

STUDENT GRADE PREDICTION USING MACHINE LEARNING ALGORITHMS

PROJECT REPORT

Submitted by:
ANANSH SINHA
(21BNM2057)

In partial fulfilment for the award of the degree of

BACHELOR OF SCIENCE
IN
Computer Science, Statistics and Mathematics



Chandigarh University
April, 2024



CHANDIGARH
UNIVERSITY
Discover. Learn. Empower.

BONAFIDE CERTIFICATE

Certified that this project report “**Student Grade Prediction using Machine Learning Algorithms**” is the bonafide work of **Anansh Sinha (21BNM2057)**, who carried out this project work under my supervision.

Dr. Harish Nagar
Head of Department
Department of Mathematics

Prof. Gagninder Kaur
Supervisor
Department of Mathematics

Submitted for the project viva-voce examination held on

Internal Examiner

External Examiner

TABLE OF CONTENTS

Abstract.....	i
List of Abbreviations.....	ii
Chapter 1. Introduction.....	1
1.1 Identification of Relevant Contemporary Issue	
1.2 Identification of Problem	
1.3 Identification of Tasks	
1.4 Timeline	
Chapter 2. Literature Review/Background Study.....	3
2.1 Timeline of Reported Problem	
2.2 Proposed Solutions	
2.3 Review Summary	
2.4 Problem Definition	
2.5 Goals and Objectives	
Chapter 3. Design Flow/Process.....	7
3.1 Evaluation and Selection of Features	
3.2 Design Constraints	
3.3 Analysis of Features and Finalisation subject to Constraints	
3.4 Implementation	
3.5 Design Flow and Selection	
Chapter 4. Result Analysis and Validation.....	11
4.1 Imports and Data Preprocessing	
4.2 Data Analysis and Visualisation	
4.3 Model Creation and Validation	
References.....	23
User Manual.....	24

LIST OF FIGURES

Figure 1.1	3
Figure 3.1	9
Figure 3.2	10

ABSTRACT

In the fast evolving world of technology, artificial intelligence (AI) has become a very integral part of our daily lives. Using machine learning algorithms is a common practice to predict any sort of phenomenon which can be encapsulated in mathematical terms and equations, in order to improve quality of life. This project aims to introduce a quality of life shift in the paradigm of conventional teaching methods by empowering educators to step in when a student is prone to not perform well in any particular subject. By inputting various environmental and academic factors, an educator can identify and then take steps to ensure that each and every student under their care performs well in the subject. Students can also benefit from such a process, if allowed to view their own predicted performance, they can direct their time and resources in the right direction in order to understand the subject of concern.

LIST OF ABBREVIATIONS

B.Sc. - Bachelors of Science

CSM - Computer Science, Statistics and Mathematics

CU - Chandigarh University

UIS - University Institute of Science

CGPA - Cumulative Grade Point Average

SGPA - Semester Grade Point Average

CSV - Comma Separated Values

CHAPTER 1

INTRODUCTION

1.1 Identification of Relevant Contemporary Issue

- There is a need for prediction of student grades in order to accurately allocate pedagogical resources efficiently to those students who are prone to not scoring well, instead of students who are already able to score well with conventional learning methods.
- The relationship between study time and test scores in a course on learning principles for college education majors reveals the importance which information-processing theorists attribute to active learning strategies [1].
- If these predictions and assessments are made frequently, the resulting feedback provides a comprehensive way of ensuring that low scoring students are brought up to their full standards by the time of their final assessment [2]. This implies the need for a very concise tool which could provide such insights to educators at all levels, and thus be able to provide such feedback in a timely manner.
- These issues are regularly highlighted in studies involving both machine learning and improvement in teaching techniques [3], which call for a systematic and comprehensive way in which prediction could be made and those results be discreetly provided to the teachers in order to improve the quality of their particular courses.

1.2 Identification of Problem

- The lack of any accurate methods of identifying students who might be prone to not scoring well, and thus miss out on any further pedagogical tool or help required by them.
- Even with an abundance of data in regards to every aspect of life, especially in education, there is minimal analysis available to either educators or students for the data that is collected on any platform, such as on CUIMS (Chandigarh University Learning Management System) for this case.
- A primary concern is the collection of such data and the subsequent results which are published and inferred from the initial data collected.
- Even when a student is not performing well, there is a possibility that they do not approach their educator in order to seek guidance due to a variety of

reasons, including being intimidated by the subject itself or a false sense of shame.

1.3 Identification of Tasks

- To conduct a survey of students under any particular program.
- To process and visualise the data that is acquired.
- To create a model based on the type of data collected.
- To validate the model and modify it accordingly, if required.

1.4 Timeline

- Phase 1: Data collection using surveys
- Phase 2: Processing the data and visualising any apparent trends
- Phase 3: Model creation and validation

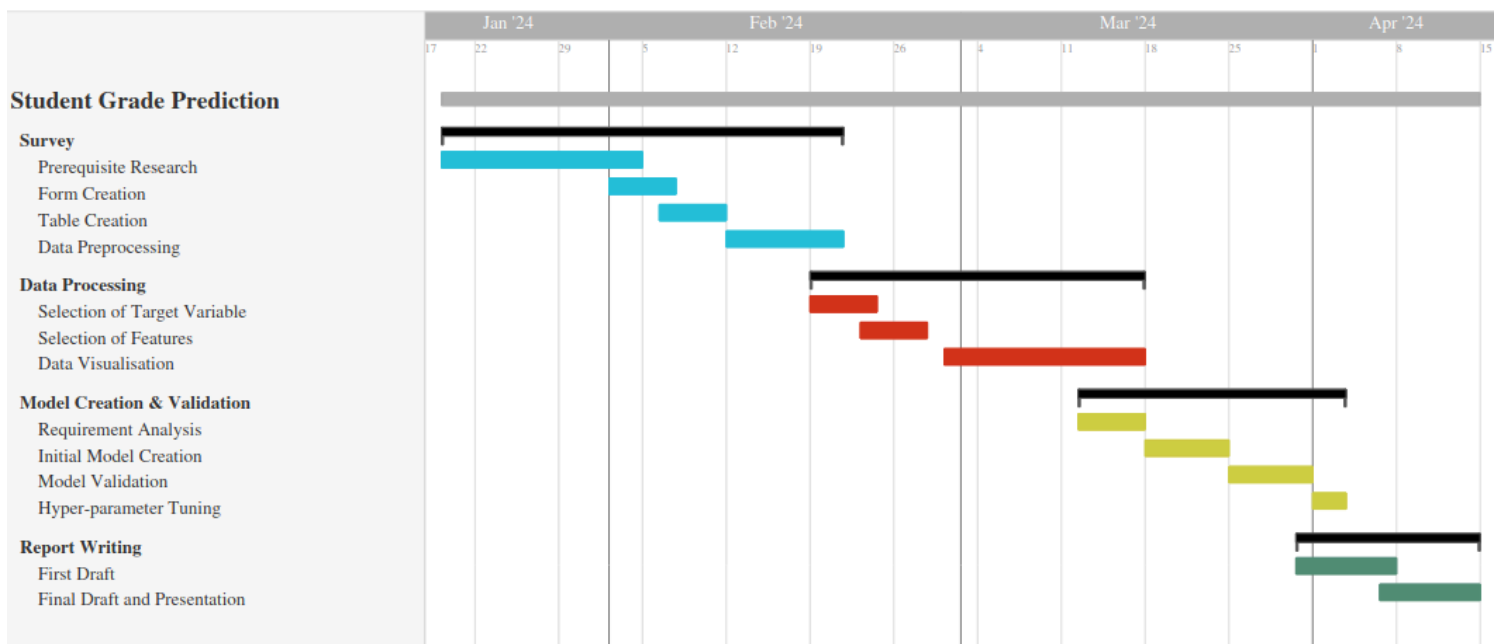


Fig 1.1 Gantt Chart of the Timeline

CHAPTER 2

LITERATURE REVIEW/ BACKGROUND STUDY

2.1 Timeline of reported problem

- In a study conducted by Oja et alia in 2013, the Noel-Levitz Student Satisfaction Inventory (SSI) was used in order to assess which areas students with lower grades were not satisfied with. One of the key areas highlighted was academic advising/counselling. The students reported that had they been provided with proper advice in time, they would have scored better [4]. In light of fairness considerations, it is untenable to anticipate that educators can offer counselling services devoid of appropriate metrics, predictive methodologies, or supportive tools.
- This notion is also supported by Sockalingam and Nachamma, who reported that of the processing elements; satisfaction on assessment had a direct influence on the course grades but only explained 1.3% of the variance in grades. Satisfaction on assessment was in turn positively and directly influenced by course content, course design and online discussion. These three factors explained 71.3% of the variance in assessment [5].
- Even studies conducted in India, especially by Kharb et al. have repeatedly stressed on the fact that the absence of machine learning and artificial intelligence tools for teachers in both high school and undergraduate level is concerning, since the methods of education have always been in tandem to developments in latest technologies in the same period [6].

2.2 Proposed solutions

- In the same study conducted by Kharb et al. the authors have discussed at length about how constructing machine learning models based on each institution's particular requirements and the students themselves, enforces a positive loop wherein any minor mistake made in prediction can be rectified immediately and increase the accuracy of the model.
- This same emotion is echoed by Sanusi et al. Even though their review has focused more on K-12 level, they are hopeful about its implication even in undergraduate and graduate levels. They mention the absolute necessity for this approach even in informal settings, and any further research into the

matter should also consider societal implication, which are reported to be mostly positive [7].

2.3 Review summary

- **Mahesh Gadhavi et al. [8]:** The project is a further extension of this particular paper, which mentioned the use of standard mathematical tools and modelling to create a linear regression model which would be a useful way of knowing whether a student should take extra steps into improving their own grades. The extensions made and proposed in this project are the inclusion of technological tools, referring to machine learning algorithms used herein along with the standard mathematical modelling already employed, as well as providing a platform for improvement in pedagogical methods.
- **Sanusi, Ismaila Temitayo, et al. [7]:** This particular study was conducted only for K-12 grades, and was very sparse in regards to employing this tool in the undergraduate level. No particular solutions were presented, and, as noted in the abstract, were only aimed at promoting further discussion into the relevance of such machine learning algorithms in all levels of teaching. This project aims to provide for a very particular analysis and implementation of the tools required for improvement in imparting education in all levels.
- **Meier, Yannick, et al. [3]:** This paper is an actual implementation of a machine learning algorithm to demonstrate that for 85% of the students, it can be predicted with 76% accuracy whether they are going to do well or poorly in the class after the fourth course week, among 700 UCLA (University of California, Los Angeles) undergraduates. This project aims to achieve similar goals, albeit with other factors in relation to the model.

2.4 Problem definition

1. Implementation

- a. Survey: A survey is to be conducted in order to get data from the students of undergraduate level from a particular course at a definitive time frame, in this case, the 5th semester of B.Sc. CSM.
- b. Data Processing: All the data that is collected should either be encoded into a proper format for categorical data, or into similar data

types for other types of data in order to ease the creation of a preliminary model.

- c. Model Creation: With all the data that is collected and processed, an initial regression model is to be created in order to give an initial idea of how the data is structured.
- d. Model Validation: This step is to ensure that all future data that could be incorporated is properly described by the model created. If this is not the case, then tuning of hyperparameters is to be done in order to better fit both the data that is given and the validation data.

2. Strategies for Implementation

- a. In case of survey, it is to be conducted in an anonymous way, that is, no identifying information is to be present in the data other than the required parameters. This is done in order to remove any bias from students divulging their grades who would have otherwise either given false data or not participated if the data was identifiable.
- b. For processing data, appropriate steps should be taken to facilitate ease of the algorithm to process such data. This might include the usage of one hot encoding method to encode the categorical values into numerical values.
- c. To select the best possible model, a multitude of measuring parameters should be used, such as mean squared error (MSE), mean absolute error (MAE) and R-squared, to compare each model that is created with each other.

3. Considerations for Implementation

- a. Not collecting any identifiable information; Similar concerns as 2(a) of section 2.2, any identifiable information might deter a lot of students with lower grades due to the stigma attached to it. Some students might fabricate the data, or may not even be a part of the survey. Thus, personal data should not be collected
- b. Overfitting: In order to fit all the data from the training dataset, there is a tendency to overfit the model, that is, it does not provide accurate estimation for any new data that is to be predicted from the given model. Thus, there should be a proper validation protocol in place

2.5 Goals and Objectives

- **Accessible**: One of the primary aims is to make this tool accessible to both educators, who can use this to provide any sort of assistance that may be

required by any student who is falling behind, and students, who can assess their own performance, even if they don't have a technical background.

- **Feature Identification:** There is a conscious strive to identify and evaluate relevant predictors of student grades, such as attendance, participation, homework completion, and assessment scores, to incorporate into the predictive model.
- **Enhancing Understanding:** This project aims to increase instructors' understanding of the complex aspects that affect students' success or failure and promote data-driven decision-making in educational settings.
- **Developing Predictive Model:** Creation of a robust predictive model capable of accurately forecasting students' grades based on a range of pertinent factors, including past academic performance, attendance, and participation.
- **Data Collection and Analysis:** Gathering and analysing comprehensive student data encompassing grades, attendance records, demographic information, and previous academic performance to inform the predictive model.

CHAPTER 3

DESIGN FLOW/PROCESS

3.1 Evaluation and selection of Features

In order to assess features for constructing a model, the method employed for constructing such a model was regression analysis. The features selected as the dependent variables are:

- ‘location’: This variable indicates where the student is residing. Possible values include:
 - ‘flat’: Indicates that a student resides in an independent capacity at some residence. They have to consider all their chores in addition to studying.
 - ‘day_scholar’: Indicates that a student resides locally, and thus does not have to consider any chores and can devote a sizable amount of their time to studying.
 - ‘hostel/PG’: Indicates that a student resides in a partially independent capacity at either university allocated hotels, or in a paying guest accommodation which functions similar to a hostel. They have to consider only a part of chores which a student living in a flat does.
- ‘board’: This variable indicates the board of education from which the student has completed their senior secondary education. Possible values are:
 - ‘CBSE’
 - ‘ICSE’
 - ‘State Board’
 - ‘Others’: These include students from International Boards and other such boards which do not come under the three already mentioned.
- ‘gender’: This variable indicates the biological gender of the student. Possible values are ‘M’ for male and ‘F’ for female.

Data was also collected for three subjects which was a part of the fifth semester curriculum of the students of B.Sc. CSM, in the department of mathematics (UIS),

CU. The subjects were Mathematics, Statistics and Computer Science. The metrics calculated were:

- Attendance: Measures the total attendance of a student over the course of the semester. An attendance of below 75% is grounds for disqualification from attempting the final exam. This was encapsulated for each subject as 'att_maths', 'att_stats' and 'att_cs'
- Focus Score: This is a self reported measure of how much the student felt engaged during the classes for the entire semester. Measured from 1 - 10, a higher score means they were engaged over a longer duration over the semester. For each subject, the variables were 'focus_maths', 'focus_stats' and 'focus_cs'.

The target variable is the CGPA, which is the final score during the entire course of the semester. For each individual subjects, the target variable would become the SGPA

3.2 Design Constraints

The design constraints pertaining to a regression model is the selection of features. A huge variety of data may have been collected, but only the most useful data points should be considered in order to have a feasible model. The model should have sufficient features, but a large number of features may increase the complexity of the prediction. It is also worth noting that the features themselves should not be auto-correlated or multicollinear in order to avoid any false or wrong interpretation of the data, which in result may cause less accurate results.

3.3 Analysis of Features and Finalisation subject to Constraints

- Location: This metric was chosen on the basis of testing whether having additional chores while not being close to family members affects the CGPA in any way.
- Gender: An attempt to quantify whether being a part of any particular gender affects the target variable is a very common practice in statistical surveys.
- Board: Some school boards are known for their introduction of harder topics to understand quite early in the education problems, whereas other boards gradually introduce these complex topics. Thus, this parameter should lead

to an understanding of whether the process of early education affects their performance later on in life or not.

- Attendance: In order to establish whether performing well in a particular subject is related to being present in most of the classroom lectures, this feature was chosen to be measured.
- Focus Score: The satisfaction from classroom interaction is found to be a very good measure of how well a student might perform in the course [5]. Thus, an attempt to capture this aspect of a student's interaction is very important for this project.

In the process of selection of all features that were collected, it was found that there was minimal auto-correlation and insignificant multicollinearity. Thus, the features selected are encapsulated in the jupyter-notebook cell below.

```
In [30]: x.columns
```

```
Out[30]: Index(['location', 'gender', 'board', 'sgpa_5', 'score_stats', 'att_stats',  
              'focus_stats', 'score_maths', 'att_maths', 'focus_maths', 'score_cs',  
              'att_cs', 'focus_cs'],  
             dtype='object')
```

Fig 3.1 Features for model training

3.4 Implementation

In order to complete the implementation of this project, the survey data was collected in .ods format, and then converted into a CSV file. This CSV file is ready to be analysed by a Python interpreter, using Jupyter Notebook as an environment to execute commands and provide clear and concise analysis into the initial data that was collected. The model was created with the help of a Python library called 'Scikit Learn', a popular machine learning library. All data visualisation was done with the help of 'Matplotlib' and 'Seaborn' libraries, both with various choices for ways in which to help visualise the data.

3.5 Design Flow and Selection

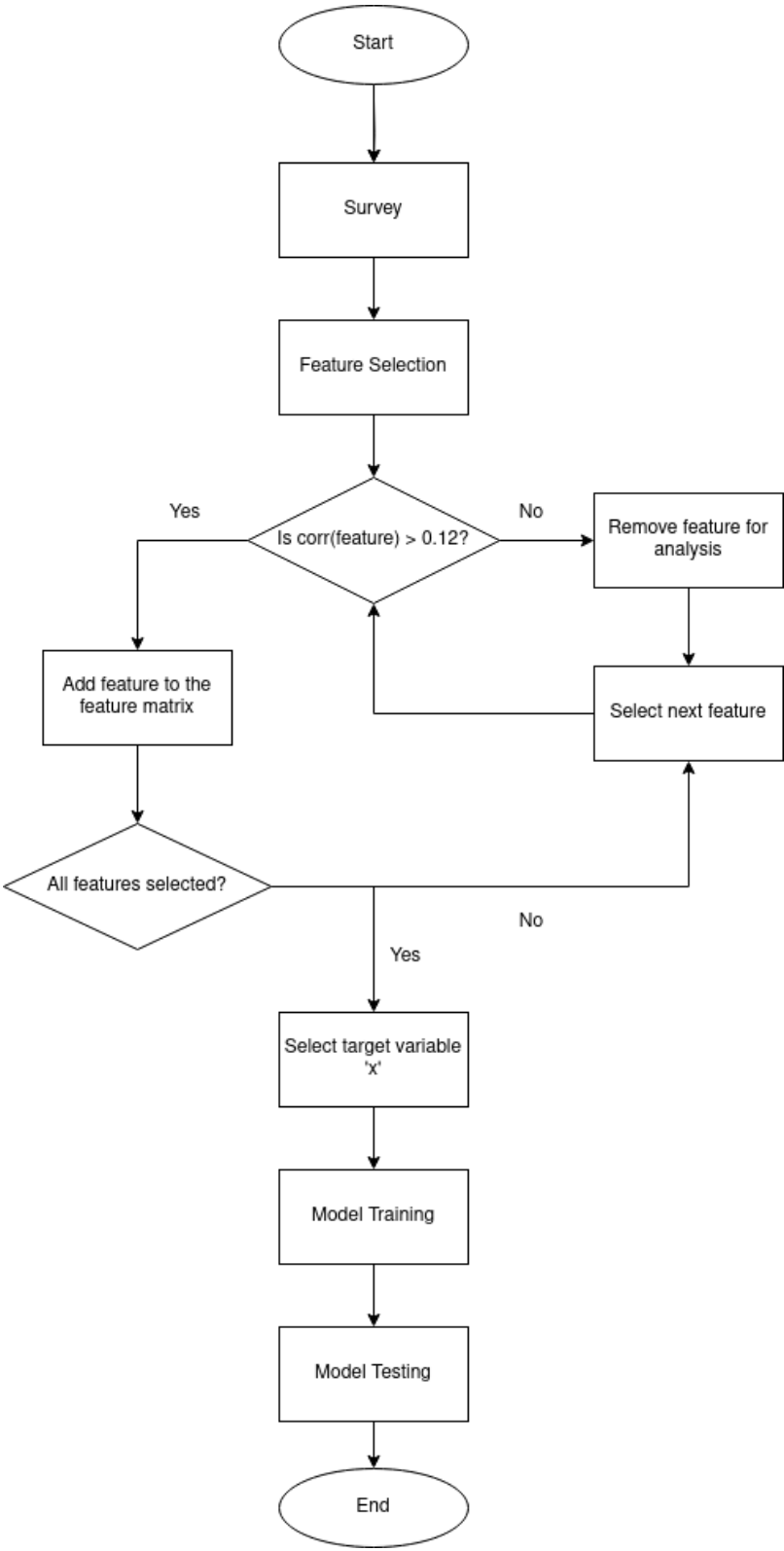


Fig 3.2 Flowchart of the Design Process

CHAPTER 4

RESULT ANALYSIS AND VALIDATION

4.1 Imports and Data Preprocessing

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from scipy.stats import gaussian_kde
from category_encoders import BinaryEncoder
from sklearn.linear_model import (
    LinearRegression,
    Ridge,
    Lasso,
    LogisticRegression,
    SGDClassifier,
    BayesianRidge,
)
from sklearn.tree import DecisionTreeRegressor
from sklearn import tree
from six import StringIO
from sklearn.ensemble import (
    RandomForestRegressor,
    GradientBoostingRegressor,
    BaggingClassifier,
    GradientBoostingClassifier,
    RandomForestClassifier,
)
from sklearn.svm import LinearSVC
from numpy.ma.core import sqrt
from sklearn.model_selection import GridSearchCV, cross_val_score, train_test_split
from numpy.polynomial.polynomial import polyfit
from sklearn.metrics import (
    accuracy_score,
    classification_report,
    mean_squared_error,
    r2_score,
    mean_absolute_error,
    confusion_matrix
)
import category_encoders as ce
from statsmodels.stats.stattools import durbin_watson
from category_encoders import BinaryEncoder
```

In [2]:

```
df = pd.read_csv('./student_survey.csv')
```

In [3]:

```
print(df.columns)
```

```
Index(['location', 'gender', 'cgpa', 'board', 'sgpa_5', 'score_stats',
      'att_stats', 'focus_stats', 'score_maths', 'att_maths', 'focus_maths',
      'score_cs', 'att_cs', 'focus_cs'],
      dtype='object')
```

In [5]:

```
categorical_columns = df.select_dtypes(include=['object']).columns.tolist()
```



```
print(categorical_columns)
print(len(categorical_columns))
```

```
['location', 'gender', 'board']
3
```

In [6]:

```
numerical_columns = df.select_dtypes(include=['number']).columns.tolist()
print(numerical_columns)
print(len(numerical_columns))
```

```
['cgpa', 'sgpa_5', 'score_stats', 'att_stats', 'focus_stats', 'score_maths', 'att_maths',
'focus_maths', 'score_cs', 'att_cs', 'focus_cs']
11
```

4.2 Data Analysis and Visualisation

In [7]:

```
df.describe().T
```

Out[7]:

	count	mean	std	min	25%	50%	75%	max
cgpa	51.0	7.432353	0.592849	5.38	7.20	7.5	7.845	8.4
sgpa_5	51.0	7.561569	0.656090	5.75	7.16	7.8	8.000	8.5
score_stats	51.0	51.392157	6.163857	30.00	49.00	52.0	55.500	60.0
att_stats	51.0	86.470588	6.688357	75.00	80.00	89.0	92.000	100.0
focus_stats	51.0	4.215686	2.484462	1.00	2.00	3.0	6.000	10.0
score_maths	51.0	47.147059	10.615222	11.00	40.00	50.0	54.500	60.0
att_maths	51.0	86.078431	6.743421	75.00	80.50	88.0	91.500	100.0
focus_maths	51.0	3.960784	2.366100	1.00	2.00	3.0	6.000	9.0
score_cs	51.0	45.294118	8.996208	5.00	40.50	46.0	51.500	58.0
att_cs	51.0	85.313725	7.524600	75.00	78.50	87.0	90.000	100.0
focus_cs	51.0	5.137255	3.072586	1.00	3.00	4.0	9.000	10.0

In [8]:

```
df.describe(include='object')
```

Out[8]:

	location	gender	board
count	51	51	51
unique	3	2	4
top	flat	F	CBSE
freq	18	30	32

In [4]:

```
gender = df['gender'].value_counts()
location = df['location'].value_counts()
board = df['board'].value_counts()

fig, axs = plt.subplots(1, 3, figsize=(15, 5))

axs[0].pie(gender, labels=gender.index, autopct='%1.1f%%', startangle=90)
axs[0].set_title('Gender')
```

```

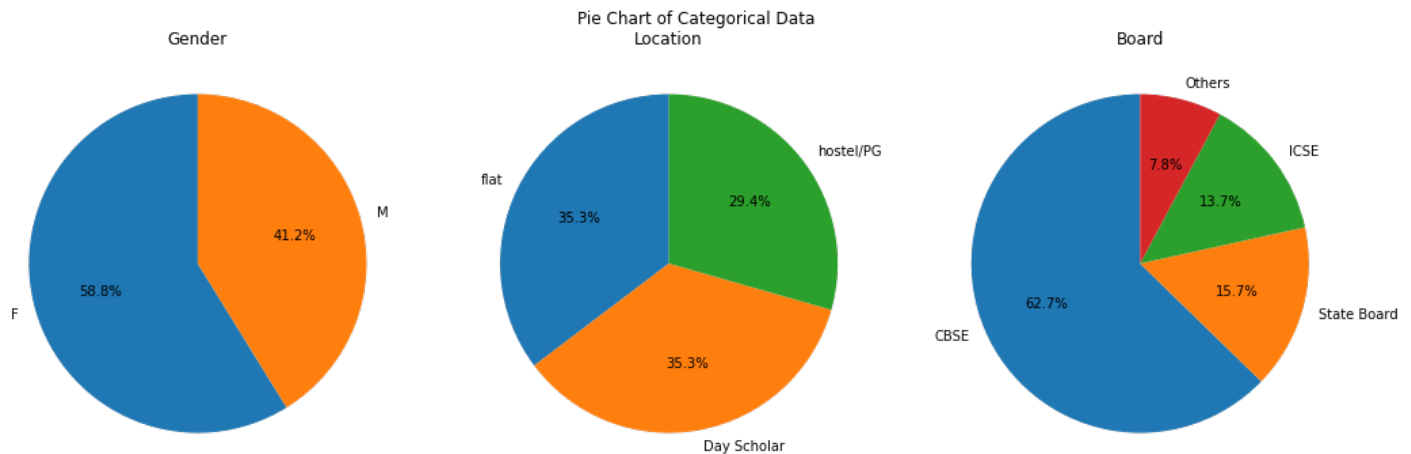
axs[1].pie(location, labels=location.index, autopct='%1.1f%%', startangle=90)
axs[1].set_title('Location')

axs[2].pie(board, labels=board.index, autopct='%1.1f%%', startangle=90)
axs[2].set_title('Board')

fig.suptitle('Pie Chart of Categorical Data')

plt.tight_layout()
plt.show()

```



In [9]:

```

# Define custom colors
custom_colors = ['#ff9999', '#004c99']

fig, ax = plt.subplots()
sns.barplot(x='gender', y='cgpa', data=df, errorbar=None, palette=custom_colors, ax=ax)

# Set labels and ticks
ax.set_xlabel('Gender', fontsize=14)
ax.set_ylabel('CGPA', fontsize=14)
ax.set_xticks([0, 1])
ax.set_xticklabels(["Female", "Male"])

plt.show()

```

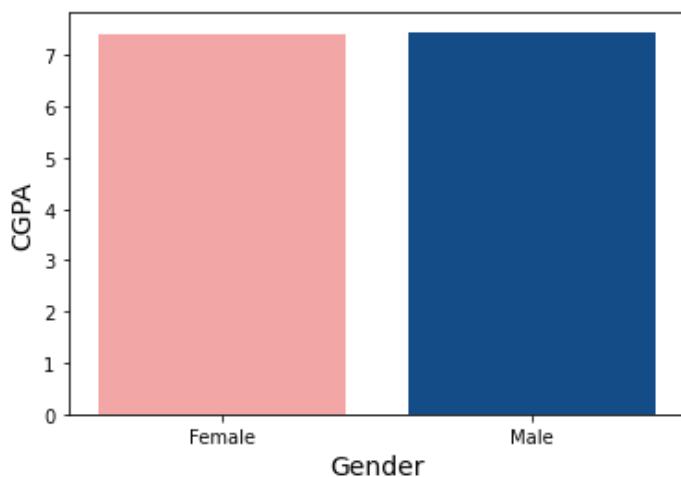
/tmp/ipykernel_3663/178767323.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. A assign the `x` variable to `hue` and set `legend=False` for the same effect.

```

sns.barplot(x='gender', y='cgpa', data=df, errorbar=None, palette=custom_colors, ax=ax)

```



Thus, there is no significant difference between the average performance of either males or females

In [31]:

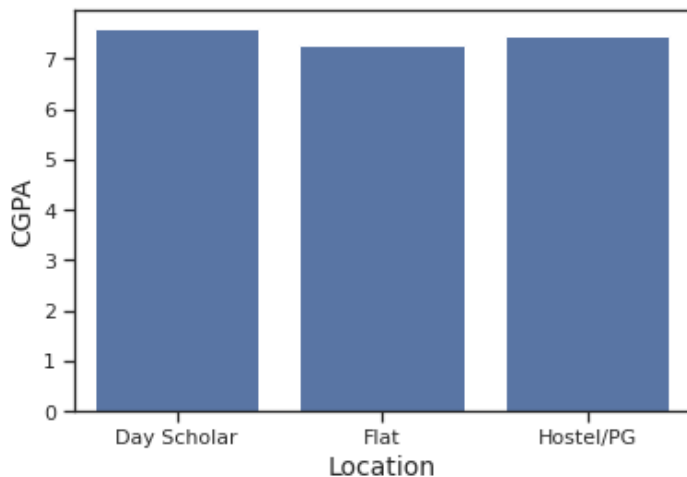
```
# Create figure and axes
fig, ax = plt.subplots()

# Plot the barplot
sns.barplot(x=df['location'], y=df['cgpa'], data=df, errorbar=None, ax=ax)

# Set labels and ticks
ax.set_xlabel('Location', fontsize=14)
ax.set_ylabel('CGPA', fontsize=14)
ax.set_xticklabels(["Day Scholar", "Flat", "Hostel/PG"])

# Show the plot
plt.show()
```

```
/tmp/ipykernel_3663/3628937291.py:10: UserWarning: FixedFormatter should only be used together with FixedLocator
ax.set_xticklabels(["Day Scholar", "Flat", "Hostel/PG"])
```



It can be observed that having a bit more allowance for studying can have a very slight advantage. Thus, Day scholars have a greater average score, followed by Hostel/PG students, and lastly Flat students (who usually have more chores during the day)

In [30]:

```
# Create figure and axes
fig, ax = plt.subplots()

# Define the order of bars
order = ["CBSE", "ICSE", "State Board", "Others"]

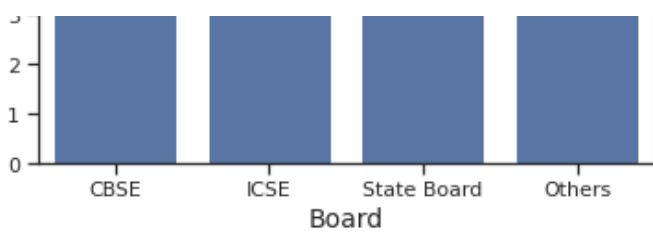
# Plot the barplot with specified order
sns.barplot(x=df['board'], y=df['cgpa'], data=df, order=order, errorbar=None, ax=ax)

# Set labels and ticks
ax.set_xlabel('Board', fontsize=14)
ax.set_ylabel('CGPA', fontsize=14)
ax.set_xticklabels(order)

# Show the plot
plt.show()
```

```
/tmp/ipykernel_3663/3238399730.py:13: UserWarning: FixedFormatter should only be used together with FixedLocator
ax.set_xticklabels(order)
```





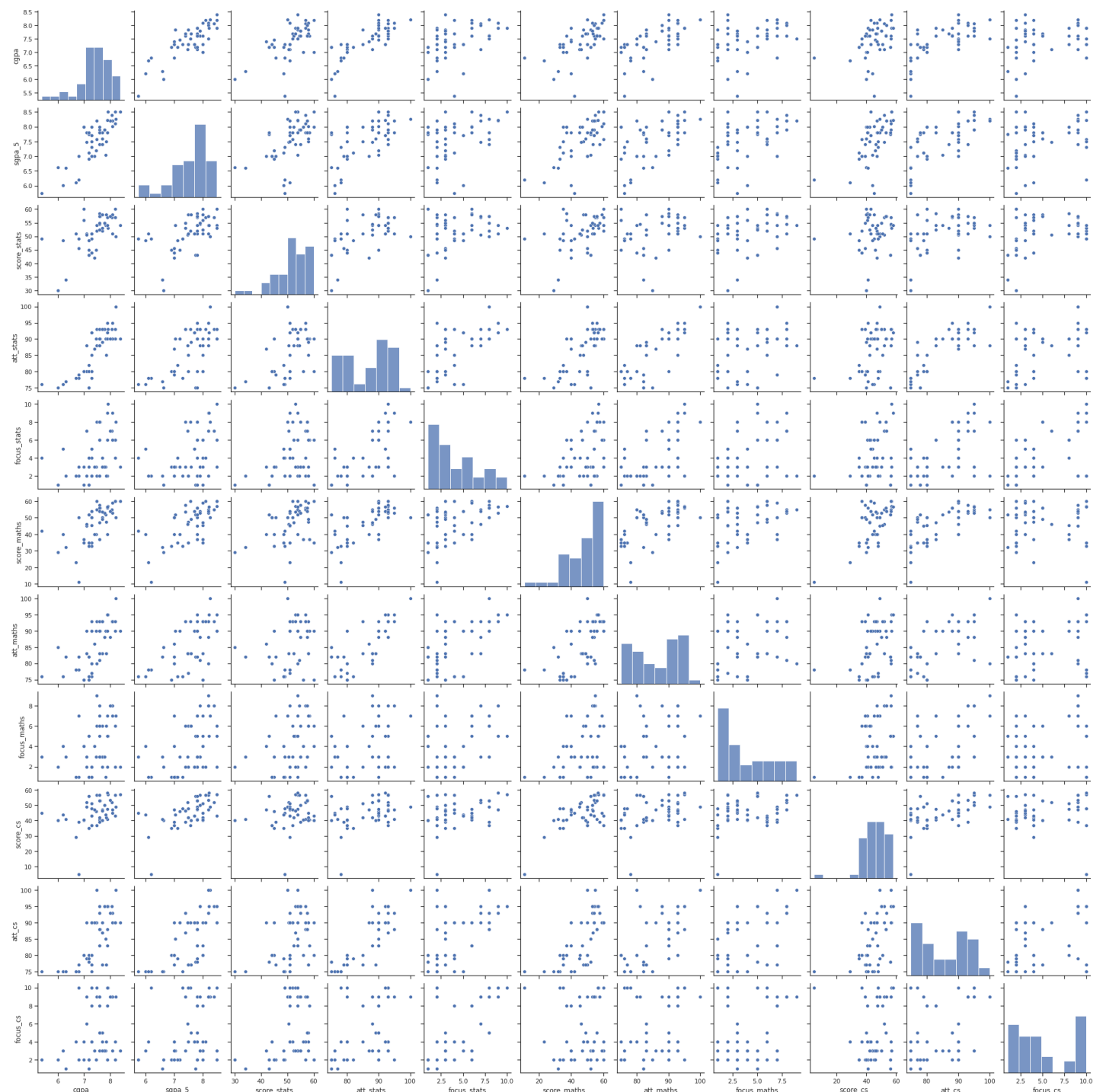
It can be observed that, in decreasing order that is, State Board, ICSE, CBSE and then others, the way curriculum is prepared for each education board in senior secondary level significantly affects the average grade that is achieved in undergraduate level

In [11]:

```
#plt.figure(figsize=(6, 4))
sns.set_theme(style="ticks")
sns.pairplot(df)
```

Out[11]:

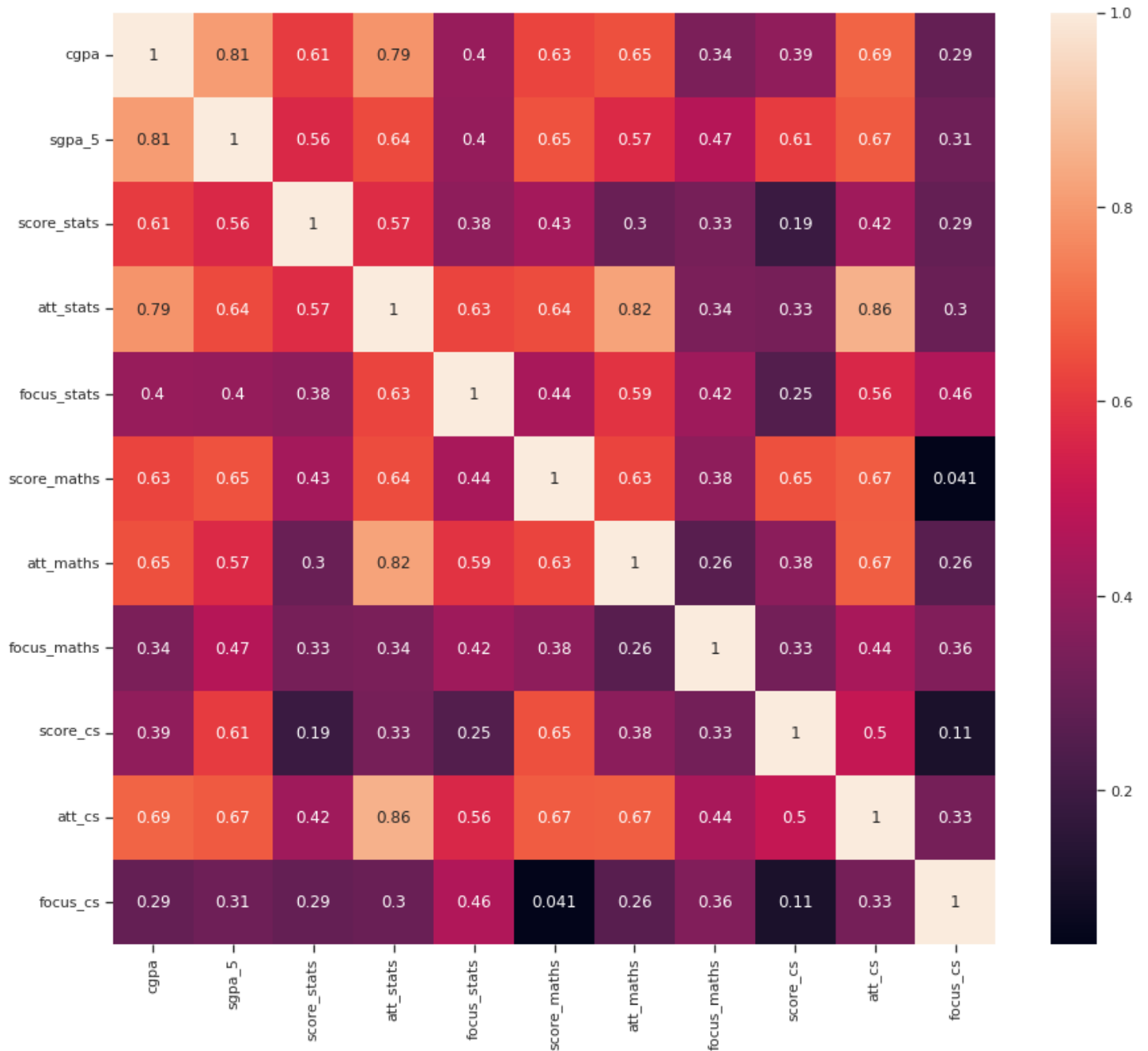
<seaborn.axisgrid.PairGrid at 0x738d95ab3610>



In [12]:

```
numeric_df = df.select_dtypes(include=['number'])
```

```
plt.figure(figsize=(15, 13))
sns.heatmap(numeric_df.corr(), annot=True)
plt.show()
```



In [13]:

```
score_cs = df['score_cs'].dropna()
score_stats = df['score_stats'].dropna()
score_maths = df['score_maths'].dropna()

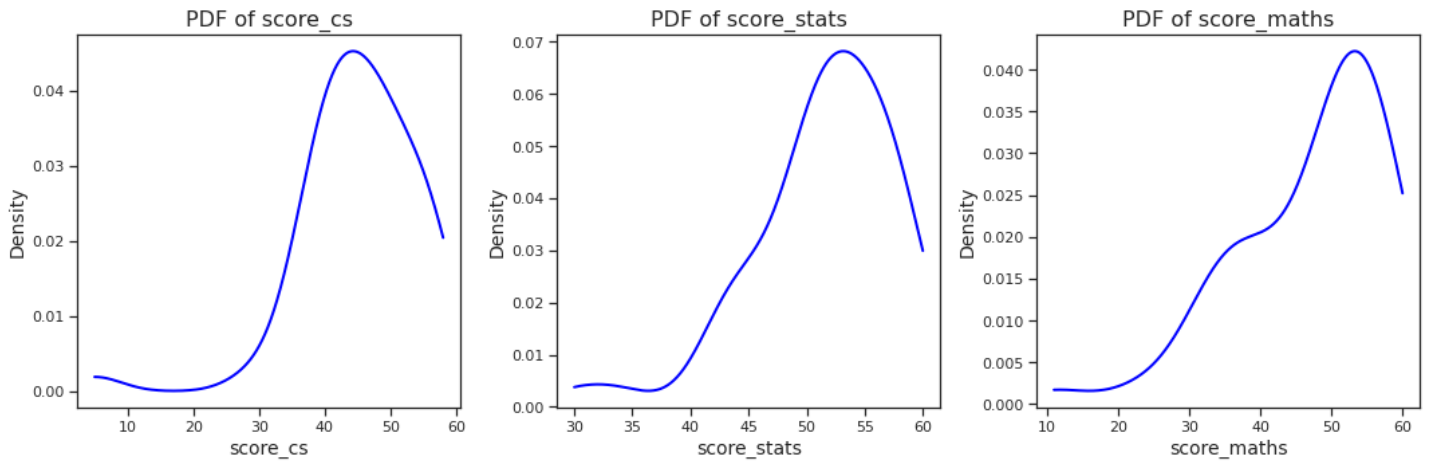
fig, axs = plt.subplots(1, 3, figsize=(15, 5))

# Loop through each column and plot KDE on a separate subplot
for i, score_data in enumerate([score_cs, score_stats, score_maths]):
    # Estimate the kernel density function
    kde = gaussian_kde(score_data)

    # Generate x values for plotting
    x_values = np.linspace(score_data.min(), score_data.max(), 1000)

    # Plot the PDF as a line graph
    axs[i].plot(x_values, kde(x_values), color='blue', linewidth=2)
    axs[i].set_title(f'PDF of {score_data.name}', fontsize=16)
    axs[i].set_xlabel(score_data.name, fontsize=14)
    axs[i].set_ylabel('Density', fontsize=14)
```

```
plt.tight_layout()
plt.show()
```



It can be observed that all the individual subject have very similar probability distributions.

In [14]:

```

maths_columns = ['score_maths', 'focus_maths', 'att_maths']
cs_columns = ['score_cs', 'focus_cs', 'att_cs']
stats_columns = ['score_stats', 'focus_stats', 'att_stats']

fig, axs = plt.subplots(1, 3, figsize=(15, 5)) # 1 row, 3 columns

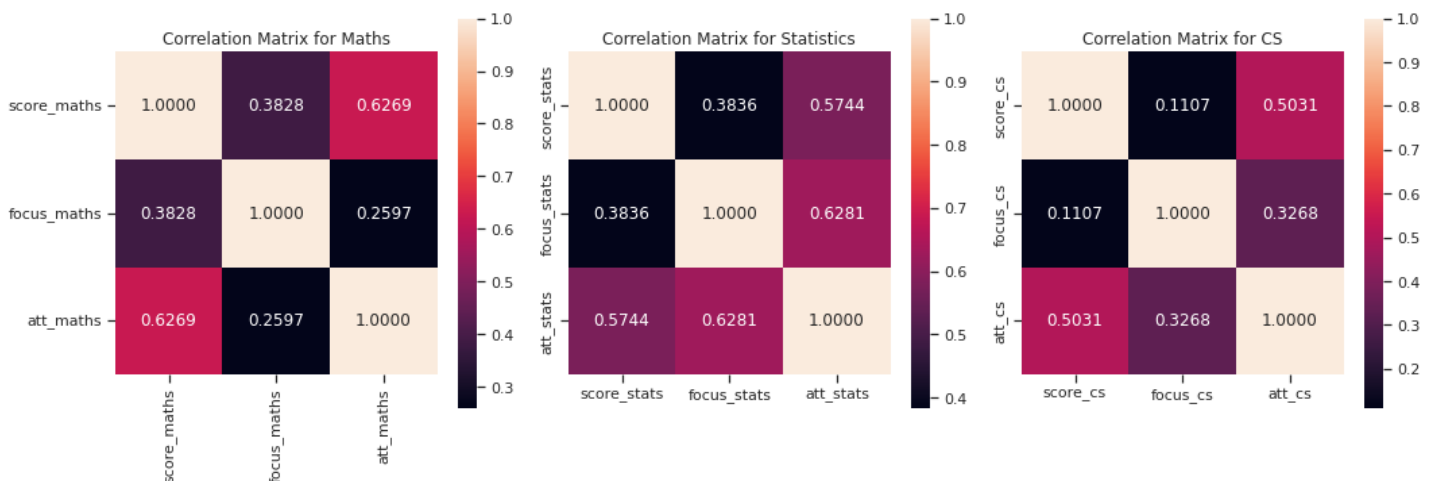
# Correlation matrix for maths columns
correlation_df_maths = df[maths_columns]
correlation_matrix_maths = correlation_df_maths.corr()
sns.heatmap(correlation_matrix_maths, ax=axs[0], annot=True, fmt=".4f", square=True)
axs[0].set_title('Correlation Matrix for Maths')

# Correlation matrix for stats columns
correlation_df_stats = df[stats_columns]
correlation_matrix_stats = correlation_df_stats.corr()
sns.heatmap(correlation_matrix_stats, ax=axs[1], annot=True, fmt=".4f", square=True)
axs[1].set_title('Correlation Matrix for Statistics')

# Correlation matrix for cs columns
correlation_df_cs = df[cs_columns]
correlation_matrix_cs = correlation_df_cs.corr()
sns.heatmap(correlation_matrix_cs, ax=axs[2], annot=True, fmt=".4f", square=True)
axs[2].set_title('Correlation Matrix for CS')

plt.tight_layout()
plt.show()

```



It can be observed that all 3 of the features for each subjects have very similar correlation values among each other

4.3 Model Creation and Validation

In [15]:

```
encoder = BinaryEncoder(cols=['board', 'gender', 'location'])
df_encoded = encoder.fit_transform(df)
df_encoded.head()
```

Out[15]:

	location_0	location_1	gender_0	gender_1	cgpa	board_0	board_1	board_2	sgpa_5	score_stats	att_stats	focus_stats	s
0	0	1	0	1	8.21	0	0	1	8.26	50.0	100	8	
1	0	1	1	0	5.38	0	0	1	5.75	49.0	76	4	
2	1	0	1	0	7.90	0	1	0	7.80	57.0	95	2	
3	0	1	0	1	8.20	0	0	1	8.50	60.0	90	6	
4	1	1	1	0	7.50	0	0	1	8.20	54.0	88	2	

In order to have a better understanding of how categorical data affects the target variable, binary encoding is used

In [16]:

```
THRESHOLD = 0.12
cgpa_corr = df_encoded.corr()["cgpa"]
```

For the purposes of this project, any feature with a correlation value less than 0.12 is considered to be insignificant

In [17]:

```
df_encoded_features = df_encoded.copy()

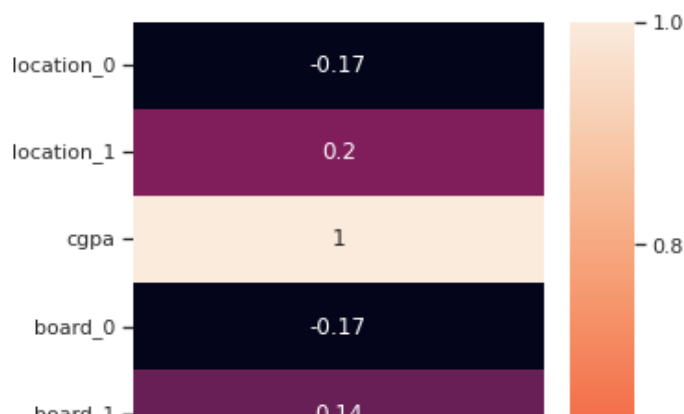
#For removal of features with less significant correlation
for key, value in cgpa_corr.items():
    if abs(value) < THRESHOLD:
        df_encoded_features.drop(columns= key, inplace=True)
```

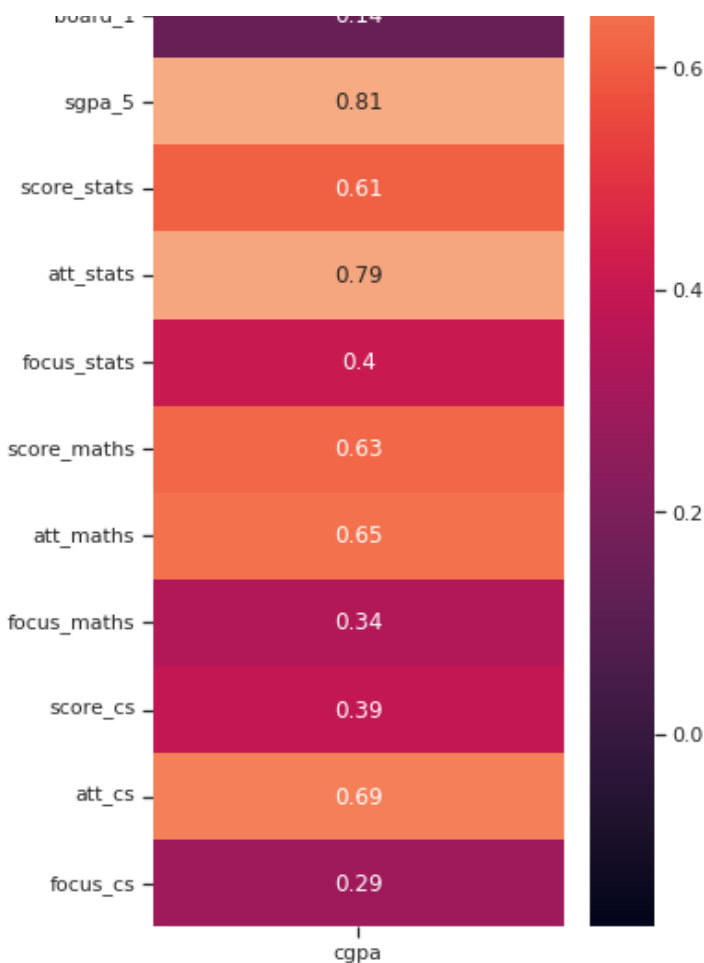
In [18]:

```
# Calculate the correlation matrix for all columns
correlation_matrix = df_encoded_features.corr()

# Extract the correlation values for the 'cgpa' column
correlation_with_cgpa = correlation_matrix['cgpa']

# Create a heatmap of the correlation values
plt.figure(figsize=(5, 13))
sns.heatmap(correlation_with_cgpa.to_frame(), annot=True, cbar=True)
plt.show()
```





This table shows the correlation of selected features with the target variable

In [19]:

```
# Function for iterating over multiple models
def train_regression_model(X, y):
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, shuffle=True, random_state=45)

    model1 = LinearRegression()
    model2 = BayesianRidge()
    model3 = RandomForestRegressor()
    model4 = GradientBoostingRegressor()
    model5 = DecisionTreeRegressor()
    model6 = Ridge()
    model7 = Lasso(tol=0.001)

    models = [model1, model2, model3, model4, model5, model6, model7]
    model_name_list = ['LinearRegression', 'BayesianRidge', 'RandomForestRegressor', 'GradientBoostingRegressor',
                       'DecisionTreeRegressor', 'Ridge', 'Lasso']

    # Dataframe for results
    results = pd.DataFrame(columns=['MAE', 'RMSE', 'RMSE by cross validation', 'MSE', 'R^2'], index=model_name_list)

    for i, model in enumerate(models):
        model.fit(X_train, y_train)

        y_test_pred = model.predict(X_test)

        mse = mean_squared_error(y_test, y_test_pred)
        mae = mean_absolute_error(y_test, y_test_pred)
        rmse = np.sqrt(mse)
        r_squared = r2_score(y_test, y_test_pred)

        scores = cross_val_score(model, X_test, y_test, scoring='neg_mean_squared_error', cv=5)
        rmse_cross_val = np.sqrt(-scores.mean())
```



```

model_name = model_name_list[i]
results.loc[model_name, :] = [mae, rmse, rmse_cross_val, mse, r_squared]

return results

```

This function incorporates a total of 7 models, which include Linear Regression, Bayesian Ridge, Random Forest Regressor, Gradient Boosting Regressor, Decision Tree Regressor, Ridge and Lasso Regression.

In [20]:

```

X = df_encoded_features.drop('cgpa', axis = 1) #Target Variable
Y = df_encoded_features['cgpa'] #Features

```

In [21]:

```

X_features = df_encoded.drop('cgpa', axis = 1)
Y_cgpa = df_encoded['cgpa']

```

In [22]:

```

train_regression_model(X_features, Y_cgpa)

```

Out[22]:

	MAE	RMSE	RMSE by cross validation	MSE	R^2
LinearRegression	0.24738	0.275328	1.603746	0.075805	0.468033
BayesianRidge	0.17023	0.206927	0.26359	0.042819	0.699517
RandomForestRegressor	0.175833	0.189234	0.350479	0.03581	0.748704
GradientBoostingRegressor	0.166026	0.178071	0.271102	0.031709	0.777479
DecisionTreeRegressor	0.183333	0.221284	0.483735	0.048967	0.656374
Ridge	0.239777	0.269983	0.264262	0.072891	0.488485
Lasso	0.092559	0.111753	0.388945	0.012489	0.91236

Since the Lasso model has the least MAE, RMSE, MSE and R^2, it is the best fitting model out of all models tested

In [23]:

```

#Lasso model training and prediction
X_train, X_test, y_train, y_test = train_test_split(X_features, Y_cgpa, test_size=0.1, s
huffle=True, random_state=42)

best_model_features = Lasso()
best_model_features.fit(X_train, y_train)

Y_test_pred2 = best_model_features.predict(X_test)

```

In [24]:

```

#Test for auto-correlation
residuals = y_test - best_model_features.predict(X_test)
dw_test_statistic = durbin_watson(residuals)

print(dw_test_statistic)

```

2.118219670943289

A score very close to 2 indicates that very negligible auto-correlation is present.

Since Lasso regression inherently penalizes multicollinearity, it is safe to assume that multicollinearity is not significant here.

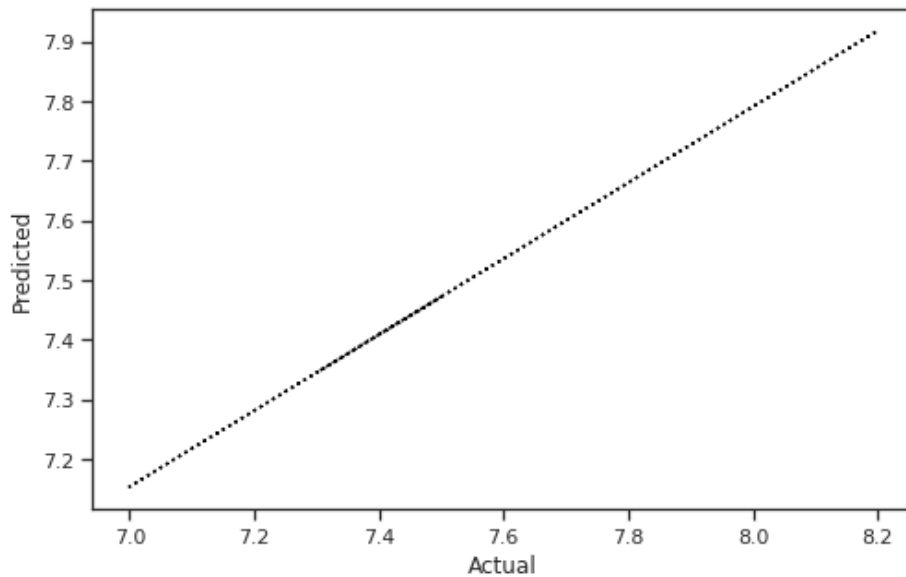
In [32]:

```
# Convert Pandas Series to NumPy arrays
y_test_np = np.array(y_test)
y_test_pred_np = np.array(Y_test_pred2)

# Perform polynomial fitting
m, n = np.polyfit(y_test_np, y_test_pred_np, 1)

# Plotting
plt.figure(figsize=(8,5))
#plt.scatter(x=y_test_np, y=y_test_pred_np, c="red")
plt.plot(y_test_np, m*y_test_np + n, ':", c="black")

plt.xlabel("Actual")
plt.ylabel("Predicted")
plt.show()
```



This plot shows a graphical representation of the model created

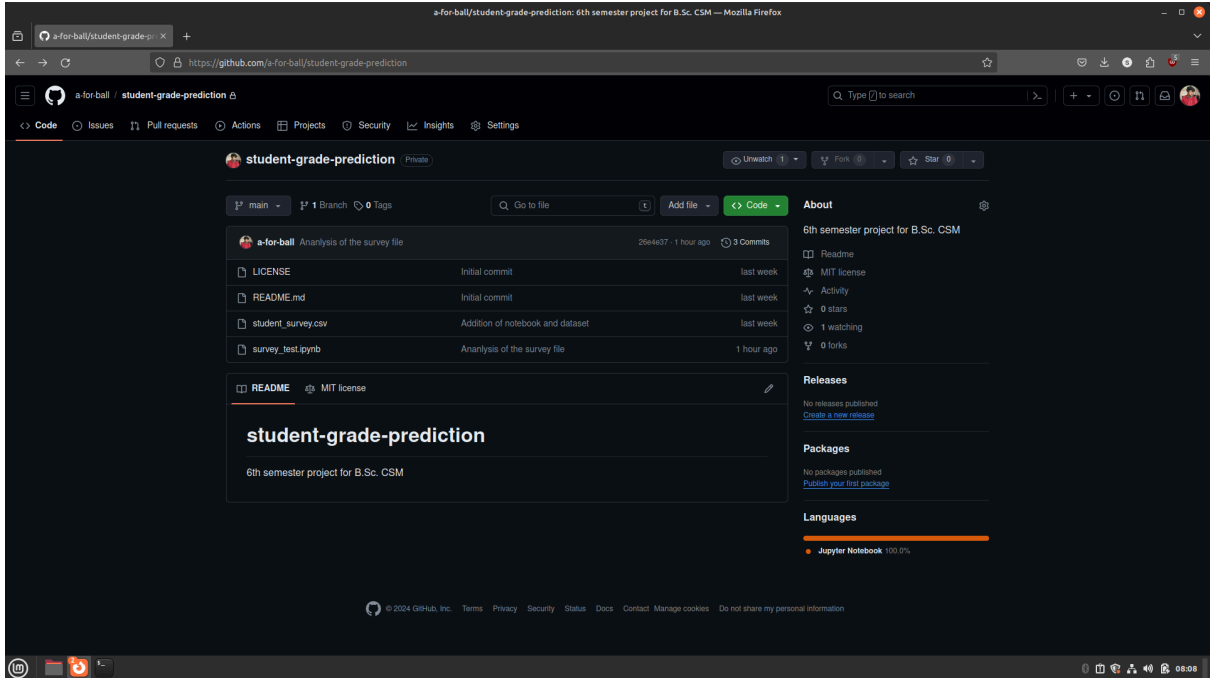
REFERENCES

- [1] Donald J. Dickinson & Debra Q. O'Connell (1990) Effect of Quality and Quantity of Study on Student Grades, *The Journal of Educational Research*, 83:4, 227-231, DOI: [10.1080/00220671.1990.10885960](https://doi.org/10.1080/00220671.1990.10885960)
- [2] Indira N. Z. Day, Floris M. van Blankenstein, Michiel Westenberg & Wilfried Admiraal (2018) A review of the characteristics of intermediate assessment and their relationship with student grades, *Assessment & Evaluation in Higher Education*, 43:6, 908-929, DOI: [10.1080/02602938.2017.1417974](https://doi.org/10.1080/02602938.2017.1417974)
- [3] Meier, Yannick, et al. "Predicting grades." *IEEE Transactions on Signal Processing* 64.4 (2015): 959-972.
- [4] Oja, Michelle. "Student satisfaction and student performance." *Journal of Applied Research in the Community College* 19.1 (2011): 47-53.
- [5] Sockalingam, Nachamma. "The Relation between Student Satisfaction and Student Performance in Blended Learning Curricula." *International Journal of Learning: Annual Review* 18.12 (2013).
- [6] Kharb, Latika, and Prateek Singh. "Role of machine learning in modern education and teaching." *Impact of AI Technologies on Teaching, Learning, and Research in Higher Education*. IGI Global, 2021. 99-123.
- [7] Sanusi, Ismaila Temitayo, et al. "A systematic review of teaching and learning machine learning in K-12 education." *Education and Information Technologies* 28.5 (2023): 5967-5997.
- [8] Gadhavi, Mahesh, and Chirag Patel. "Student final grade prediction based on linear regression." *Indian J. Comput. Sci. Eng* 8.3 (2017): 274-279.

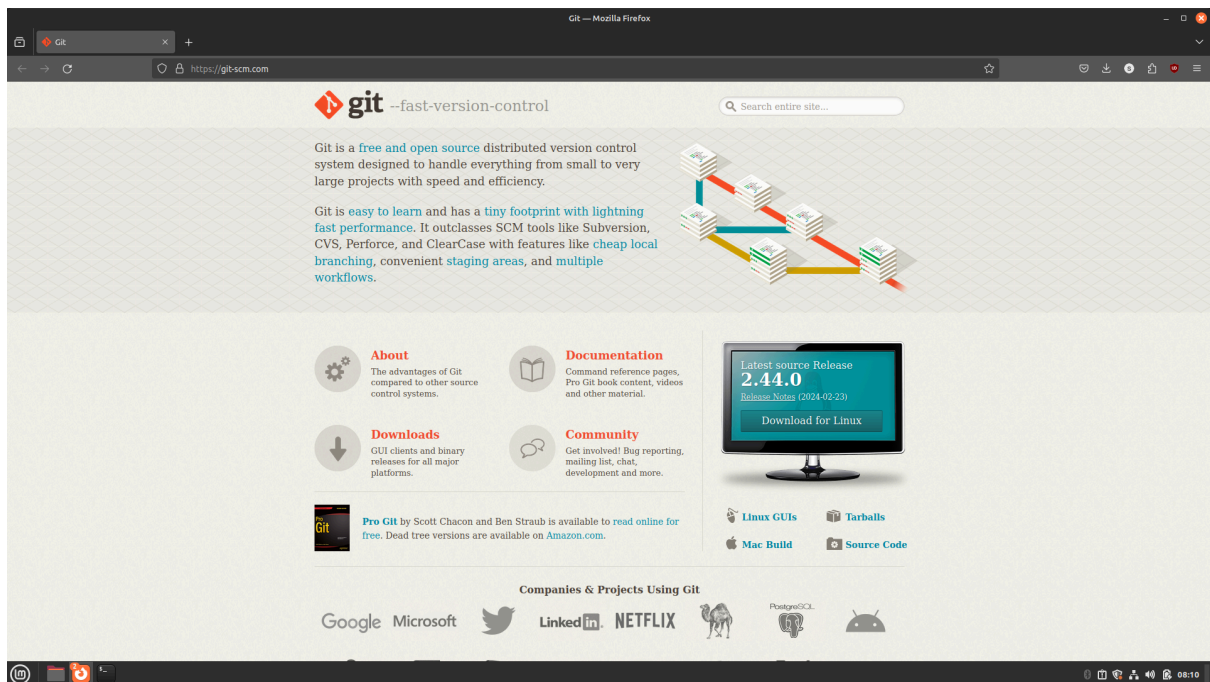
USER MANUAL

1. Go to the github repository:

<https://github.com/a-for-ball/student-grade-prediction>

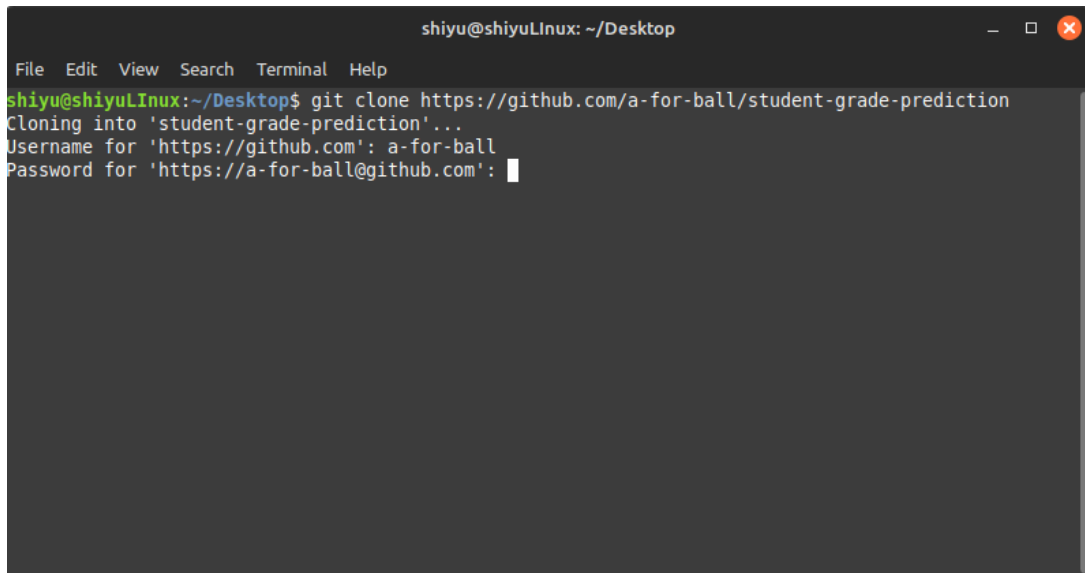


2. Ensure that Git is installed in the system. If not, download the package from <https://git-scm.com>



3. Clone the repository using the command “git clone <https://github.com/a-for-ball/student-grade-prediction>”
4. Enter your username and the passkey that is provided

[Screenshot of terminal]



```
shiyu@shiyuLinux: ~/Desktop
File Edit View Search Terminal Help
shiyu@shiyuLinux:~/Desktop$ git clone https://github.com/a-for-ball/student-grade-prediction
Cloning into 'student-grade-prediction'...
Username for 'https://github.com': a-for-ball
Password for 'https://a-for-ball@github.com':
```

5. After successfully cloning the repository, run the Jupyter Notebook named ‘survey_test.ipynb’
6. Under the ‘Cells’ tab, select the ‘Run all’ option to execute all the cells

