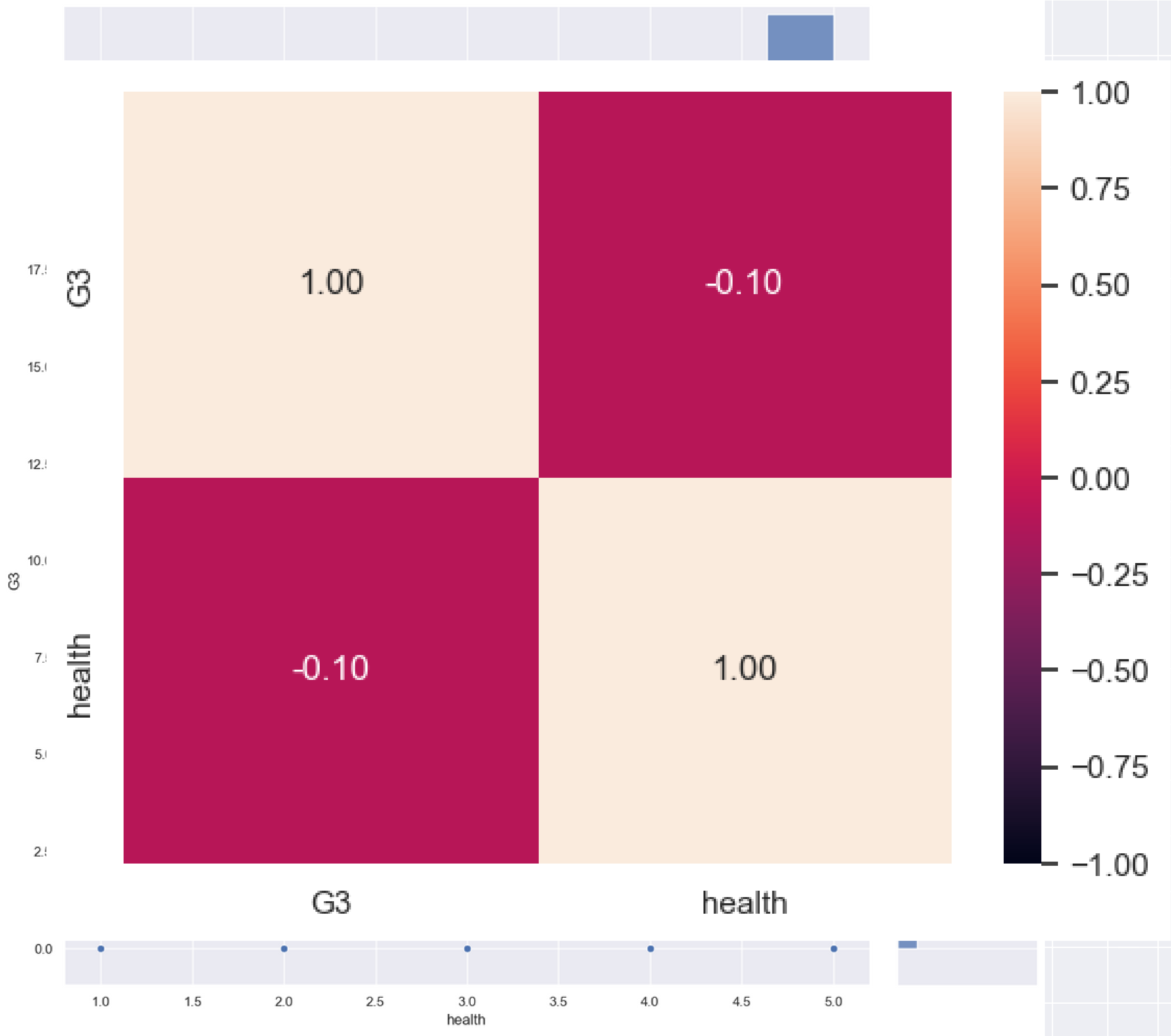


Numerical Variables

- **Health**: Current Health Status
  (The higher the better)

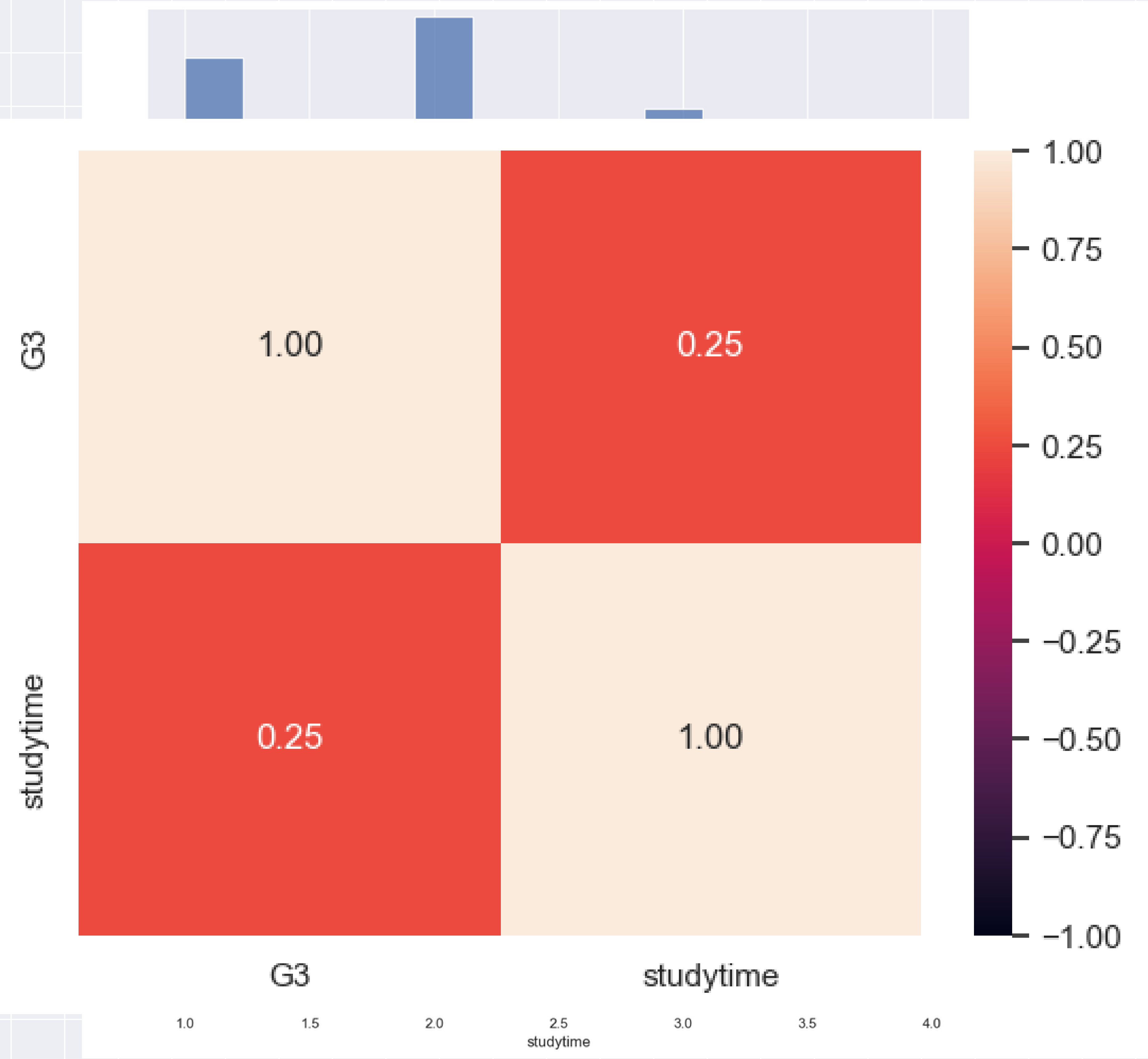
- Data spread out across entire graph

- Very slight Negative Correlation



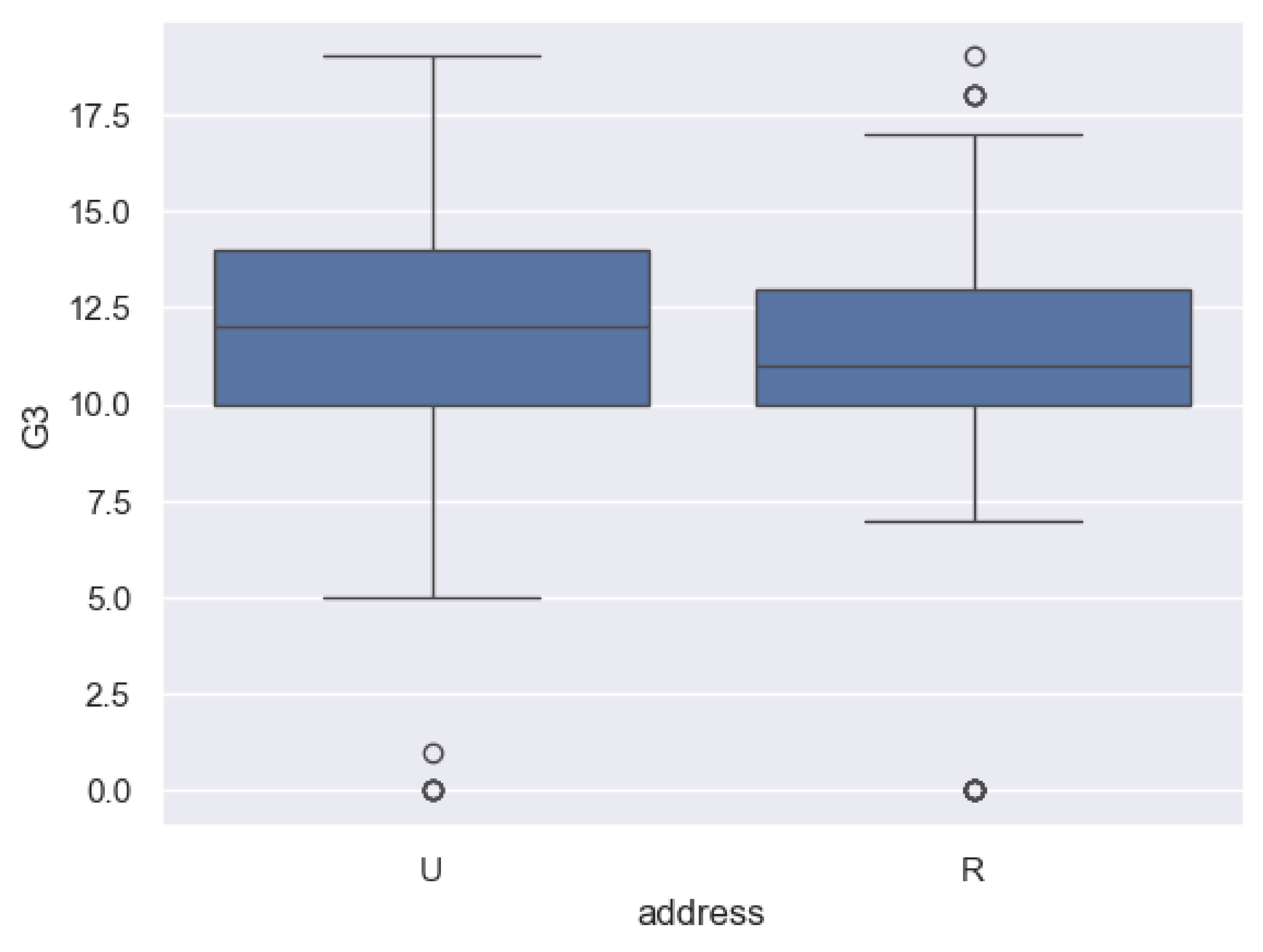
Numerical Variables

- **Study Time:** Weekly Study Time
- Most data at the middle portion of the graph
- Slight Positive Correlation



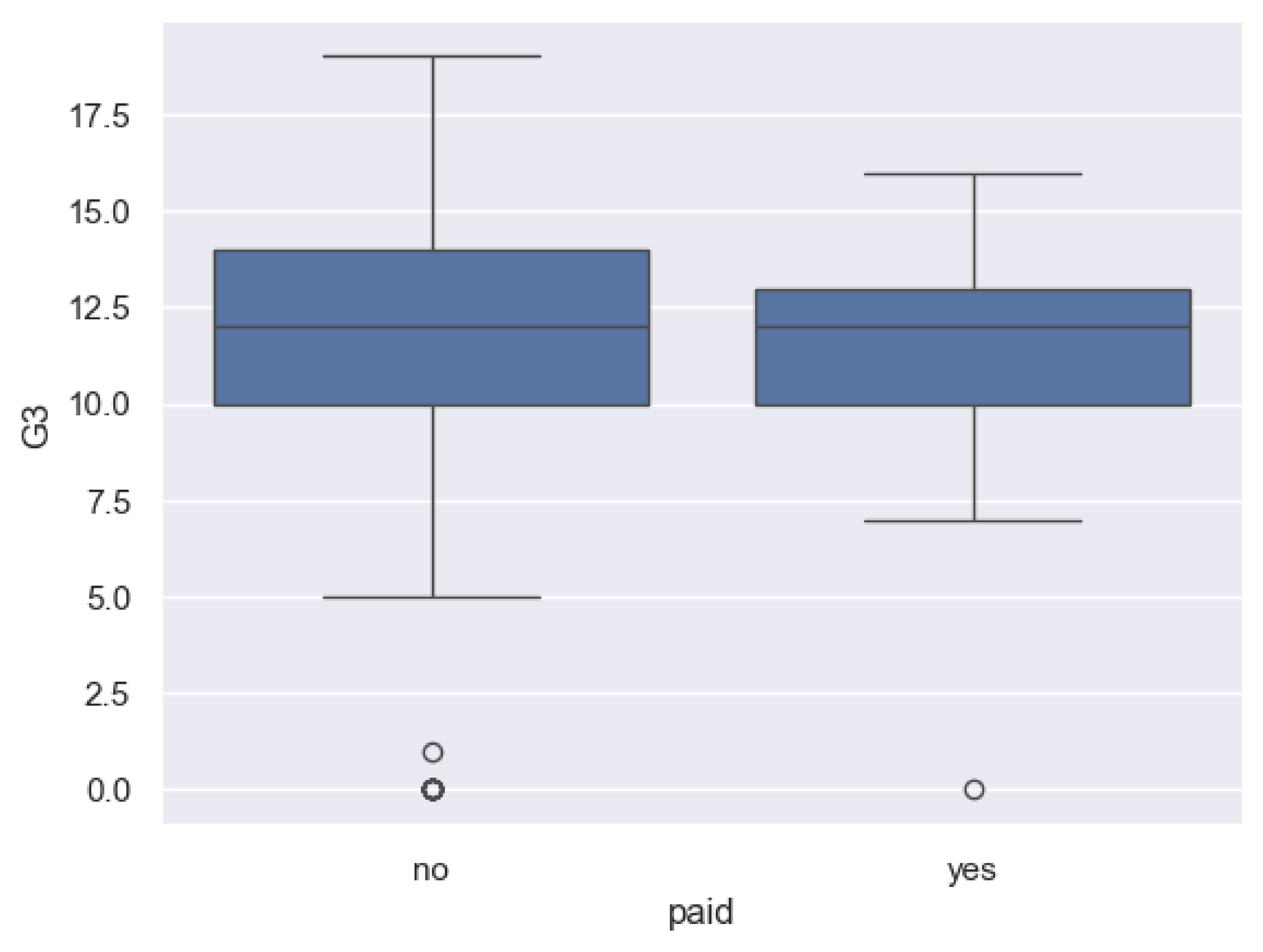
Categorical Variables

- **Address**: Rural/Urban
- Urban has slightly higher median G3
- Urban has a larger range of scores



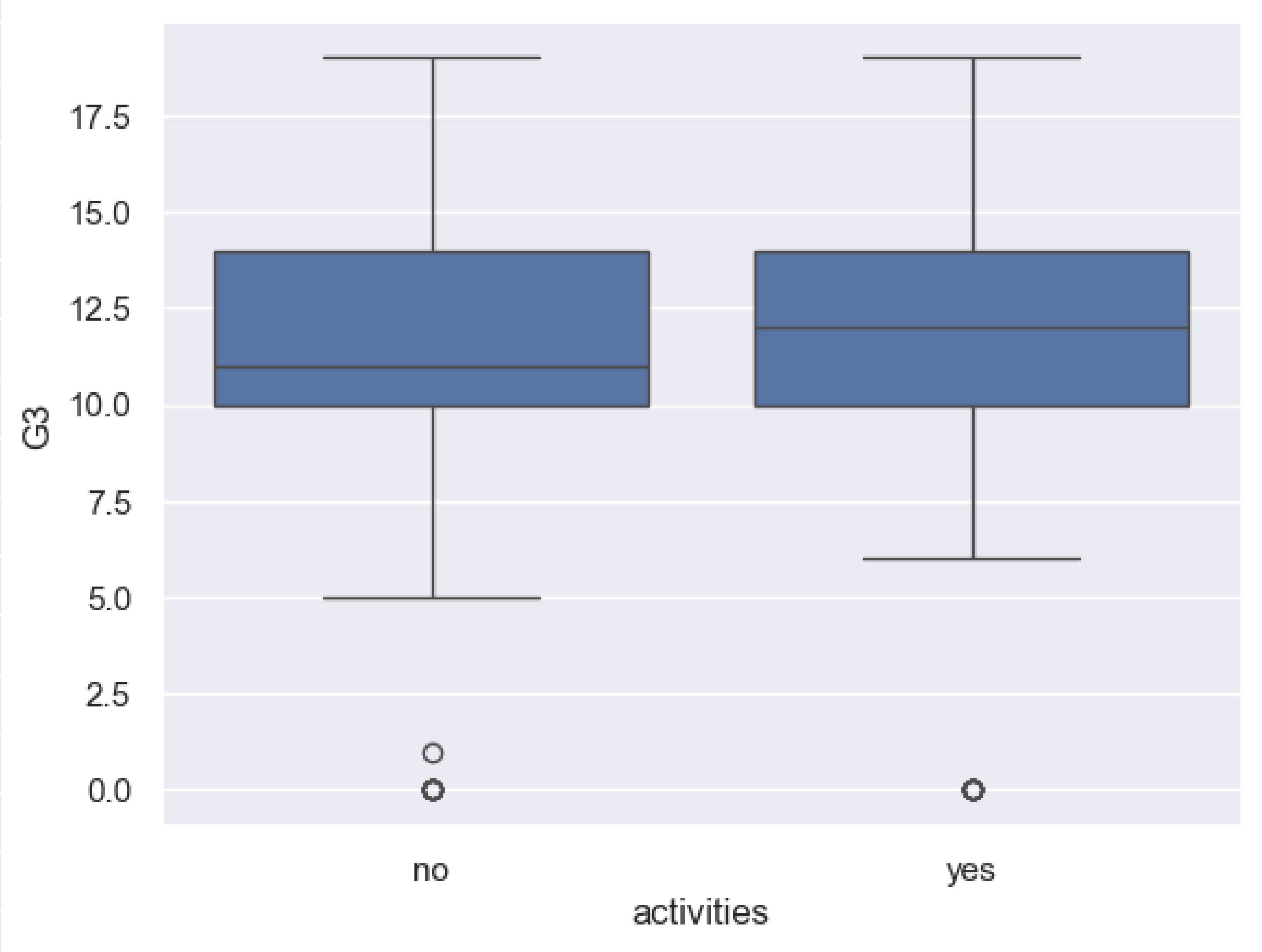
Categorical Variables

- **Paid**: Extra Tuition classes (Yes/No)
- Both have the same median score
- ‘No’ has a larger range of scores



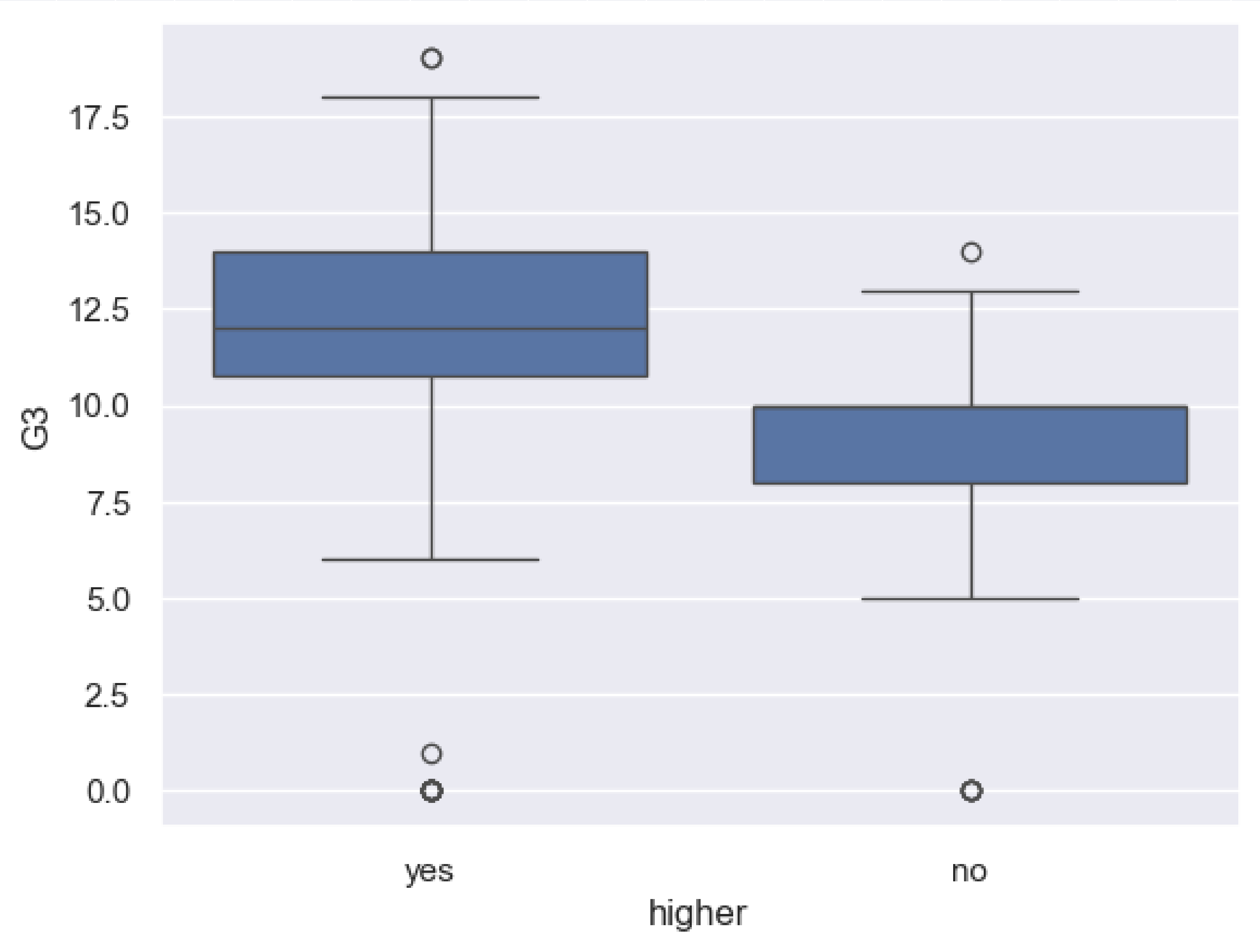
Categorical Variables

- **Activities**: Extracurriculars (Yes/No)
- ‘Yes has slightly higher median score
- Both have similar range of scores



Categorical Variables

- **Higher:** Wants higher education (Yes/No)
- ‘Yes’ has a larger median score and higher range of scores

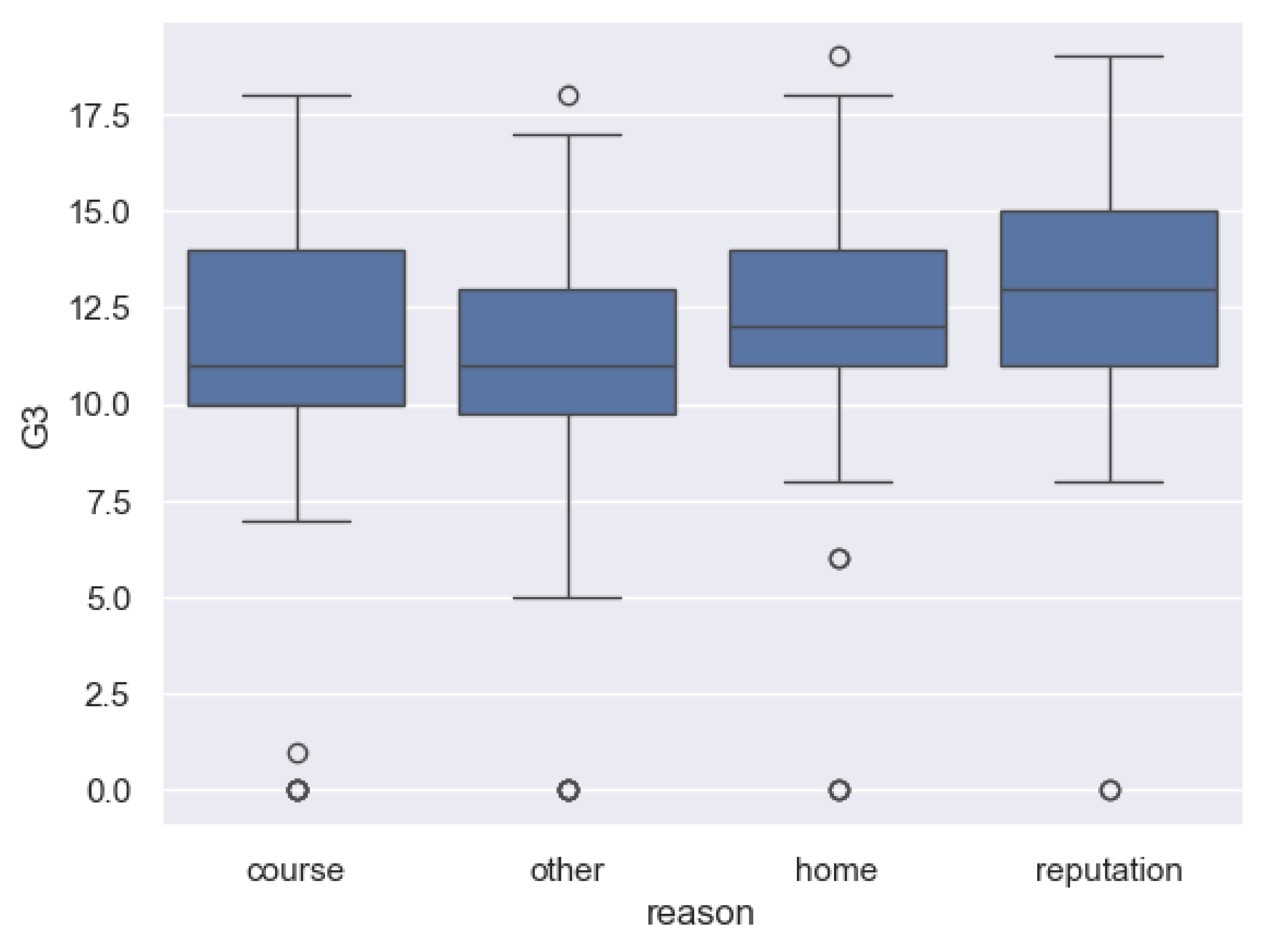


Categorical Variables

- **Reason:** Reason for choice of school:
  home - close to home
  reputation - reputation of school
  course - preference of course
  other

- Reputation has the highest median score

- Other has the largest range of scores

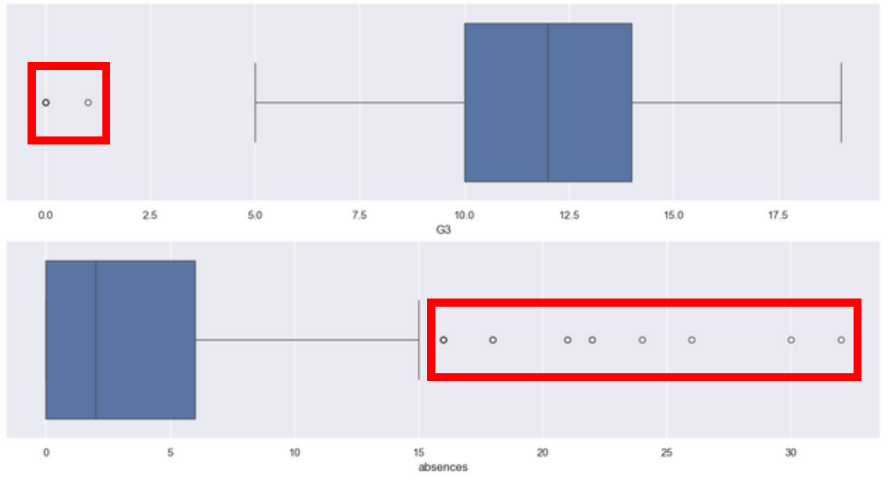


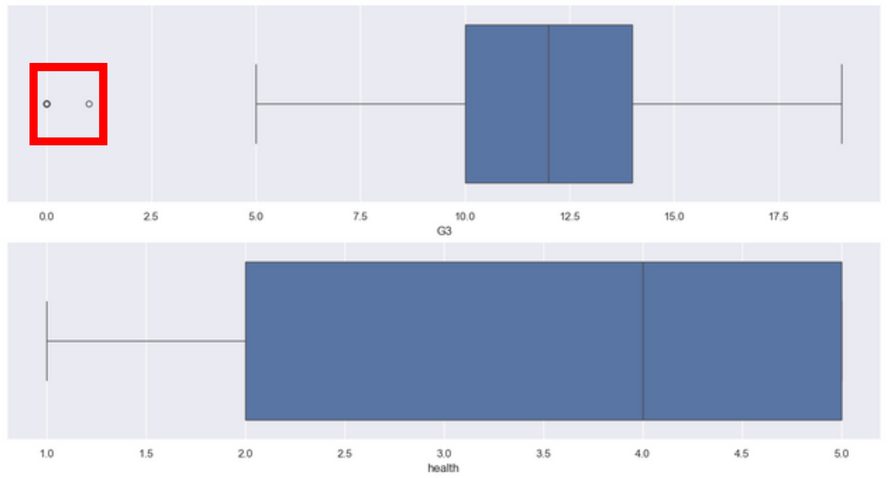
Data Cleaning and Preparation

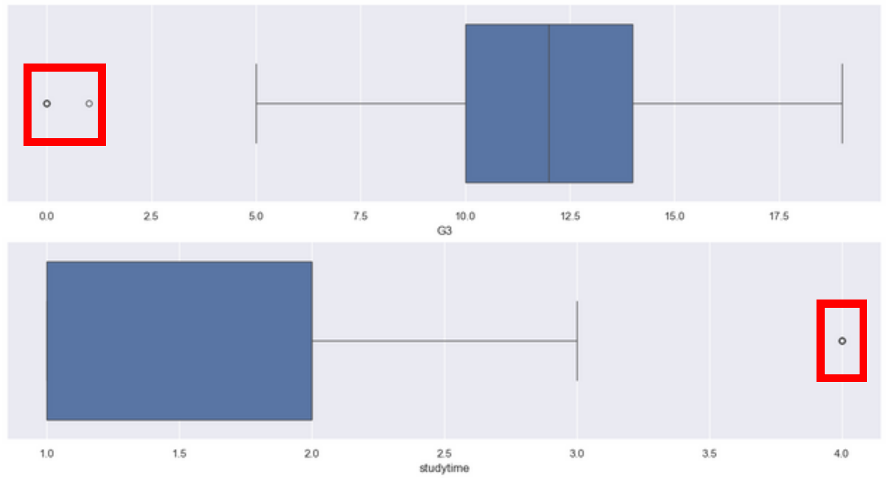
Numeric Variables

- Removal of outliers using boxplots (data outside the ‘whiskers’)

- Creation of separate data frames for each variable, containing both the variable data and G3 data







Categorical Variables

- Creation of separate data frames for each variable, containing both the variable data and G3 data

03

Machine Learning

ML problems and techniques

Outcome: determine the final G3 score of students

Type of problem: **Regression**

- Score is a numerical value
- Analyse numerical and categorical variables from data set
- Numerical variables: Linear/Random Forest model to determine Explained Variance (R^2)
- Categorical variables: Decision Tree to determine Classification Accuracy
- Random Forest model to compare numerical and categorical variables

Cross Validation: K-fold

Purpose:

⭐ Reduce the variance of model

⭐ Reduce overfitting

Key:

- Dataset is divided in k subsets/folds
- Model is trained and evaluated k times
- Higher accuracy

Numeric Data

Absence, Health, Studytime

Attempt 1: using linear regression model to find explained variance

Numeric variable: absence

```
#boxplot method
# Import essential models and functions from sklearn
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

#train and test in an 80:20 ratio
G3 = pd.DataFrame(outliers['G3'])   # Response
absences = pd.DataFrame(outliers['absences'])   # predictor

# Split the Dataset into Train and Test
X_train, X_test, y_train, y_test = train_test_split(absences, G3, test_size = 0.20 , random_state = 4

# Linear Regression using Train Data
linreg = LinearRegression()         # create the linear regression object
linreg.fit(X_train, y_train)        # train the linear regression model

# Coefficients of the Linear Regression Line
print('Intercept of Regression \t: b = ', linreg.intercept_)
print('Coefficients of Regression \t: a = ', linreg.coef_)
print()
# Predict G3 corresponding to absences Train
y_train_pred = linreg.predict(X_train)

# Plot the Linear Regression Line
f = plt.figure(figsize=(16, 8))
plt.scatter(X_train, y_train)
plt.scatter(X_train, y_train_pred, color = "r")
plt.show()
```

```
Intercept of Regression          : b =  [12.79112321]
Coefficients of Regression       : a =  [[-0.16637969]]
```

```
# Explained Variance (R^2)
print('Goodness of Fit of Model on Test dataset')
print("Explained Variance (R^2) \t:", linreg.score(X_test, y_test))

# Mean Squared Error (MSE)
def mean_sq_err(actual, predicted):
    '''Returns the Mean Squared Error of actual and predicted values'''
    return np.mean(np.square(np.array(actual) - np.array(predicted)))

mse = mean_sq_err(y_test, y_test_pred)
print("Mean Squared Error (MSE) \t:", mse)
print("Root Mean Squared Error (RMSE) \t:", np.sqrt(mse))
```

```
Goodness of Fit of Model on Test dataset
Explained Variance (R^2)        : -0.0019495838508398755
Mean Squared Error (MSE)        : 7.626711221909096
Root Mean Squared Error (RMSE)  : 2.7616500904186063
```

Error: Explained variance negative!

Numeric Data

Absence, Health, Studytime

Attempt 2: using kfold + random forest model

Numeric variable: absence

```
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestRegressor

X = pd.DataFrame(outliers['absences'])
y = pd.DataFrame(outliers['G3'])

# Define the number of folds
k = 5

# Initialize the KFold
kf = KFold(n_splits=k, shuffle=True, random_state=42)

# Initialize the Random Forest model
model = RandomForestRegressor(n_estimators=100, random_state=42)

# List to store cross-validation scores
cv_scores = []

# Perform k-fold cross-validation
for train_index, test_index in kf.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
```

```
# Train the model
model.fit(X_train, y_train)

# Evaluate the model
score = model.score(X_test, y_test)

# Append the score to the list of cross-validation scores
cv_scores.append(score)
```

```
Mean R^2 Score: 0.013754883215355517
```

Correction: explained variance positive

Numeric Data

Absence, Health, Studytime

Numeric variable: health

```python
X = pd.DataFrame(outliers['health'])
y = pd.DataFrame(outliers['G3'])

# Define the number of folds
k = 5

# Initialize the KFold
kf = KFold(n_splits=k, shuffle=True, random_state=42)

# Initialize the Random Forest model
model = RandomForestRegressor(n_estimators=100, random_state=42)

# List to store cross-validation scores
cv_scores = []

# Perform k-fold cross-validation
for train_index, test_index in kf.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
```

```python
model.fit(X_train, y_train)

# Evaluate the model
score = model.score(X_test, y_test)

# Append the score to the list of cross-validation sco
cv_scores.append(score)

lculate and print the mean of the cross-validation sco
```

```
R^2 Score: 0.001503349347629035
```

*Compare using **Random Forest** and generate **Explained Variance**

Numeric Data

Absence, Health, Studytime

Numeric variable: studytime

```python
X = pd.DataFrame(outliers['studytime'])
y = pd.DataFrame(outliers['G3'])

# Define the number of folds
k = 5

# Initialize the KFold
kf = KFold(n_splits=k, shuffle=True, random_state=42)

# Initialize the Random Forest model
model = RandomForestRegressor(n_estimators=100, random_state=42)

# List to store cross-validation scores
cv_scores = []

# Perform k-fold cross-validation
for train_index, test_index in kf.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    # Train the model
    model.fit(X_train, y_train)
```

```python
    # Train the model
    model.fit(X_train, y_train)

    # Evaluate the model
    score = model.score(X_test, y_test)

    # Append the score to the list of cross-validation scores
    cv_scores.append(score)

# Calculate and print the mean of the cross-validation scores
print("Mean R^2 Score:", np.mean(cv_scores))
```

```
Mean R^2 Score: 0.04641826761747649
```

*Compare using **Random Forest** and generate **explained variance**

Comparison

0<=R^2<=1

Numeric data	Explained Variance (R^2)
absence	0.0137548832153555517
health	0.001503349347629035
studytime	0.04641826761747649

As numeric variable “studytime” has a **R^2 value closest to 1**, this shows that the **goodness of fit** of random forest model with independent variable “studytime” and dependent variable “G3” is the **strongest**.

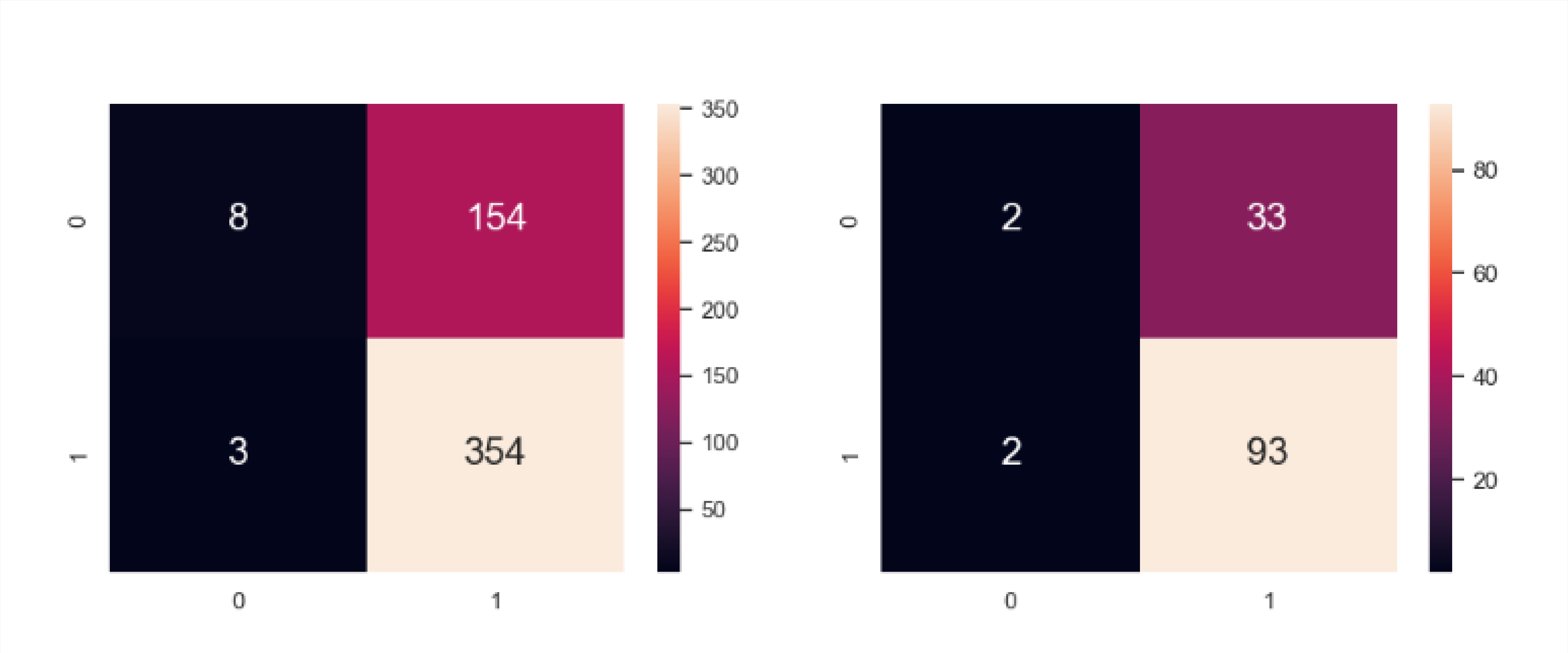
Categorical Data

address, paid, activities, higher education, reason

Categorical variable: address

train set

test set



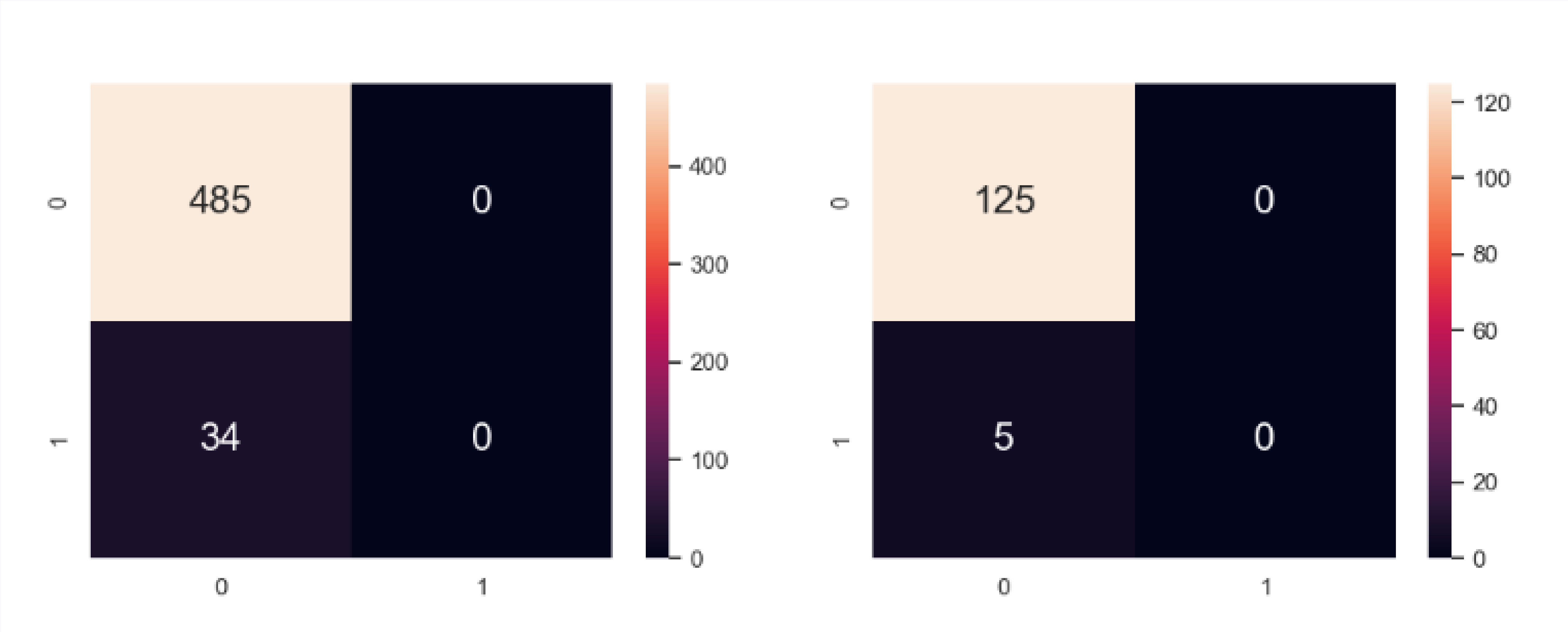

```
Goodness of Fit of Model        Train Dataset
Classification Accuracy         : 0.697495183044316

Goodness of Fit of Model        Test Dataset
Classification Accuracy         : 0.7307692307692307
```

Categorical Data

address, paid, activities, higher education, reason

Categorical variable: paid

train set test set



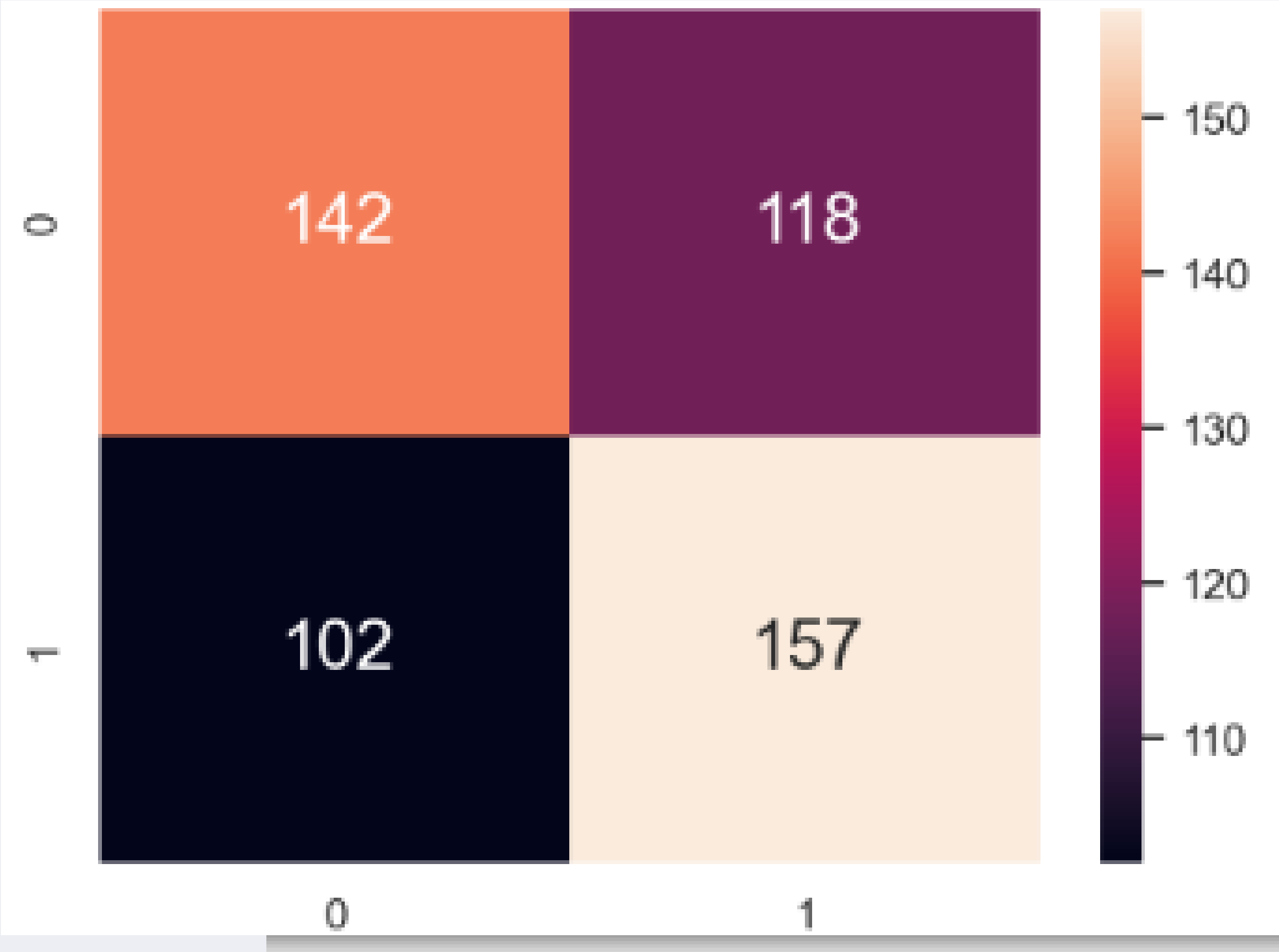

```
Goodness of Fit of Model     Train Dataset
Classification Accuracy      : 0.9344894026974951

Goodness of Fit of Model     Test Dataset
Classification Accuracy      : 0.9615384615384616
```

Categorical Data

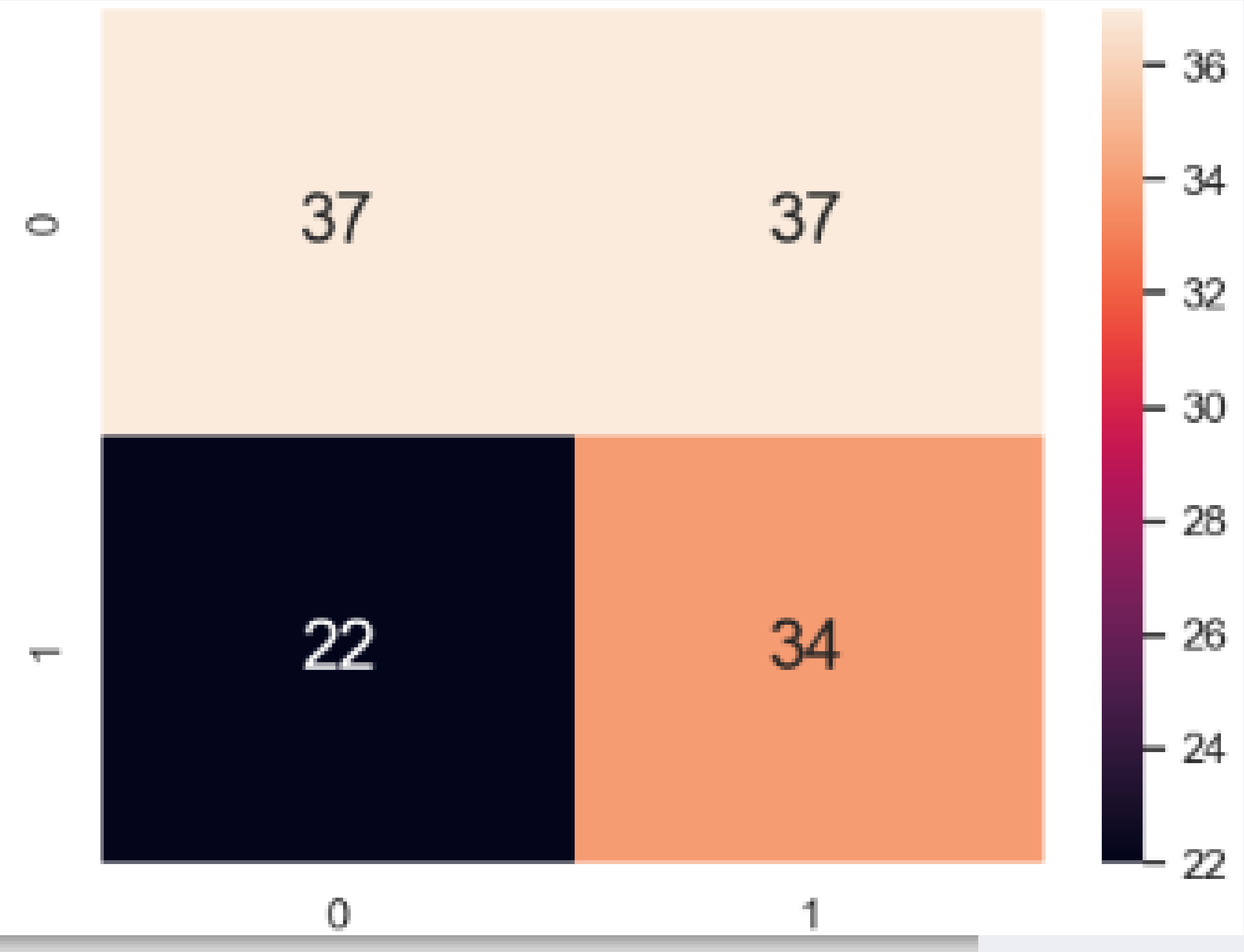
address, paid, activities, higher education, reason

Categorical variable: activities

train set



test set




```
Goodness of Fit of Model        Train Dataset
Classification Accuracy         : 0.5761078998073218

Goodness of Fit of Model        Test Dataset
Classification Accuracy         : 0.5461538461538461
```

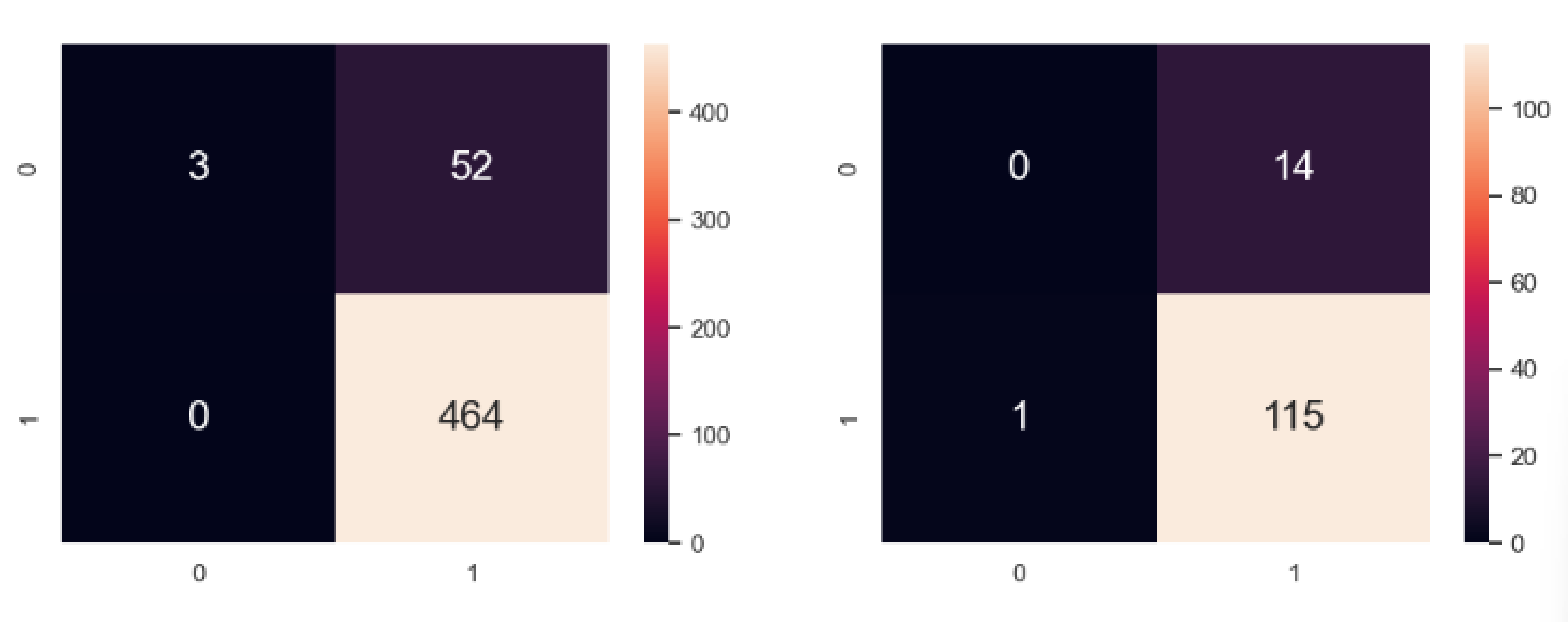
Categorical Data

address, paid, activities, higher education, reason

Categorical variable: higher education

train set

test set



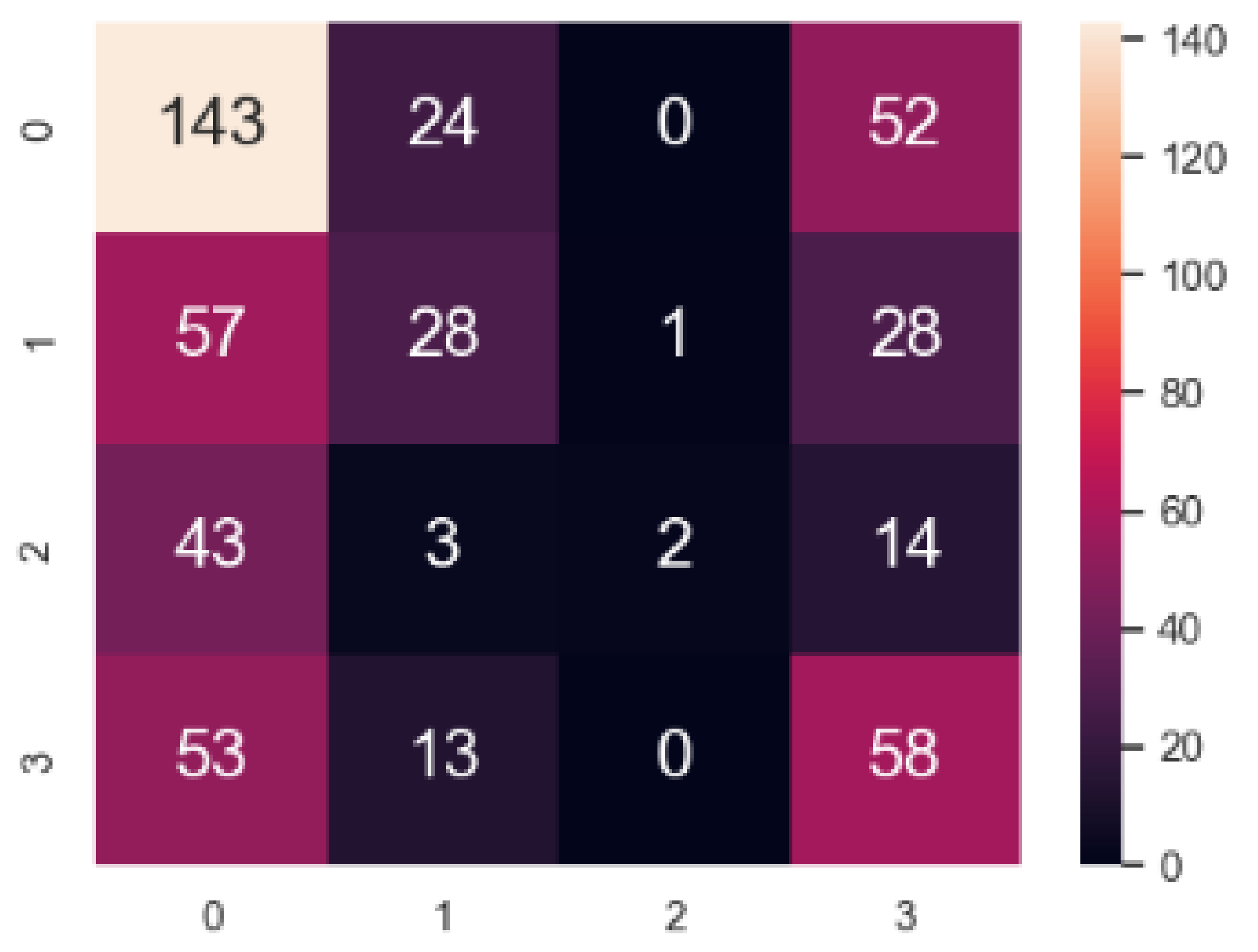

```
Goodness of Fit of Model        Train Dataset
Classification Accuracy         : 0.8998073217726397

Goodness of Fit of Model        Test Dataset
Classification Accuracy         : 0.8846153846153846
```

Categorical Data

address, paid, activities, higher education, reason

Categorical variable: reason

train set



test set


```
Goodness of Fit of Model        Train Dataset
Classification Accuracy         : 0.44508670520231214

Goodness of Fit of Model        Test Dataset
Classification Accuracy         : 0.35384615384615387
```

Comparison

0<=Classification Accuracy<=1

Categorical data	Classification Accuracy
address	0.7307692307692307
paid	0.9615384615384616
activities	0.8846153846153846
higher education	0.5461538461538461
reason	0.35384615384615387

The categorical data”paid” has the **highest Classification Accuracy** of **0.9615384615384616 (96%)**. This implies that the percentage of getting **true positives and true negatives** is the highest for “paid” among all categorical variables

Compare “studytime” and “paid”

- encode categorical variable”paid”
- compare both variables using Random Forest model

```python
# One hot encoding
from sklearn.preprocessing import OneHotEncoder

paid = pd.DataFrame(data['paid'])

# Encoding the 'paid' column which is categorical (assuming 'yes' = 1 and 'no' = 0)
one_hot_encoded = pd.get_dummies(paid, dtype=int)

# Defining the features and target variable
combined = pd.concat([data[['studytime']], one_hot_encoded], axis=1)
y = data['G3']

# Splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(combined, y, test_size=0.2, random_state=4)

# Define the number of folds
k = 5

# Initialize the KFold
kf = KFold(n_splits=k, shuffle=True, random_state=42)
```

```python
# Initialize the Random Forest model
model = RandomForestRegressor(n_estimators=100, random_state=42)

# List to store cross-validation scores
cv_scores = []

# Perform k-fold cross-validation
for train_index, test_index in kf.split(X_train):
    X_train_fold, X_val_fold = X_train.iloc[train_index], X_train.iloc[test_index]
    y_train_fold, y_val_fold = y_train.iloc[train_index], y_train.iloc[test_index]

    # Train the model
    model.fit(X_train_fold, y_train_fold)

    # Predict on the validation set
    y_pred = model.predict(X_val_fold)

    # Evaluate the model
    score = model.score(X_val_fold, y_val_fold)

    # Append the score to the list of cross-validation scores
    cv_scores.append(score)

# Calculate and print the mean of the cross-validation scores
print("Mean R^2 Score:", np.mean(cv_scores))

# Predict on the test set
y_pred_test = model.predict(X_test)

# Calculate and print the mean squared error on the test set
print("Mean Squared Error (MSE):", mean_squared_error(y_test, y_pred_test))
```

Compare “studytime” and “paid”

- encode categorical variable”paid”
- compare both variables using Random Forest model

```
Mean R^2 Score: 0.029098006759019034
Mean Squared Error (MSE): 10.414881422945598
```

Compare “studytime” and “paid”

why did we not remove outliers for “studytime”?

- Model Robustness
  - makes use of the ensemble method
- Valuable Insights
  - not a one-size-fits all approach

Final Analysis

Data	Explained Variance (R^2)	Mean Squared Error (MSE)
studytime	0.04641826761747649	6.381007474288847
studytime + paid	0.029098006759019034	10.414881422945598

Final Analysis

- “paid” is not being compared alone as a single model
  - “paid” uses decision tree to attain classification accuracy
  - not comparable with the result of Random Forest model (Explained Variance)
- Conclusion is under the assumption that “paid” is not a better independent variable than “studytime”

Final Analysis

- Compare **RME**: **combined model (10.4)** has a **higher MSE** compared to the **individual studytime model (6.38)**
- Compare **R^2**: **studytime (0.0464)** has a **slightly higher** R^2 score compared to the **combined model (0.0291)**

studytime alone may offer a simpler and more interpretable model compared to the combined model involving categorical encoding and interactions between variables.

Project Outcome and Conclusion

- Some variables like ‘paid’ and ‘activities’ were quite accurate in predicting grades
- Unexpectedly, combining both our best variables from both categories into a single model yielded poorer performance
- Moving forward, a possible solution could be to use more data that measures other aspects of a students learning, and combine it into 1 model

Thank you