Introduction to Big Data Analytics

Term Project Report

Topic: Student Academic Performance Data Analysis

(Powerful translation by DeepL)

Yu-Lu Zhang	107AEA002
Zi-Jian Li	108AEA001
Xiang Wang	108AEA008

December 2020

Motivations:

Academic performance is very important to a student, but we don't know what the most important factors are that affect it. We analyze the dataset and develop models to predict student performance.

Summary:

We used this dataset(xAPI-Edu-Data) to analyze the data and visualize it to predict student achievement by analyzing the data and developing a model.

Research Steps:

Data pre-processing->data visualization->model building analysis->parameter tuning->prediction effect

Environment:

Python 3.7 + Jupyter Notebook

Python required packages：

pandas、sklearn、seaborn、matplotlib、numpy

Reference:

Students' Academic Performance Dataset (xAPI-Edu-Data)
https://www.kaggle.com/aljarah/xAPI-Edu-Data
Seaborn API Website
https://seaborn.pydata.org/api.html

Attributes:

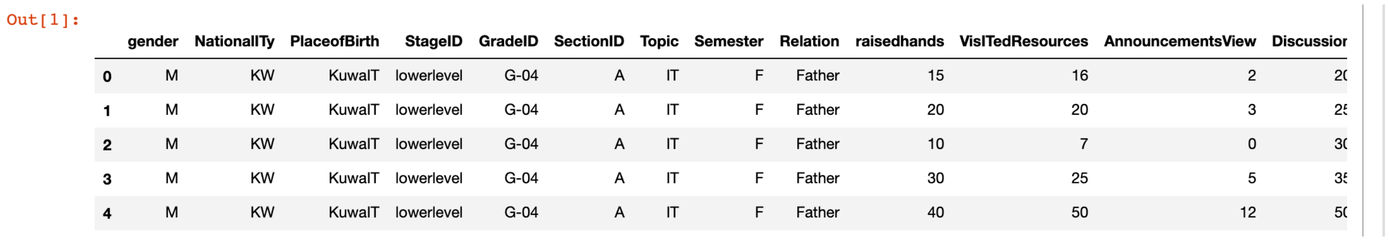
1 Gender - student's gender (nominal: 'Male' or 'Female’)
2 Nationality- student's nationality (nominal:’ Kuwait’,’ Lebanon’,’ Egypt’,’ SaudiArabia’,’ USA’,’ Jordan’,’
Venezuela’,’ Iran’,’ Tunis’,’ Morocco’,’ Syria’,’ Palestine’,’ Iraq’,’ Lybia’)
3 Place of birth- student's Place of birth (nominal:’ Kuwait’,’ Lebanon’,’ Egypt’,’ SaudiArabia’,’ USA’,’ Jordan’,’
Venezuela’,’ Iran’,’ Tunis’,’ Morocco’,’ Syria’,’ Palestine’,’ Iraq’,’ Lybia’)
4 Educational Stages- educational level student belongs (nominal: ‘lowerlevel’,’MiddleSchool’,’HighSchool’)
5 Grade Levels- grade student belongs (nominal: ‘G-01’, ‘G-02’, ‘G-03’, ‘G-04’, ‘G-05’, ‘G-06’, ‘G-07’, ‘G-08’, ‘G-09’, ‘G-10’, ‘G-11’, ‘G-12 ‘)
6 Section ID- classroom student belongs (nominal:’A’,’B’,’C’)

7 Topic- course topic (nominal:’ English’,’ Spanish’, ‘French’,’ Arabic’,’ IT’,’ Math’,’ Chemistry’, ‘Biology’, ‘Science’,’ History’,’ Quran’,’ Geology’)
8 Semester- school year semester (nominal:’ First’,’ Second’)
9 Parent responsible for student (nominal:’mom’,’father’)
10 Raised hand- how many times the student raises his/her hand on classroom (numeric:0-100)
11- Visited resources- how many times the student visits a course content(numeric:0-100)
12 Viewing announcements-how many times the student checks the new announcements(numeric:0-100)
13 Discussion groups- how many times the student participate on discussion groups (numeric:0-100)
14 Parent Answering Survey- parent answered the surveys which are provided from school or not
(nominal:’Yes’,’No’)
15 Parent School Satisfaction- the Degree of parent satisfaction from school(nominal:’Yes’,’No’)
16 Student Absence Days-the number of absence days for each student (nominal: above-7, under-7)

Analysis Process:

1. Loading packages and dataset

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn import preprocessing, svm
from sklearn.linear_model import Perceptron
from sklearn.tree import DecisionTreeClassifier

data = pd.read_csv('xAPI-Edu-Data.csv')
data.head()
```



Out[1]:

	gender	NationalITy	PlaceofBirth	StageID	GradeID	SectionID	Topic	Semester	Relation	raisedhands	VisITedResources	AnnouncementsView	Discussior
0	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	15	16	2	20
1	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	20	20	3	25
2	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	10	7	0	30
3	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	30	25	5	35
4	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	40	50	12	50

View Data

2. Data Preprocessing

2.1 View Data Sheet

Input**:** data.info()

Output**:**

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 480 entries, 0 to 479
Data columns (total 17 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   gender                    480 non-null    object
 1   NationalITy               480 non-null    object
 2   PlaceofBirth              480 non-null    object
 3   StageID                   480 non-null    object
 4   GradeID                   480 non-null    object
 5   SectionID                 480 non-null    object
 6   Topic                     480 non-null    object
 7   Semester                  480 non-null    object
 8   Relation                  480 non-null    object
 9   raisedhands               480 non-null    int64
 10  VisITedResources          480 non-null    int64
 11  AnnouncementsView         480 non-null    int64
 12  Discussion                480 non-null    int64
 13  ParentAnsweringSurvey     480 non-null    object
 14  ParentschoolSatisfaction  480 non-null    object
 15  StudentAbsenceDays        480 non-null    object
 16  Class                     480 non-null    object
```

According to the analysis in the above table, there are no null values and the data does not need to be pre-processed.

2.2 View student grades by category

Input: data.Class.unique()

Output: array(['M', 'L', 'H'], dtype=object)

Student grades are divided into three categories ['L', 'M', 'H'], which will be used as criteria for evaluating students.

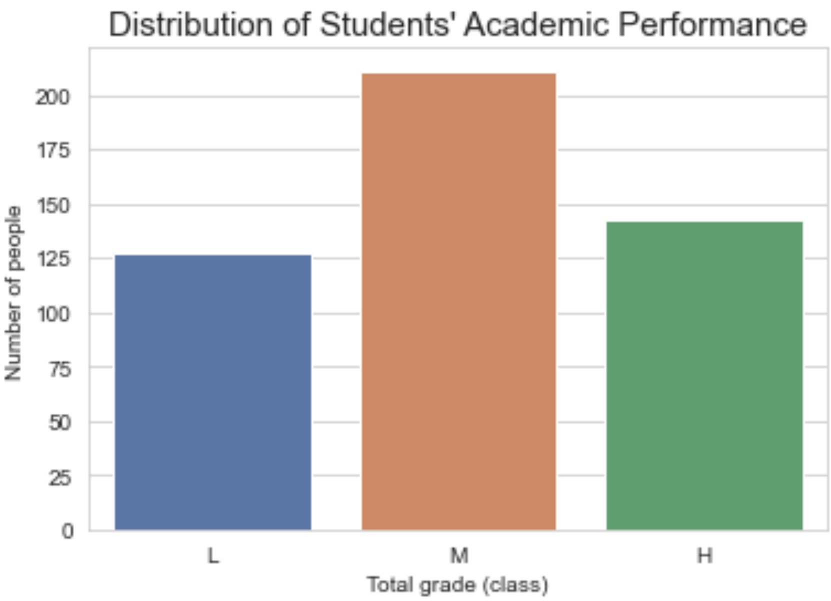
L:0-59 Failure.

M:60-89 Medium.

H:90-100 High scores.

3. Data Visualization

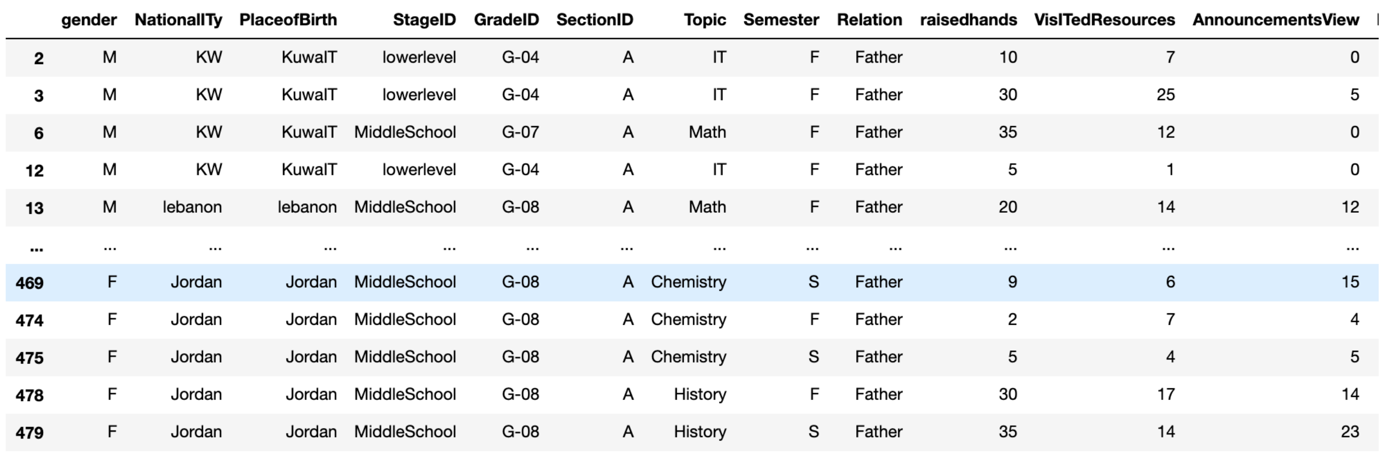
```
sns.set_style("whitegrid")
ax = sns.countplot(x='Class', data=data, order=['L', 'M', 'H'], palette="deep")
plt.xlabel('Total grade (class)')
plt.ylabel('Number of people')
plt.title("Distribution of Students' Academic Performance", size=15)
plt.show()
```



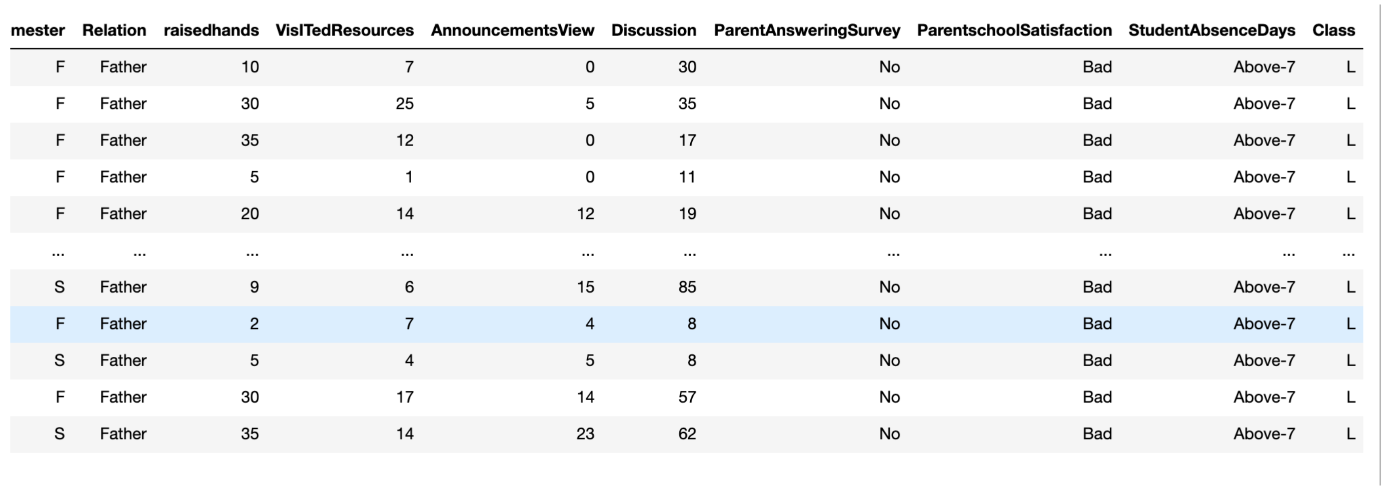
According to the chart above, most students were in the middle of the range, followed by high scores and the fewest number of failing grades.

3.1 View information for failing students

```
data.loc[data["Class"]=='L']
```



	gender	NationalITy	PlaceofBirth	StageID	GradeID	SectionID	Topic	Semester	Relation	raisedhands	VisITedResources	AnnouncementsView
2	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	10	7	0
3	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	30	25	5
6	M	KW	KuwaIT	MiddleSchool	G-07	A	Math	F	Father	35	12	0
12	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	5	1	0
13	M	lebanon	lebanon	MiddleSchool	G-08	A	Math	F	Father	20	14	12
...	...	...	...	...	...	...	...	...	...	...	...	...
469	F	Jordan	Jordan	MiddleSchool	G-08	A	Chemistry	S	Father	9	6	15
474	F	Jordan	Jordan	MiddleSchool	G-08	A	Chemistry	F	Father	2	7	4
475	F	Jordan	Jordan	MiddleSchool	G-08	A	Chemistry	S	Father	5	4	5
478	F	Jordan	Jordan	MiddleSchool	G-08	A	History	F	Father	30	17	14
479	F	Jordan	Jordan	MiddleSchool	G-08	A	History	S	Father	35	14	23



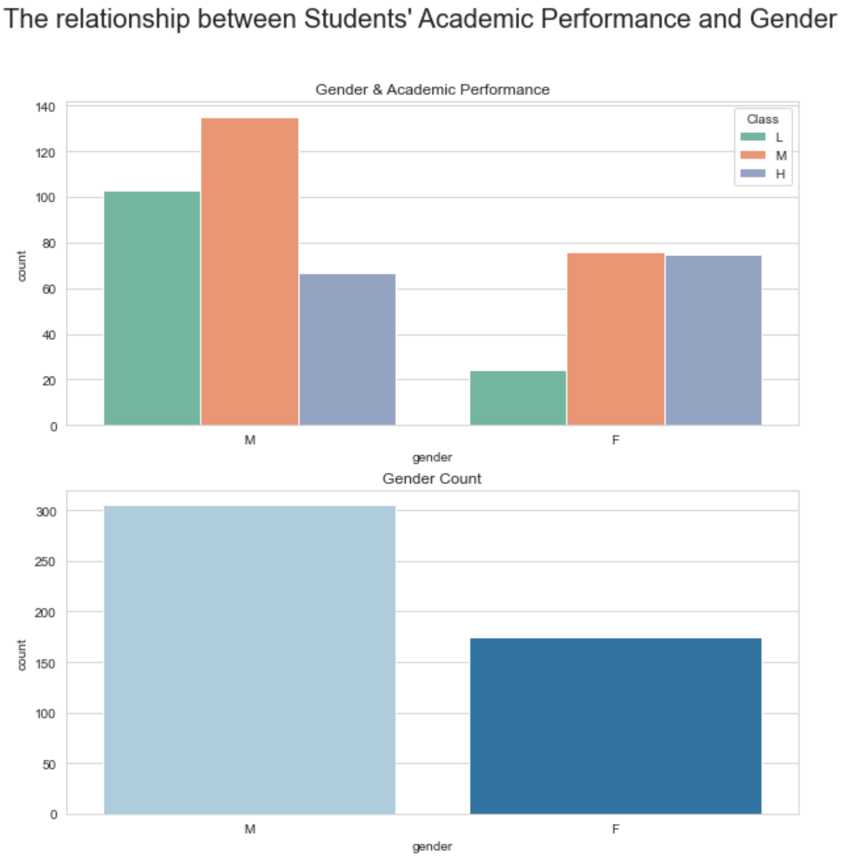
mester	Relation	raisedhands	VisITedResources	AnnouncementsView	Discussion	ParentAnsweringSurvey	ParentschoolSatisfaction	StudentAbsenceDays	Class
F	Father	10	7	0	30	No	Bad	Above-7	L
F	Father	30	25	5	35	No	Bad	Above-7	L
F	Father	35	12	0	17	No	Bad	Above-7	L
F	Father	5	1	0	11	No	Bad	Above-7	L
F	Father	20	14	12	19	No	Bad	Above-7	L
...	...	...	...	...	...	...	...	...	...
S	Father	9	6	15	85	No	Bad	Above-7	L
F	Father	2	7	4	8	No	Bad	Above-7	L
S	Father	5	4	5	8	No	Bad	Above-7	L
F	Father	30	17	14	57	No	Bad	Above-7	L
S	Father	35	14	23	62	No	Bad	Above-7	L

Based on the table above, it appears that the failing students missed more than seven days of school, and all values were concentrated in a low area, such as those who raised their hands less often and did not participate in discussions (see the graph below).

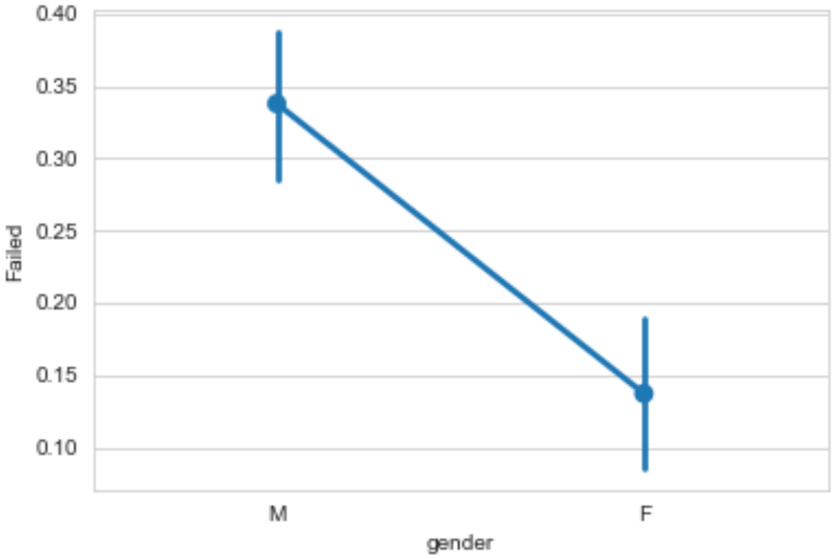
We can also observe that the guardians of the failing students were usually the fathers, and they did not take the school survey and were not very satisfied with the school.

3.2 The relationship between Students' Academic Performance and Gender

```
fig, axarr  = plt.subplots(2,figsize=(10,10))
sns.countplot(x='gender', hue='Class', data=data, order=['M', 'F'],hue_order = ['L', 'M', 'H'], ax=axarr[0], palette="Set2")
sns.countplot(x='gender', data=data, order=['M','F'], ax=axarr[1], palette="Paired")

axarr[0].set_title('Gender & Academic Performance')
axarr[1].set_title('Gender Count')
fig.suptitle("The relationship between Students' Academic Performance and Gender", size=20)
plt.show()
```



```
sns.pointplot(x='gender', y='Failed', data=data)
```

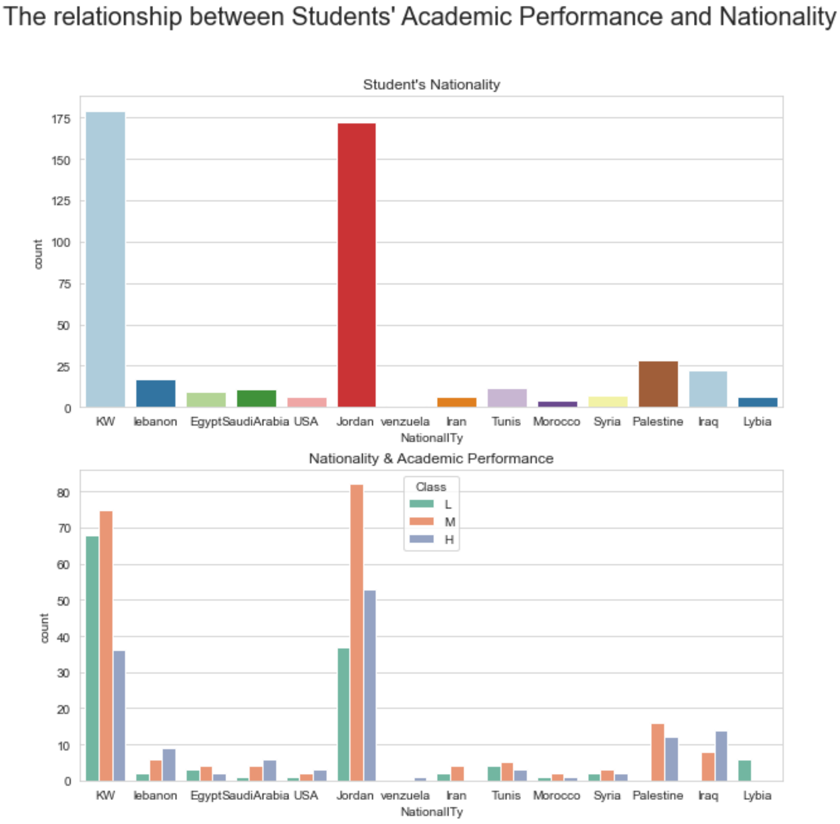


Based on this analysis, we can clearly see that the number of female students failing is much smaller than that of male students, and that the number of female students in the middle and high grades is almost equal. The difference in the number of male and female students at the high end of the scale is not large. From this we can conclude that gender may affect students' performance.

3.3The relationship between Students' Academic Performance and Nationality

```
fig, axarr  = plt.subplots(2,figsize=(10,10))
```

```
axarr[0].set_title("Student's Nationality")
axarr[1].set_title('Nationality & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and Nationality", size=20)
sns.countplot(x='NationalITy', data=data, ax=axarr[0], palette="Paired")
sns.countplot(x='NationalITy', hue='Class', data=data,hue_order = ['L', 'M', 'H'], ax=axarr[1], palette="Set2")
plt.show()
```



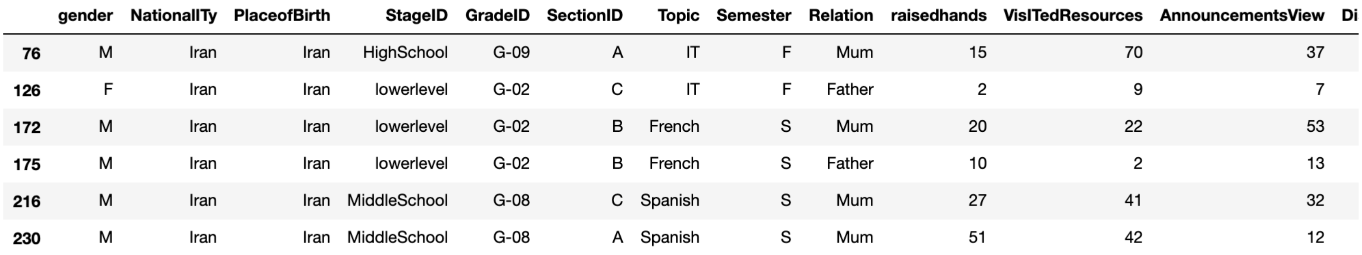
Based on this graph, except for the KW and Jordan students, the sample size for the other nationalities is relatively small and no valid information can be deduced from this analysis.

What we can deduce is that Jordan's students failed half as many as KW's students and had better overall grades than KW's students, while Iranian and Lybia's students did not get high grades.

Let's look at a sample of the Iranian and Lybia students and try to analyze the factors that affect their academic performance.

To see the samples of Iranian students:

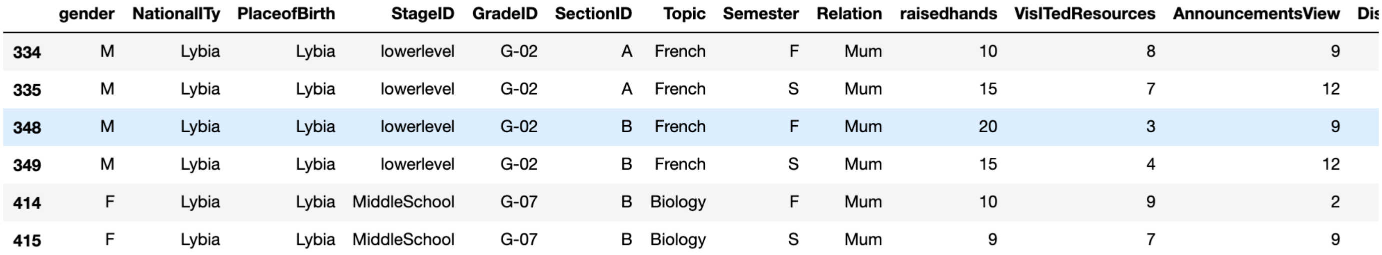
```
data.loc[data['NationalITy'] == 'Iran']
```



	gender	NationalITy	PlaceofBirth	StageID	GradeID	SectionID	Topic	Semester	Relation	raisedhands	VisITedResources	AnnouncementsView	Di
76	M	Iran	Iran	HighSchool	G-09	A	IT	F	Mum	15	70	37	
126	F	Iran	Iran	lowerlevel	G-02	C	IT	F	Father	2	9	7	
172	M	Iran	Iran	lowerlevel	G-02	B	French	S	Mum	20	22	53	
175	M	Iran	Iran	lowerlevel	G-02	B	French	S	Father	10	2	13	
216	M	Iran	Iran	MiddleSchool	G-08	C	Spanish	S	Mum	27	41	32	
230	M	Iran	Iran	MiddleSchool	G-08	A	Spanish	S	Mum	51	42	12	

To see the samples of Lybia students:

```
data.loc[data['NationalITy'] == 'Lybia']
```

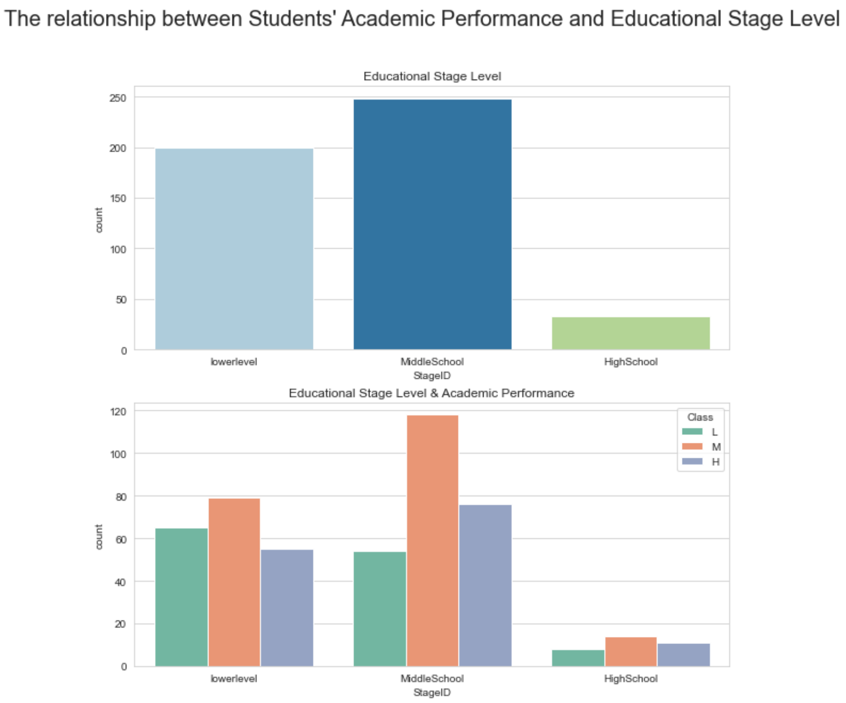


	gender	NationalITy	PlaceofBirth	StageID	GradeID	SectionID	Topic	Semester	Relation	raisedhands	VisITedResources	AnnouncementsView	Dis
334	M	Lybia	Lybia	lowerlevel	G-02	A	French	F	Mum	10	8	9	
335	M	Lybia	Lybia	lowerlevel	G-02	A	French	S	Mum	15	7	12	
348	M	Lybia	Lybia	lowerlevel	G-02	B	French	F	Mum	20	3	9	
349	M	Lybia	Lybia	lowerlevel	G-02	B	French	S	Mum	15	4	12	
414	F	Lybia	Lybia	MiddleSchool	G-07	B	Biology	F	Mum	10	9	2	
415	F	Lybia	Lybia	MiddleSchool	G-07	B	Biology	S	Mum	9	7	9	

From our observations, we know that there seems to be a high degree of overlap between the data for Lybia students and all students who did not pass the exam (missing more than 7 days, low values, no school survey, etc.).

3.4The relationship between Students' Academic Performance and Educational Stage Level

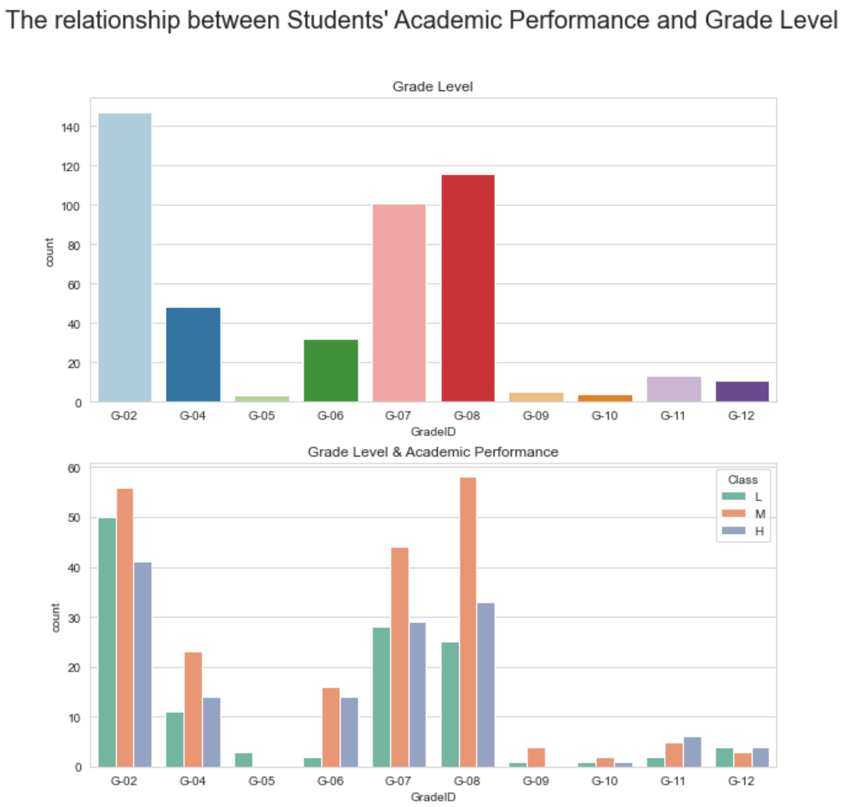
```
fig, axarr  = plt.subplots(2,figsize=(10,10))
axarr[0].set_title('Educational Stage Level')
axarr[1].set_title('Educational Stage Level & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and Education
al Stage Level", size=20)
sns.countplot(x='StageID', data=data, ax=axarr[0], palette="Paired")
sns.countplot(x='StageID', hue='Class', data=data, hue_order = ['L', 'M', 'H'], ax=
axarr[1], palette="Set2")
plt.show()
```



Based on this graph, we can see that students are over-represented in the middle grades regardless of their educational level.

3.5 The relationship between Students' Academic Performance and Grade Level

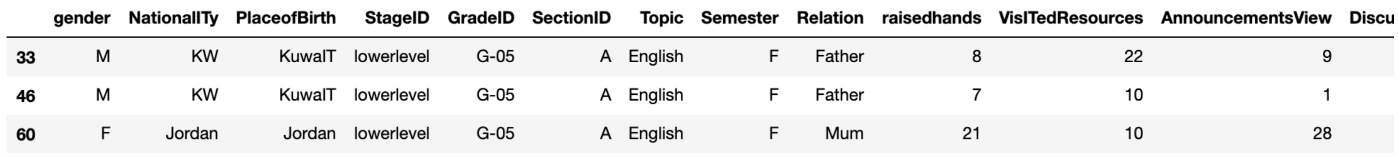
```
fig, axarr  = plt.subplots(2,figsize=(10,10))
axarr[0].set_title('Grade Level')
axarr[1].set_title('Grade Level & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and Grade Level", size=20)
sns.countplot(x='GradeID',
              data=data,
              order=['G-02', 'G-04', 'G-05', 'G-06', 'G-07', 'G-08', 'G-09', 'G-10', 'G-11', 'G-12'],
              ax=axarr[0], palette="Paired")
sns.countplot(x='GradeID',
              hue='Class',
              data=data,
              order=['G-02', 'G-04', 'G-05', 'G-06', 'G-07', 'G-08', 'G-09', 'G-10', 'G-11', 'G-12'],
              hue_order = ['L', 'M', 'H'],
              ax=axarr[1], palette="Set2")
plt.show()
```



Based on this analysis, we can see that the number of students in grades 5, 9, and 10 is very small. In addition, no fifth-grader passed and no ninth-grader scored high marks.

To see the samples of fifth-grade students:

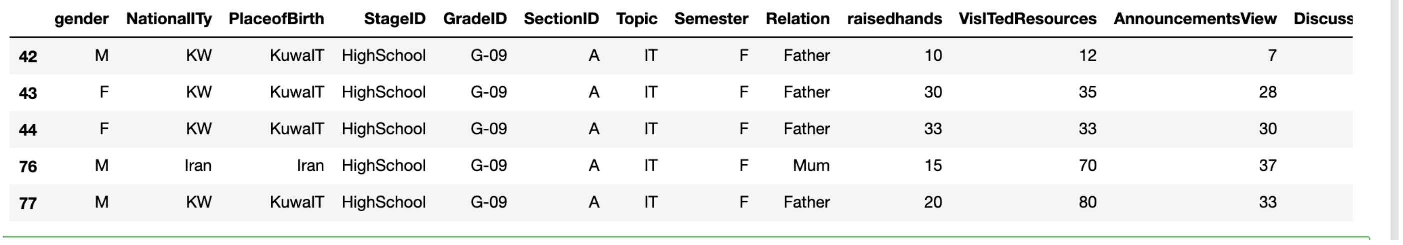
```
data.loc[data['GradeID']=='G-05']
```



	gender	NationalITy	PlaceofBirth	StageID	GradeID	SectionID	Topic	Semester	Relation	raisedhands	VisITedResources	AnnouncementsView	Discu
33	M	KW	KuwaIT	lowerlevel	G-05	A	English	F	Father	8	22	9	
46	M	KW	KuwaIT	lowerlevel	G-05	A	English	F	Father	7	10	1	
60	F	Jordan	Jordan	lowerlevel	G-05	A	English	F	Mum	21	10	28	

To see the samples of ninth-grade students:

```
data.loc[data['GradeID']=='G-09']
```



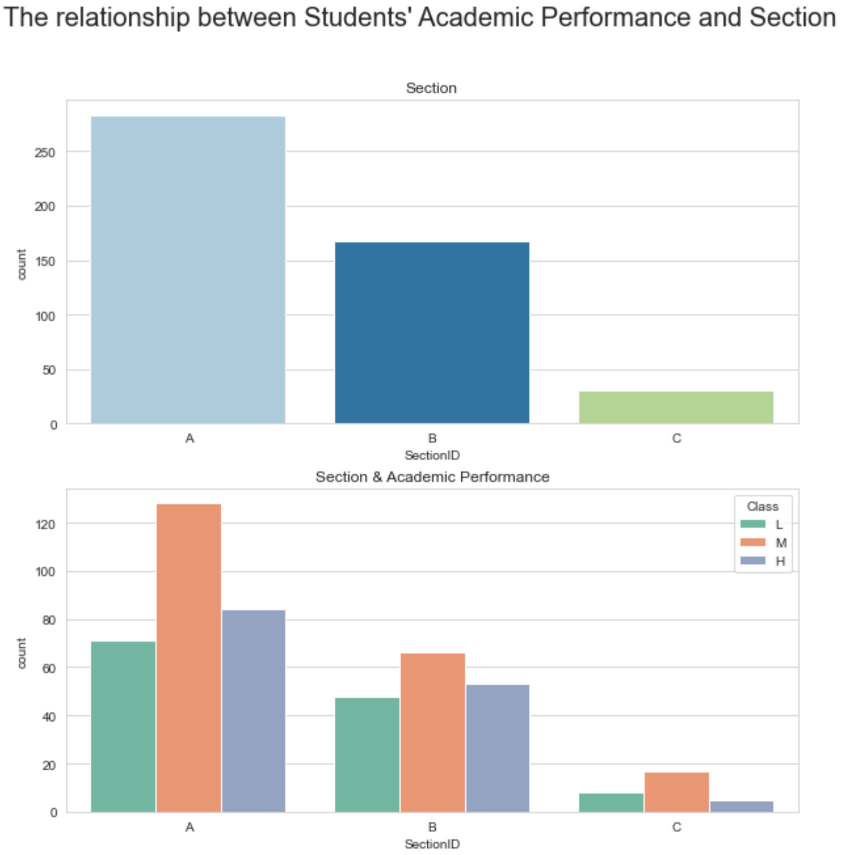
	gender	NationalITy	PlaceofBirth	StageID	GradeID	SectionID	Topic	Semester	Relation	raisedhands	VisITedResources	AnnouncementsView	Discuss
42	M	KW	KuwaIT	HighSchool	G-09	A	IT	F	Father	10	12	7	
43	F	KW	KuwaIT	HighSchool	G-09	A	IT	F	Father	30	35	28	
44	F	KW	KuwaIT	HighSchool	G-09	A	IT	F	Father	33	33	30	
76	M	Iran	Iran	HighSchool	G-09	A	IT	F	Mum	15	70	37	
77	M	KW	KuwaIT	HighSchool	G-09	A	IT	F	Father	20	80	33	

After observation, there seems to be a high degree of overlap between the data for 5th and 9th graders and all students who did not pass the test (missed more than 7 days, low values, no school survey, etc.).

3.6 The relationship between Students' Academic Performance and Section

```
fig, axarr  = plt.subplots(2,figsize=(10,10))
axarr[0].set_title('Section')
```

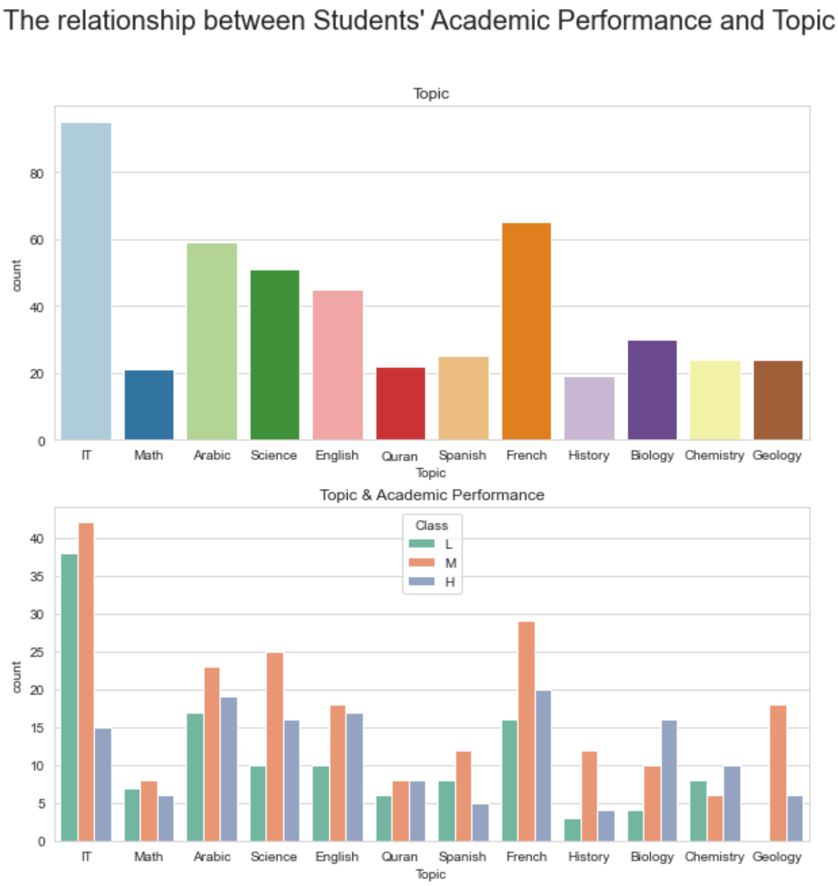
```
axarr[1].set_title('Section & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and Section", size=20)
sns.countplot(x='SectionID', data=data,
              order=['A', 'B', 'C'], ax = axarr[0], palette="Paired")
sns.countplot(x='SectionID', hue='Class',
              data=data, order=['A', 'B', 'C'],
              hue_order = ['L', 'M', 'H'], ax = axarr[1], palette="Set2")
plt.show()
```



Based on this graphical analysis, we know that the overall trend for all three classes is similar, and we cannot derive any valid information.

3.7 The relationship between Students' Academic Performance and Topic

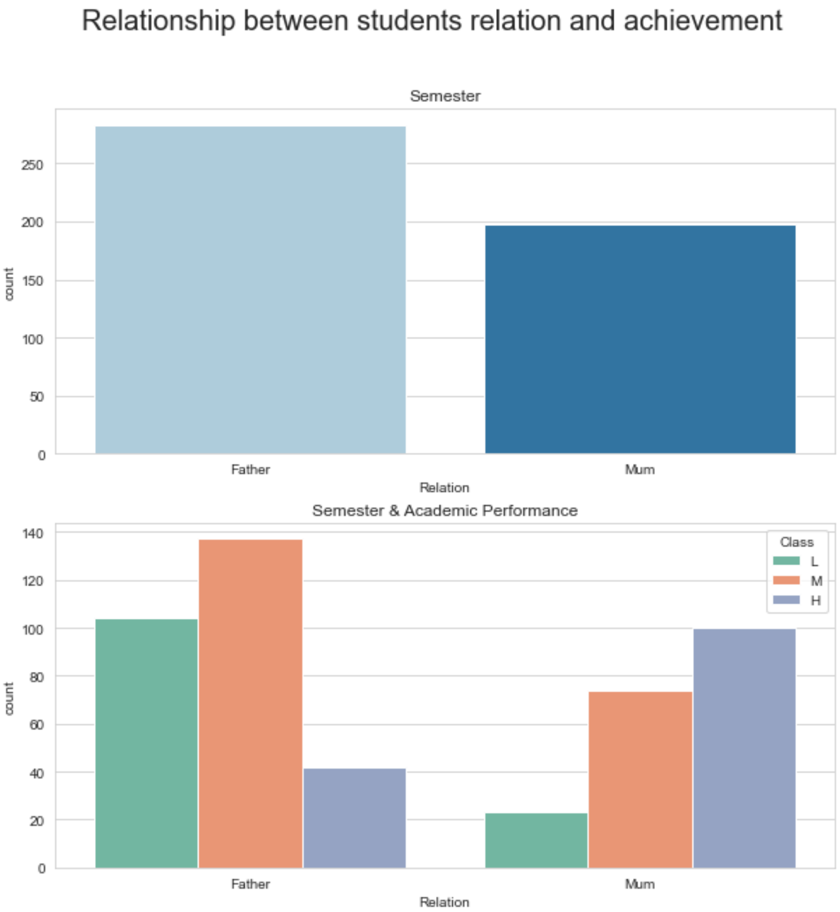
```
fig, axarr  = plt.subplots(2,figsize=(10,10))
axarr[0].set_title('Topic')
axarr[1].set_title('Topic & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and Topic", size=20)
sns.countplot(x='Topic', data=data, ax = axarr[0], palette="Paired")
sns.countplot(x='Topic', hue='Class', data=data,hue_order = ['L', 'M', 'H'], ax = axarr[1], palette="Set2")
plt.show()
```



Based on this graph analysis, we can see an interesting phenomenon that there are no failing students in Geology. Why is that?

3.8 The relationship between Students' Academic Performance and Semester

```
fig, axarr  = plt.subplots(2,figsize=(10,10))
axarr[0].set_title('Semester')
axarr[1].set_title('Semester & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and Semester", size=20)
sns.countplot(x='Semester', data=data, ax = axarr[0], palette="Paired")
sns.countplot(x='Semester', hue='Class', data=data,hue_order = ['L', 'M', 'H'], ax = axarr[1], palette="Set2")
plt.show()
```

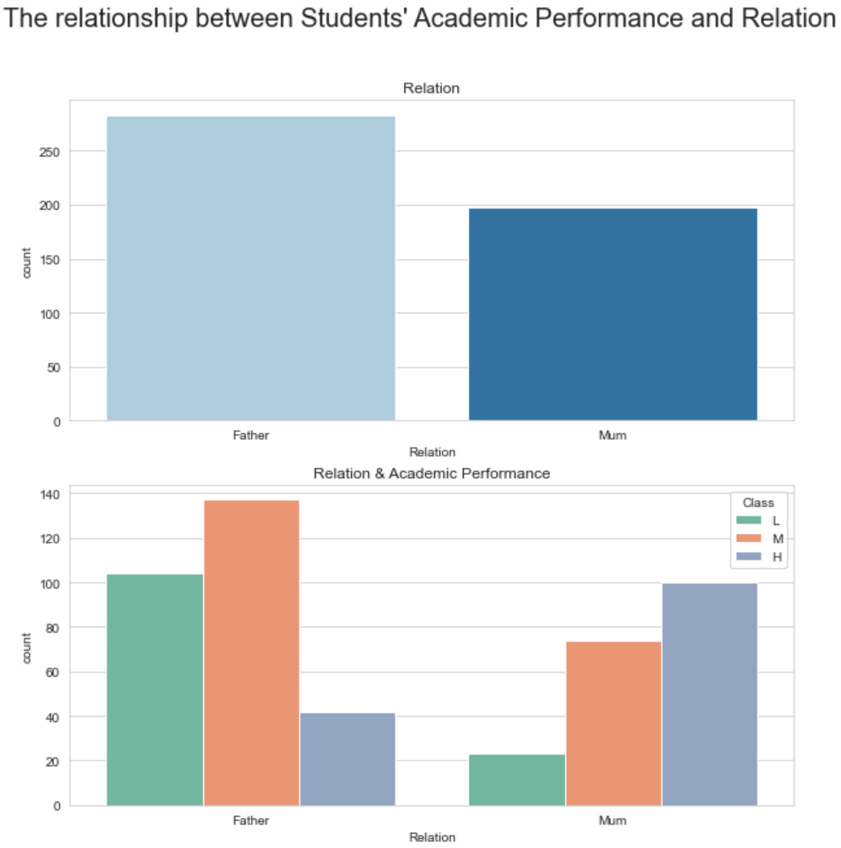


According to this graph, there were fewer failures in the second year than in the first year, and more high scorers than in the first year.

From this, we can conclude that the academic year may affect the grade.

3.9 The relationship between Students' Academic Performance and Relation

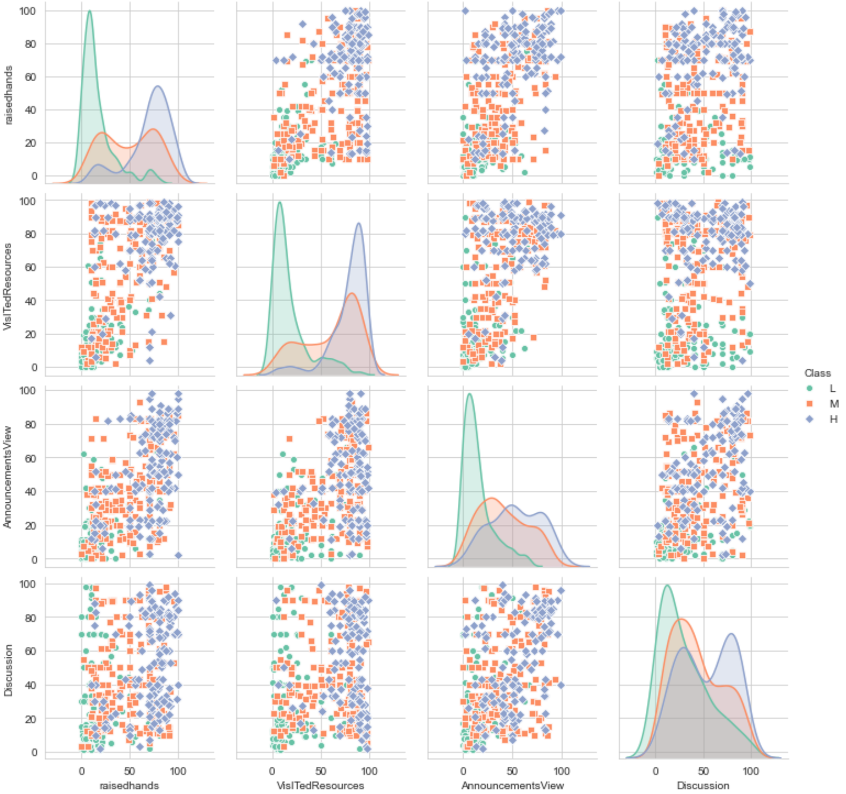
```python
fig, axarr  = plt.subplots(2,figsize=(10,10))
axarr[0].set_title('Relation')
axarr[1].set_title('Relation & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and Relation", size=20)
sns.countplot(x='Relation', data=data, ax = axarr[0], palette="Paired")
sns.countplot(x='Relation', hue='Class', data=data,hue_order = ['L', 'M', 'H'], ax = axarr[1], palette="Set2")
plt.show()
```



According to the analysis of the two figures above, there seems to be a correlation between the mother as guardian and the student passing, and a correlation between the father as guardian and the student failing.

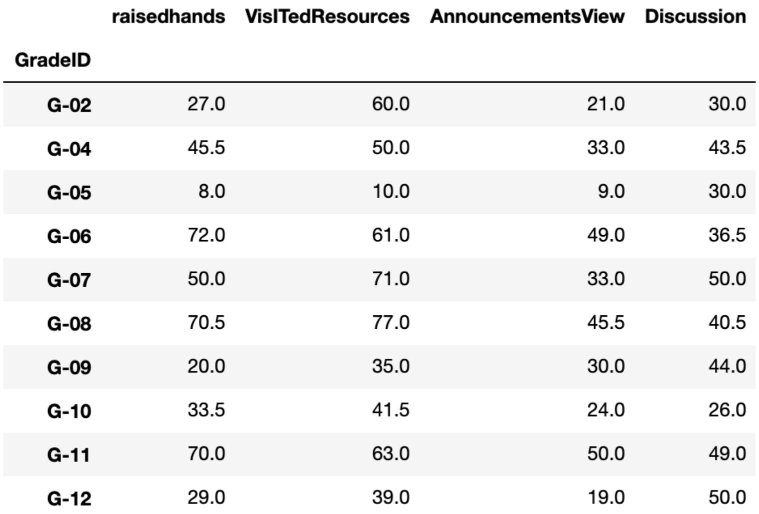
3.10 The relationship between the number of times students raised their hands in class, the number of times they visited course content, the number of times they checked new announcements, the number of times they participated in discussions, and their grades

```
sns.pairplot(data, hue="Class",
                 diag_kind="kde",
                 hue_order = ['L', 'M', 'H'],
                 markers=["o", "s", "D"], palette="Set2")
plt.show()
```



View raisedhands, VisiTedResources, AnnouncementsView, Discussion at different educational levels, and get the median here:

```
data.groupby('GradeID').median()
```



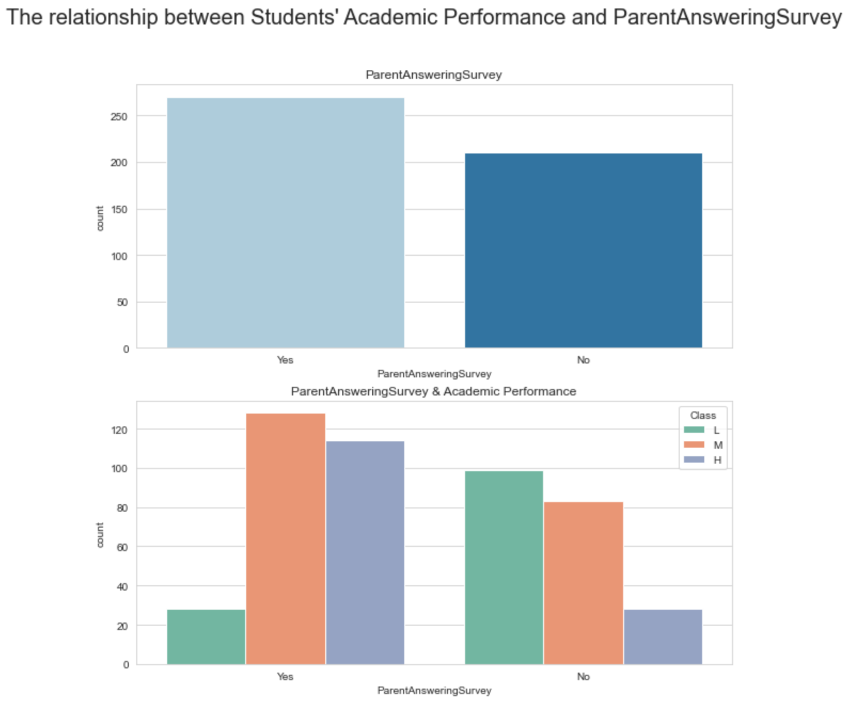
	raisedhands	VisITedResources	AnnouncementsView	Discussion
GradeID				
G-02	27.0	60.0	21.0	30.0
G-04	45.5	50.0	33.0	43.5
G-05	8.0	10.0	9.0	30.0
G-06	72.0	61.0	49.0	36.5
G-07	50.0	71.0	33.0	50.0
G-08	70.5	77.0	45.5	40.5
G-09	20.0	35.0	30.0	44.0
G-10	33.5	41.5	24.0	26.0
G-11	70.0	63.0	50.0	49.0
G-12	29.0	39.0	19.0	50.0

Here we can see that the data for grades 5 and 9 are much lower than most other grades.

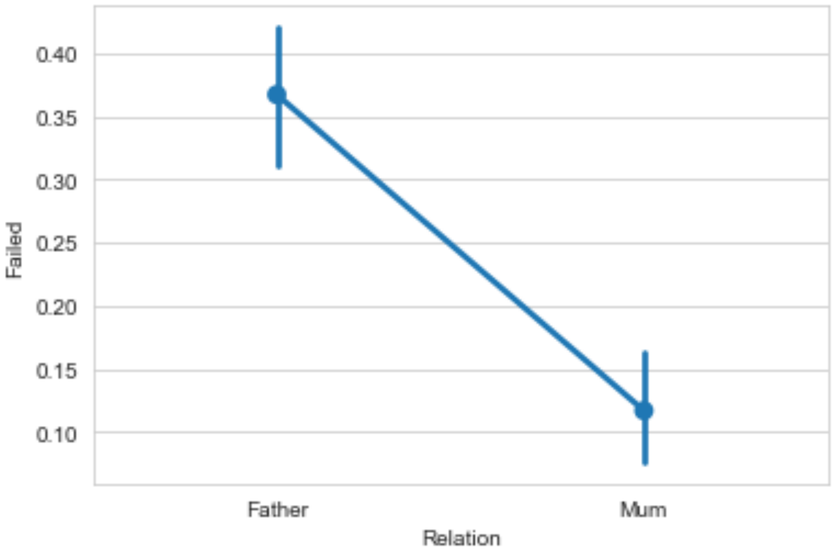
3.11 The relationship between Students' Academic Performance and ParentAnsweringSurvey

```
fig, axarr  = plt.subplots(2,figsize=(10,10))
axarr[0].set_title('ParentAnsweringSurvey')
axarr[1].set_title('ParentAnsweringSurvey & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and ParentAnsweringSurvey", size=20)
```

```
sns.countplot(x='ParentAnsweringSurvey', data=data,
              order=['Yes', 'No'], ax = axarr[0], palette="Paired")
sns.countplot(x='ParentAnsweringSurvey', hue='Class',
              data=data, order=['Yes', 'No'], hue_order = ['L', 'M', 'H'],
              ax = axarr[1], palette="Set2")
plt.show()
```



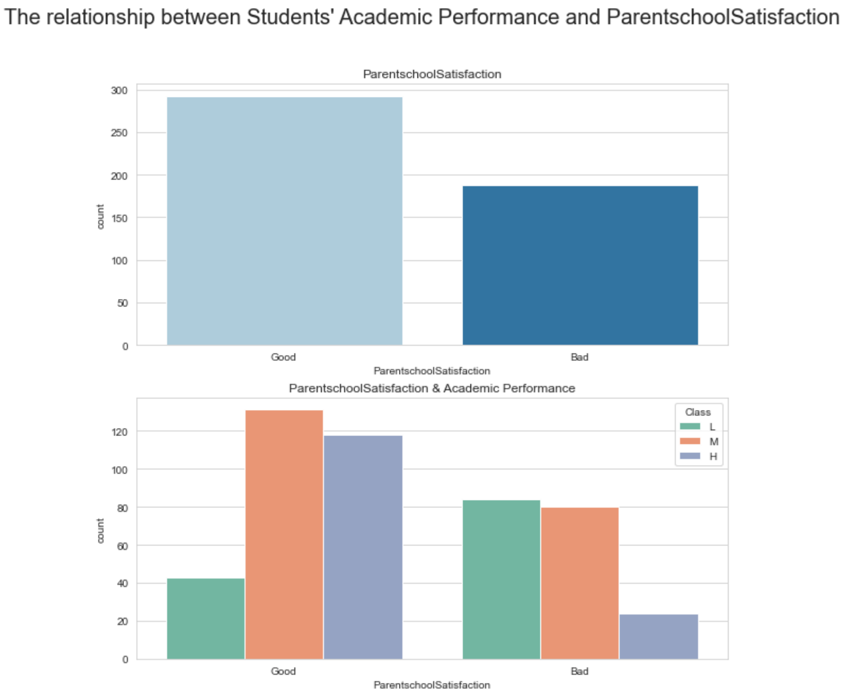
```
sns.pointplot(x='Relation', y='Failed', data=data)
```



Among high-achieving students, the vast majority of parents were satisfied with the education they received. Students whose parents were least satisfied with the school fared much worse. Students whose mothers were responsible for them fared better.

3.12The relationship between Students' Academic Performance and ParentschoolSatisfaction

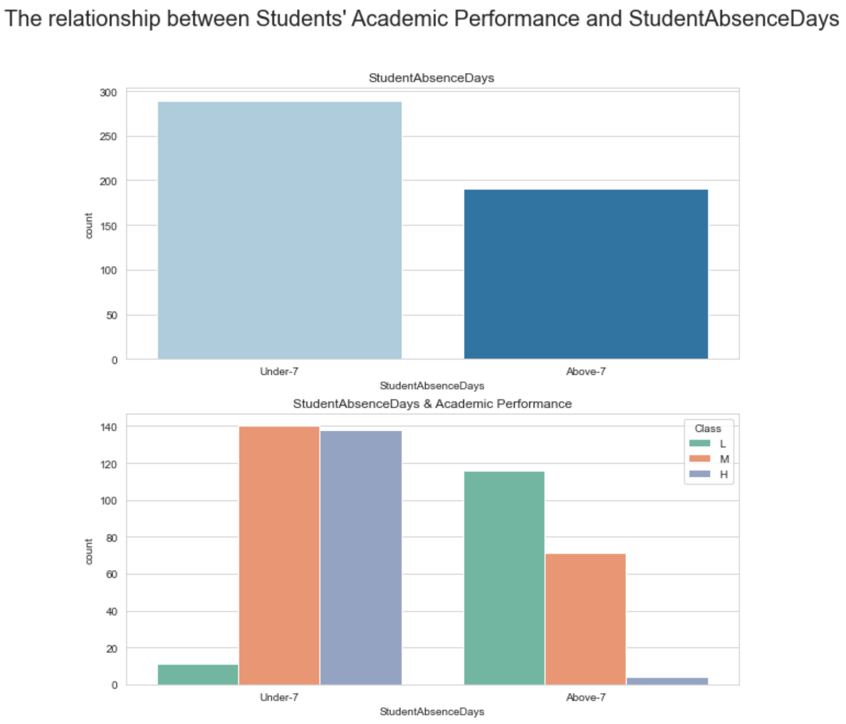
```python
fig, axarr  = plt.subplots(2,figsize=(10,10))
axarr[0].set_title('ParentschoolSatisfaction')
axarr[1].set_title('ParentschoolSatisfaction & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and ParentschoolSatisfaction", size=20)
sns.countplot(x='ParentschoolSatisfaction', data=data,
              order=['Good', 'Bad'], ax = axarr[0], palette="Paired")
sns.countplot(x='ParentschoolSatisfaction', hue='Class',
              data=data, order=['Good', 'Bad'],
              hue_order = ['L', 'M', 'H'], ax = axarr[1], palette="Set2")
plt.show()
```



The same observations as in 3.11 are not described.

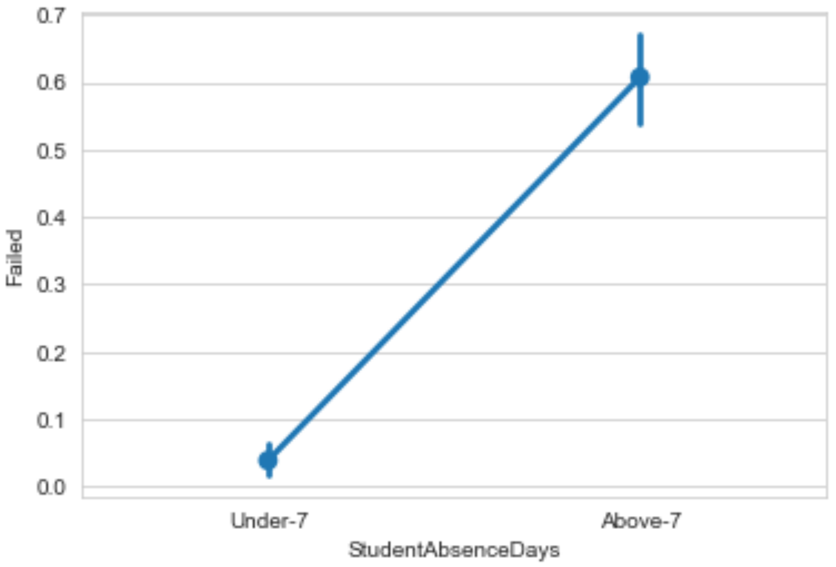
3.13 The relationship between Students' Academic Performance and StudentAbsenceDays

```python
fig, axarr  = plt.subplots(2,figsize=(10,10))
axarr[0].set_title('StudentAbsenceDays')
axarr[1].set_title('StudentAbsenceDays & Academic Performance')
fig.suptitle("The relationship between Students' Academic Performance and StudentAbsenceDays", size=20)
sns.countplot(x='StudentAbsenceDays', data=data,
              order=['Under-7', 'Above-7'],
              ax = axarr[0], palette="Paired")
sns.countplot(x='StudentAbsenceDays', hue='Class',
              data=data, order=['Under-7', 'Above-7'],
              hue_order = ['L', 'M', 'H'],
              ax = axarr[1], palette="Set2")
plt.show()
```



According to this graph, there is a strong correlation between study time and student performance. Students who have missed more than seven days of school rarely get a high grade, and students who have missed less than seven days of school rarely fail.

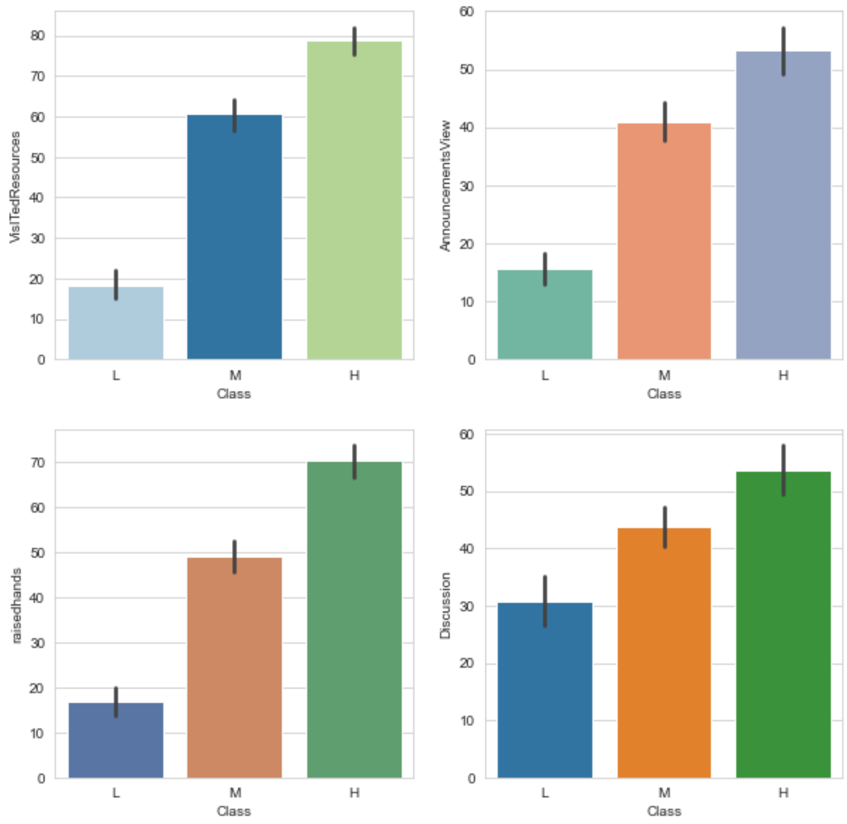
```
data['Failed'] = np.where(data['Class']=='L',1,0)
sns.pointplot(x='StudentAbsenceDays', y='Failed', data=data)
```



The largest intuitive trend can be seen in the frequency of student absences. Low-performing students had more than seven absences, while high-performing students hardly ever had more than seven absences.

3.14 A bar graph of the number of times students raised their hands in class, the number of times they visited course content, the number of times they checked new announcements, the number of discussions they attended, and their grades

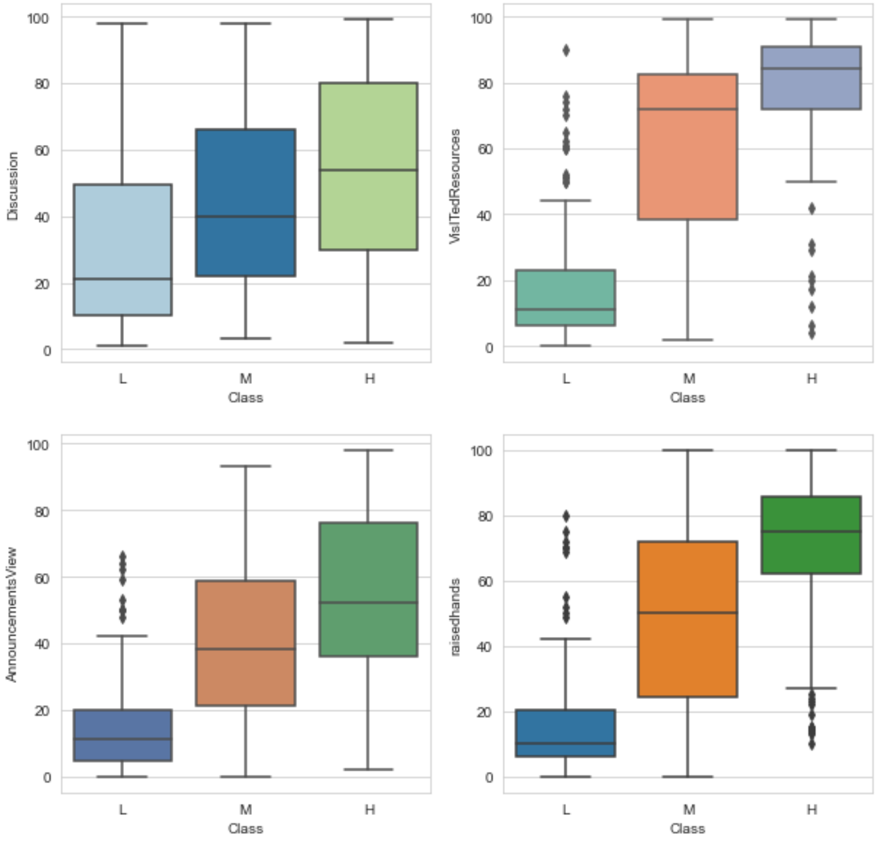
```
fig, axarr  = plt.subplots(2,2,figsize=(10,10))
sns.barplot(x='Class', y='VisITedResources', data=data, order=['L','M','H'], ax=axarr[0,0], palette="Paired")
sns.barplot(x='Class', y='AnnouncementsView', data=data, order=['L','M','H'], ax=axarr[0,1], palette="Set2")
sns.barplot(x='Class', y='raisedhands', data=data, order=['L','M','H'], ax=axarr[1,0], palette="deep")
sns.barplot(x='Class', y='Discussion', data=data, order=['L','M','H'], ax=axarr[1,1], palette="tab10")
```



As expected, those who participate more (more discussions, more hands, more bulletin views, more hands) perform better ...... This is the correlation and causality thing.

3.15 Classroom Activity Contrast

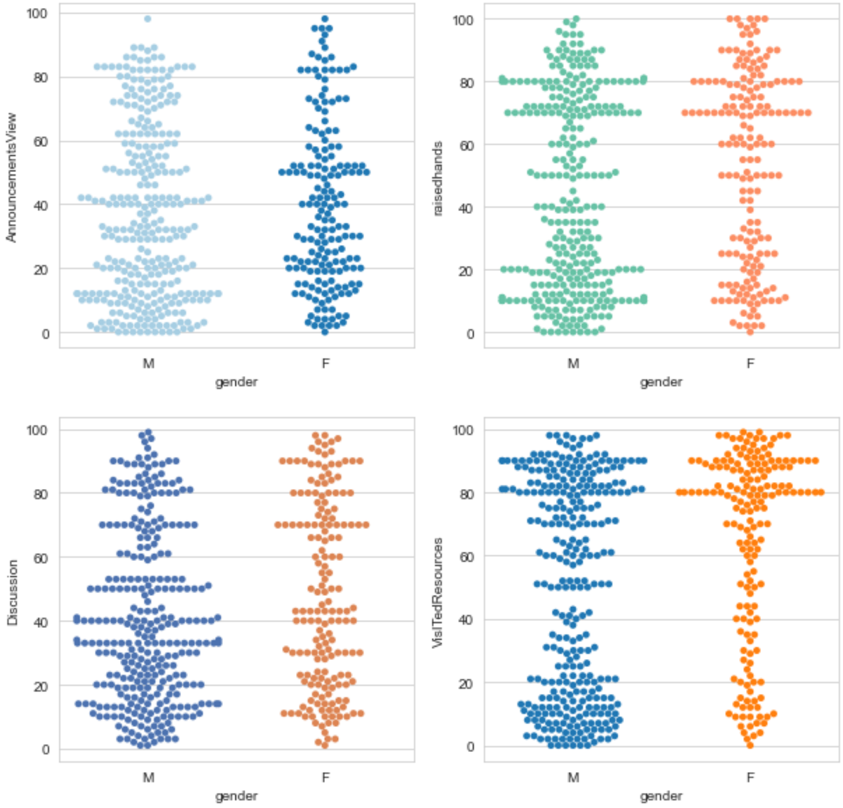
```
fig, axarr = plt.subplots(2, 2,figsize=(10,10))
sns.boxplot(x='Class', y='Discussion', data=data, order=['L','M','H'], ax=axarr[0,0], palette="Paired")
sns.boxplot(x='Class', y='VisITedResources', data=data, order=['L','M','H'], ax=axarr[0,1], palette="Set2")
sns.boxplot(x='Class', y='AnnouncementsView', data=data, order=['L','M','H'], ax=axarr[1,0], palette="deep")
sns.boxplot(x='Class', y='raisedhands', data=data, order=['L','M','H'], ax=axarr[1,1], palette="tab10")
```



According to the above analysis, access to course content may not be as necessary for good performance as discussion, and raising hands may not be as necessary for good performance as checking for new announcements.

3.16 Contrast of Gender and Classroom Participation

```
fig, axarr = plt.subplots(2, 2,figsize=(10,10))
sns.swarmplot(x='gender', y='AnnouncementsView', data=data, ax=axarr[0,0], palette="Paired")
sns.swarmplot(x='gender', y='raisedhands', data=data, ax=axarr[0,1], palette="Set2")
sns.swarmplot(x='gender', y='Discussion', data=data, ax=axarr[1,0], palette="deep")
sns.swarmplot(x='gender', y='VisITedResources', data=data, ax=axarr[1,1], palette="tab10")
```



This swarm diagram tells us that students with low scores (L) access a much hotter resource than students with M or H. In addition, women with high scores (H) access almost exclusively online resources. In addition, women with high scores (H) visited almost exclusively a large number of online resources.

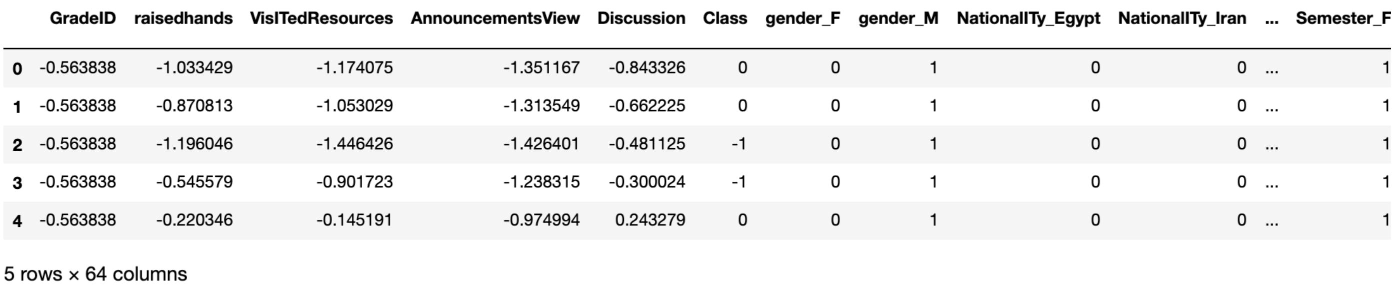
In summary, student grades were related to attributes such as number of visits to course content, days absent, number of hands raised in class, number of times checked for new announcements, participation in discussions, gender, guardian, and semester.

4. Modeling the relationship between predictions and student performance

4.1 Processing Data

```
# Convert grades into data
gradeID_dict = {"G-01" : 1,
                "G-02" : 2,
                "G-03" : 3,
                "G-04" : 4,
                "G-05" : 5,
                "G-06" : 6,
                "G-07" : 7,
                "G-08" : 8,
                "G-09" : 9,
                "G-10" : 10,
                "G-11" : 11,
                "G-12" : 12}

```

```python
data = data.replace({"GradeID" : gradeID_dict})
# Convert scores into data
class_dict = {"L" : -1,
                "M" : 0,
                "H" : 1}
data = data.replace({"Class" : class_dict})

# Convert to Scale data
data["GradeID"] = preprocessing.scale(data["GradeID"])
data["raisedhands"] = preprocessing.scale(data["raisedhands"])
data["VisITedResources"] = preprocessing.scale(data["VisITedResources"])
data["AnnouncementsView"] = preprocessing.scale(data["AnnouncementsView"])
data["Discussion"] = preprocessing.scale(data["Discussion"])

# Use virtual code conversion to convert 11 columns into 64 columns
data = pd.get_dummies(data, columns=["gender",
                                     "NationalITy",
                                     "PlaceofBirth",
                                     "SectionID",
                                     "StageID",
                                     "Topic",
                                     "Semester",
                                     "Relation",
                                     "ParentAnsweringSurvey",
                                     "ParentschoolSatisfaction",
                                     "StudentAbsenceDays"])


data.head()
```



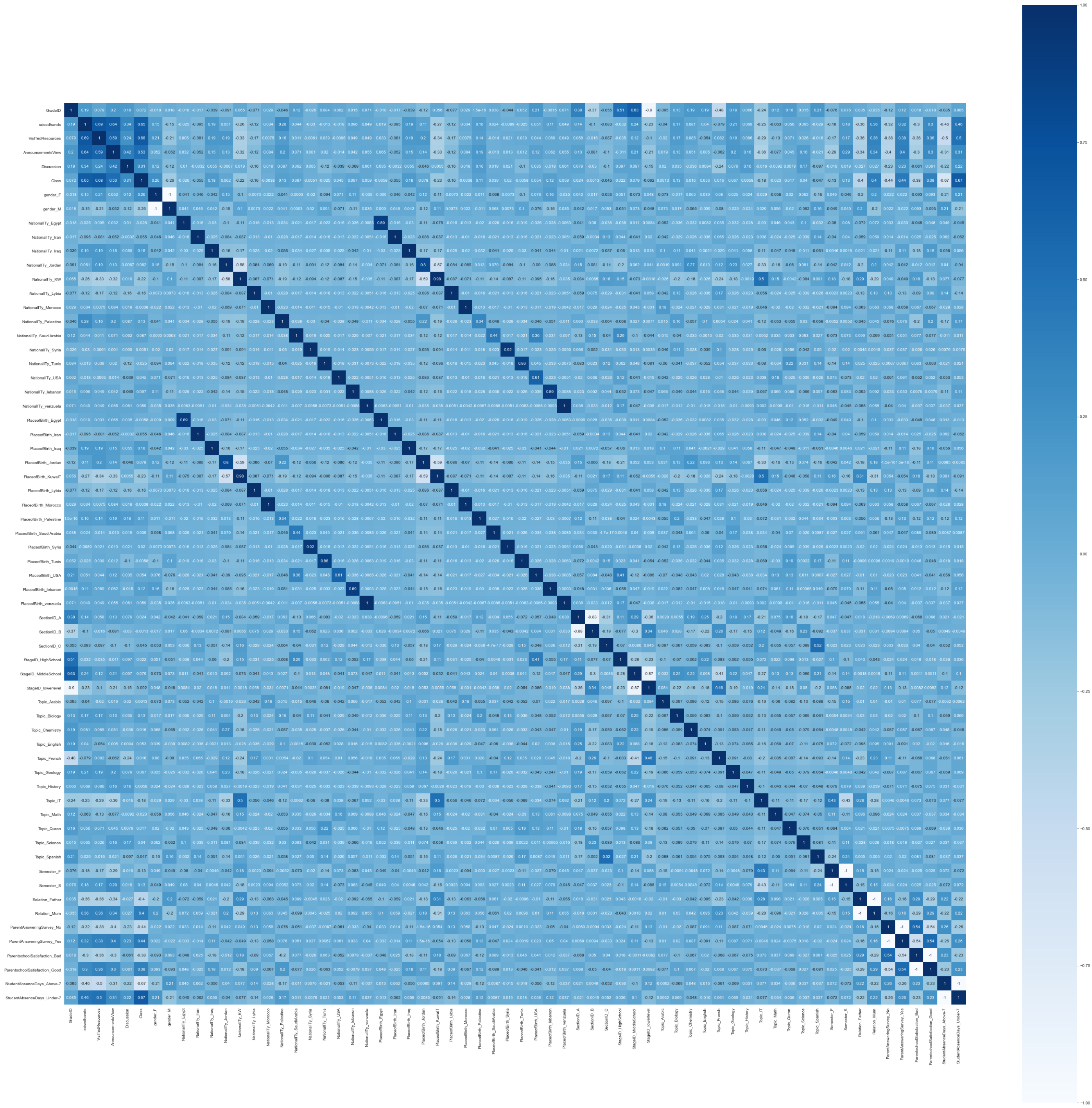
	GradeID	raisedhands	VisITedResources	AnnouncementsView	Discussion	Class	gender_F	gender_M	NationalITy_Egypt	NationalITy_Iran	...	Semester_F
0	-0.563838	-1.033429	-1.174075	-1.351167	-0.843326	0	0	1	0	0	...	1
1	-0.563838	-0.870813	-1.053029	-1.313549	-0.662225	0	0	1	0	0	...	1
2	-0.563838	-1.196046	-1.446426	-1.426401	-0.481125	-1	0	1	0	0	...	1
3	-0.563838	-0.545579	-0.901723	-1.238315	-0.300024	-1	0	1	0	0	...	1
4	-0.563838	-0.220346	-0.145191	-0.974994	0.243279	0	0	1	0	0	...	1

5 rows × 64 columns

4.2 List the correlation between grades and other attributes

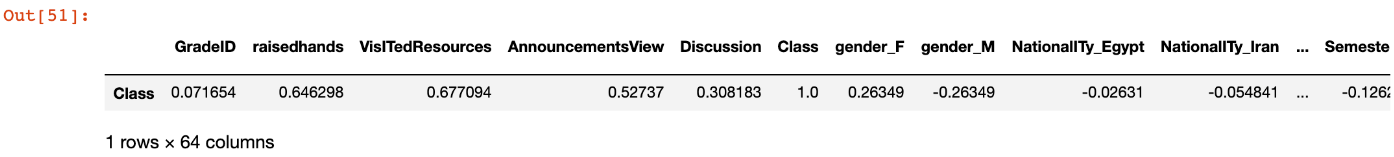
```python
corr = data.corr()
mask = np.triu(np.ones_like(corr, dtype=bool))
f, ax = plt.subplots(figsize=(11, 9))
```

```python
cmap = sns.diverging_palette(230, 20, as_cmap=True)
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
```



We look at Class in relation to the other columns individually:

```python
corr = data.corr()
corr.iloc[[5]]
```



Out[51]:

	GradeID	raisedhands	VisITedResources	AnnouncementsView	Discussion	Class	gender_F	gender_M	NationalITy_Egypt	NationalITy_Iran	...	Semeste
Class	0.071654	0.646298	0.677094	0.52737	0.308183	1.0	0.26349	-0.26349	-0.02631	-0.054841	...	-0.1262

1 rows × 64 columns

Based on the above graphs and tables, we can see that the number of visits to the course content, the number of days absent, the number of hands raised in class, the number of times new announcements were checked, whether or not discussions were held, gender, guardians, and semesters all have strong correlations with Class, as in our previous analysis.

5. Training and Forecasting

5.1 Find the most accurate classifier

```
X = data.drop('Class', axis=1)
y = data['Class']

# Encoding our categorical columns in X
labelEncoder = LabelEncoder()
cat_columns = X.dtypes.pipe(lambda x: x[x == 'object']).index
for col in cat_columns:
    X[col] = labelEncoder.fit_transform(X[col])

# Train Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=52)

# Create the radial basis function kernel version of a Support Vector Machine classifier
rbf_clf = svm.SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
              decision_function_shape='ovo', degree=3, gamma='auto', kernel='rbf',
              max_iter=-1, probability=False, random_state=None, shrinking=True,
              tol=0.001, verbose=False)
# Create the linear kernel version of a Support Vector Machine classifier
lin_clf = svm.SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
              decision_function_shape='ovo', degree=3, gamma='auto', kernel='linear',
              max_iter=-1, probability=False, random_state=None, shrinking=True,
              tol=0.001, verbose=False)
# Create the polynomial kernel version of a Support Vector Machine classifier
poly_clf = svm.SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
              decision_function_shape='ovo', degree=3, gamma='auto', kernel='poly',

              max_iter=-1, probability=False, random_state=None, shrinking=True,
              tol=0.001, verbose=False)
# Create the sigmoid kernel version of a Support Vector Machine classifier
sig_clf = svm.SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
              decision_function_shape='ovo', degree=3, gamma='auto', kernel='sigmoid',
              max_iter=-1, probability=False, random_state=None, shrinking=True,
              tol=0.001, verbose=False)
keys = []
scores = []
models = {'Logistic Regression': LogisticRegression(max_iter=3000), 'Decision Tree': DecisionTreeClassifier(),
         'Random Forest': RandomForestClassifier(n_estimators=300, random_state=52),'Perceptron':Perceptron(eta0=0.1, random_state=15),'RBF':rbf_clf,'Linear':lin_clf,
        'Polynomial':poly_clf,'Sigmoid':sig_clf}

for k,v in models.items():
    mod = v
    mod.fit(X_train, y_train)
    pred = mod.predict(X_test)
    print('Results for: ' + str(k) + '\n')
    print(confusion_matrix(y_test, pred))
    print(classification_report(y_test, pred))
    acc = accuracy_score(y_test, pred)
    print("accuracy is "+ str(acc))
    print('\n' + '\n')
    keys.append(k)
    scores.append(acc)
    table = pd.DataFrame({'model':keys, 'accuracy score':scores})

print(table)
```

Output:

```
Results for: Logistic Regression

[[28  7  1]
 [ 4 43  9]
 [ 0 12 40]]
              precision    recall  f1-score   support

          -1       0.88      0.78      0.82        36
           0       0.69      0.77      0.73        56
           1       0.80      0.77      0.78        52

    accuracy                           0.77       144
   macro avg       0.79      0.77      0.78       144
weighted avg       0.78      0.77      0.77       144

accuracy is 0.7708333333333334



Results for: Decision Tree

[[29  6  1]
 [ 4 39 13]
 [ 0 21 31]]
              precision    recall  f1-score   support

          -1       0.88      0.81      0.84        36
           0       0.59      0.70      0.64        56
           1       0.69      0.60      0.64        52

    accuracy                           0.69       144
   macro avg       0.72      0.70      0.71       144
weighted avg       0.70      0.69      0.69       144

accuracy is 0.6875



Results for: Random Forest

[[31  4  1]
 [ 3 49  4]
 [ 0 12 40]]
              precision    recall  f1-score   support

          -1       0.91      0.86      0.89        36
           0       0.75      0.88      0.81        56
           1       0.89      0.77      0.82        52

    accuracy                           0.83       144
   macro avg       0.85      0.84      0.84       144
```

```
weighted avg       0.84      0.83      0.83       144

accuracy is 0.8333333333333334



Results for: Perceptron

[[33  2  1]
 [16 34  6]
 [ 3 25 24]]
              precision    recall  f1-score   support

          -1       0.63      0.92      0.75        36
           0       0.56      0.61      0.58        56
           1       0.77      0.46      0.58        52

    accuracy                           0.63       144
   macro avg       0.66      0.66      0.64       144
weighted avg       0.65      0.63      0.62       144

accuracy is 0.6319444444444444



Results for: RBF

[[32  4  0]
 [ 5 48  3]
 [ 0 14 38]]
              precision    recall  f1-score   support

          -1       0.86      0.89      0.88        36
           0       0.73      0.86      0.79        56
           1       0.93      0.73      0.82        52

    accuracy                           0.82       144
   macro avg       0.84      0.83      0.83       144
weighted avg       0.83      0.82      0.82       144

accuracy is 0.8194444444444444

```

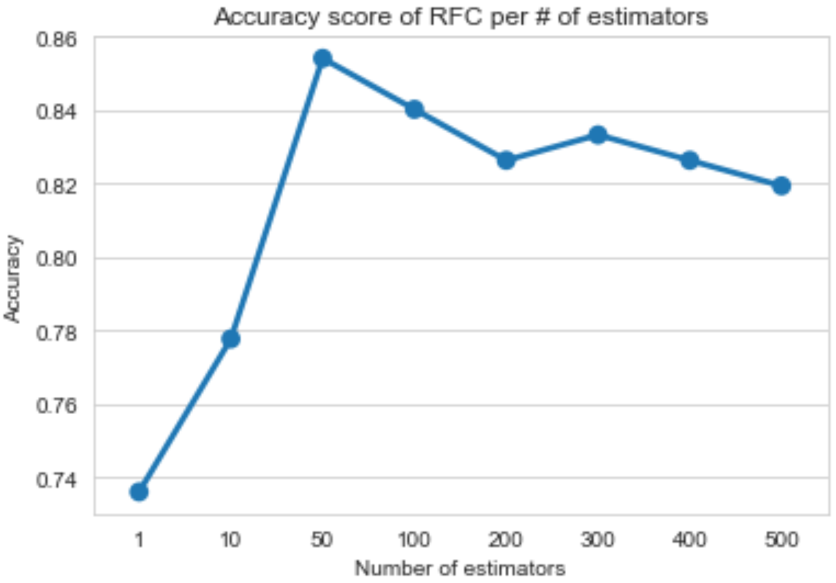
```


Results for: Linear

[[29  6  1]
 [ 4 43  9]
 [ 0 12 40]]
              precision    recall  f1-score   support

          -1       0.88      0.81      0.84        36
           0       0.70      0.77      0.74        56
           1       0.80      0.77      0.78        52

    accuracy                           0.78       144
   macro avg       0.79      0.78      0.79       144
weighted avg       0.78      0.78      0.78       144

accuracy is 0.7777777777777778



Results for: Polynomial

[[ 0 36  0]
 [ 0 56  0]
 [ 0 52  0]]
              precision    recall  f1-score   support

          -1       0.00      0.00      0.00        36
           0       0.39      1.00      0.56        56
           1       0.00      0.00      0.00        52

    accuracy                           0.39       144
   macro avg       0.13      0.33      0.19       144
weighted avg       0.15      0.39      0.22       144

accuracy is 0.3888888888888889



Results for: Sigmoid

```

```
[[32  4  0]
 [ 6 46  4]
 [ 1 18 33]]
              precision    recall  f1-score   support

          -1       0.82      0.89      0.85        36
           0       0.68      0.82      0.74        56
           1       0.89      0.63      0.74        52

    accuracy                           0.77       144
   macro avg       0.80      0.78      0.78       144
weighted avg       0.79      0.77      0.77       144

accuracy is 0.7708333333333334



                 model  accuracy score
0  Logistic Regression        0.770833
1        Decision Tree        0.687500
2        Random Forest        0.833333
3           Perceptron        0.631944
4                  RBF        0.819444
5               Linear        0.777778
6           Polynomial        0.388889
7              Sigmoid        0.770833
```

As can be seen in the table above, Random Forest is the most accurate classifier, with an accuracy of 83.3%. Let's further explore the number of estimators in the forest. As a general rule, when the number of estimators increases, the classifier performs better.

5.2 Exploratory Tonalities Random Forest Classifier

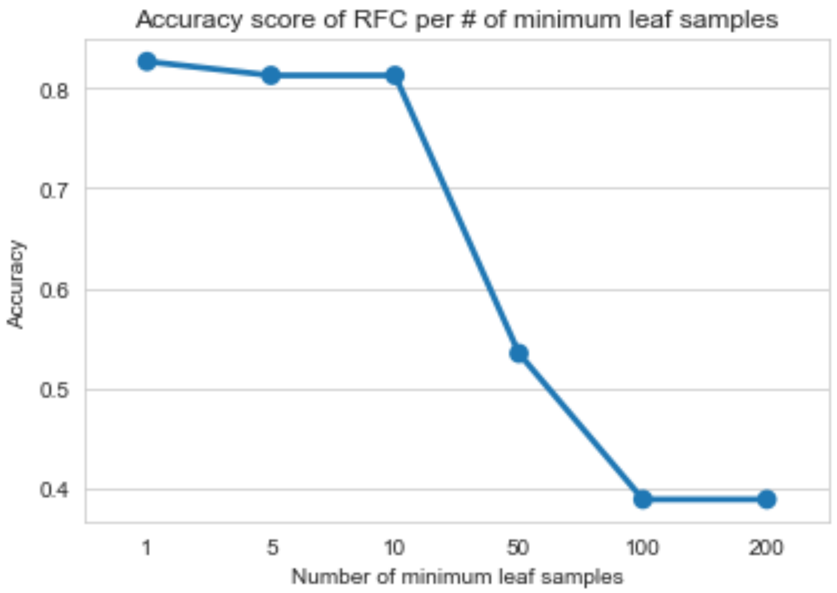
```
# Exploring the number of estimators in the random forest
score = []
est = []
estimators = [1, 10, 50, 100, 200, 300, 400, 500]
for e in estimators:
    rfc1 = RandomForestClassifier(n_estimators=e, random_state=52)
    pred1 = rfc1.fit(X_train, y_train).predict(X_test)
    accuracy = accuracy_score(y_test, pred1)
    score.append(accuracy)
    est.append(e)
plot = sns.pointplot(x=est, y=score)
plot.set(xlabel='Number of estimators', ylabel='Accuracy',
         title='Accuracy score of RFC per # of estimators')
plt.show()
```



In fact, when the number of estimators increases, the RFC performs better. However, at 200 estimators, it tends to stabilize. Obviously, 200 estimators are sufficient for this dataset.

Another variable can be explored, such as the minimum number of samples required for a leaf node.

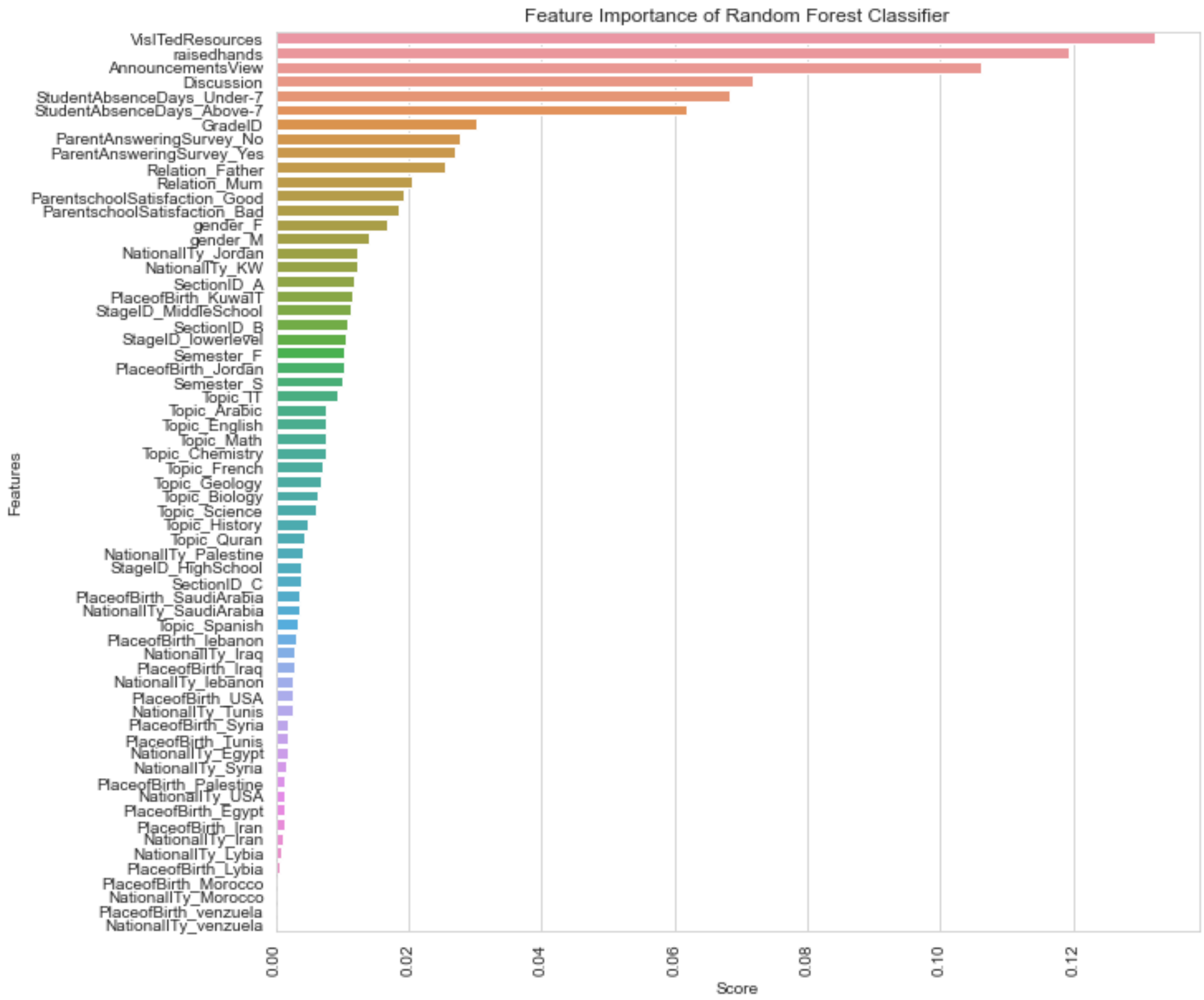
```
# Exploring minimum leaf samples
score = []
leaf = []
leaf_options = [1, 5, 10, 50, 100, 200]
for l in leaf_options:
    rfc2 = RandomForestClassifier(n_estimators=200, random_state=52, min_samples_leaf=l)
    pred2 = rfc2.fit(X_train, y_train).predict(X_test)
    accuracy = accuracy_score(y_test, pred2)
    score.append(accuracy)
    leaf.append(l)
plot = sns.pointplot(x=leaf, y=score)
plot.set(xlabel='Number of minimum leaf samples', ylabel='Accuracy',
         title='Accuracy score of RFC per # of minimum leaf samples')
plt.show()
```



In this case, we can see that the accuracy fraction simply decreases as the minimum leaf sample increases. Therefore, it is better to keep this value at the default value of 1.

Let's evaluate the importance of RFC features.

```
rfc = RandomForestClassifier(n_estimators=200, random_state=52)
pred = rfc.fit(X_train, y_train).predict(X_test)
dn = {'features':X.columns, 'score':rfc.feature_importances_}
df = pd.DataFrame.from_dict(data=dn).sort_values(by='score', ascending=False)
plot = sns.barplot(x='score', y='features', data=df, orient='h')
plot.set(xlabel='Score', ylabel='Features',
         title='Feature Importance of Random Forest Classifier')
plt.rcParams['figure.figsize']=(20,20)
plt.setp(plot.get_xticklabels(), rotation=90)
plt.show()
```



The number of visits to the course content is the most important feature.

Conclusion:

From our results, the number of visits to the course content, the number of days absent, the number of times students raised their hands in class, the number of times they checked for new announcements, whether or not they participated in discussions, gender, guardianship, and semester were indeed factors that influenced students' academic performance.