# Title: Factor affecting student performance

## Group 1

## Group Members:


1. Lim Chin Woon [Leader] (s2124307)
2. Muhammad Amirul Daniel bin Badrul Hisham (s2115750)
3. Nurul Shahirah binti Sha'ari (s2120876)
4. Qiyi Liu (s2106632)
5. Yeoh Li Tian (s2120306)


# Overview and Motivation

With the popularity of the Internet, sensors and various digital terminal devices, an interconnected world is taking shape. At the same time, with the explosive exponential growth of data, data analysis has penetrated into all aspects of our daily life and pushed us to an era of profound change. The education field has no exception. Through the mining and analysis of a large number of students' education process and organization management data collected by modern schools, we can get the characteristics with strong data causality with students' final scores, and then make a reliable early prediction of students' future performance. The prediction of students' academic performance not only helps the school to timely intervene in the curriculum, enrich the classroom content and improve the training plan, but also helps to analyze students' academic performance, understand students' learning trends, encourage excellent students to achieve better results, and encourage students with relatively weak foundation to establish confidence and make continuous progress.

# Related Work

1. Relationship of Parental Support on Healthy Habits, School Motivations and Academic Performance in Adolescents
   https://doi.org/10.3390/ijerph17030882

   -The objective of the study was to analyze how parental support relates to the physical activity practice, satisfaction with sports, level of physical activity, academic performance and alcohol consumption. Adolescents with little parental support show ($p < 0.001$) more boredom, less fun, worse academic performance and higher alcohol consumption. Gender shows differences ($p < 0.001$) experiencing girls more boredom, less fun, less PA practice and higher academic performance than boys. Age establishes ($p < 0.01$) that older adolescents (15–16 years old) experience more boredom, less fun, less PA practice, lower academic performance and higher alcohol consumption than young boys and girls (12–14 years old).
2. The Influence of Parents Educational Level on Secondary School Students Academic Achievements in District Rajanpur.

   https://files.eric.ed.gov/fulltext/EJ1079955.pdf

-Impact of parents educational level on students academic achievement at secondary level of education. After analysis of the data the research finds significant positive relationship between parents education level and academic achievements of students.

3. Gender Gaps in Student Academic Achievement and Inequality

   https://www.michaelsmith.cz/wp-content/uploads/2018/02/Gender_Gaps_Tsai_Smith_Hauser_RSE.pdf

   -PISA surveys, suggest that boys do better in math and science, whereas girls do better in reading but in overall, gender differences in family and school influences do not account for gender differences in academic achievement in any of the six countries.
4. Student Performance Prediction by Using Data Mining Classification Algorithms
   https://www.researchgate.net/profile/Dorina-Kabakchieva/publication/272178031_Student_Performance_Prediction_by_Using_Data_Mining_Classification_Algorithms/links/5a151ffda6fdccd697bc01ea/Student-Performance-Prediction-by-Using-Data-Mining-Classification-Algorithms.pdf

   -Predicting student performance using 4 algorithms which are a Rule Learner, a Decision tree, a Neural network, and K-Nearest Neighbour
5. Predicting Student Performance in a Portuguese Secondary Institution
   https://www.leonshpaner.com/projects/post/student_performance_models/Student_Performance_Models.pdf

   -Study on grades in on Portugal High school. Uses 6 different machine learning model and compare 8 different metric scores

# Research Question

We managed to obtain a relevant dataset here: https://www.kaggle.com/ishandutta/student-performance-data-set.

Based on the dataset, we will be answering these specific questions:

1. Studying how the variables can influence individual student performance in Portugal language study.
2. Predict student performance according to the variables after training the model using different machine learning algorithm.
3. Evaluate model accuracy and performance.

# Data Cleaning/Data Preprocessing

## Load library

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

## Import and read dataset

The initial dataset is separated with semicolon(;). The data is converted into table form.

```
dataseturl = 'https://drive.google.com/uc?id=18gjZIQO6OKZqxbk-BD2MpaEKZqJqUWWG'
pd.set_option('display.max_columns', None)
rawdf = pd.read_csv(dataseturl, sep=';')
print(rawdf.head())
```

```
  school sex  age address famsize Pstatus  Medu  Fedu     Mjob      Fjob  \
0     GP   F   18       U     GT3       A     4     4  at_home   teacher
1     GP   F   17       U     GT3       T     1     1  at_home     other
2     GP   F   15       U     LE3       T     1     1  at_home     other
3     GP   F   15       U     GT3       T     4     2   health  services
4     GP   F   16       U     GT3       T     3     3    other     other

    reason guardian  traveltime  studytime  failures schoolsup famsup paid  \
0  course   mother           2        2.0         0       yes     no   no
1  course   father           1        2.0         0        no    yes   no
2   other   mother           1        2.0         0       yes     no   no
3    home   mother           1        3.0         0        no    yes   no
4    home   father           1        2.0         0        no    yes   no

  activities nursery higher internet romantic  famrel  freetime  goout  Dalc
\
0         no     yes    yes       no       no       4         3      4     1

1         no      no    yes      yes       no       5         3      3     1

2         no     yes    yes      yes       no       4         3      2     2

3        yes     yes    yes      yes      yes       3         2      2     1

4         no     yes    yes       no       no       4         3      2     1


   Walc  health  absences  G1  G2  G3
0     1       3         4   0  11  11
1     1       3         2   9  11  11
2     3       3         6  12  13  12
3     1       5         0  14  14  14
4     2       5         0  11  13  13
```

# Attributes for the variables in the dataset.

1. school - student's school (binary: "GP" - Gabriel Pereira or "MS" - Mousinho da Silveira)
2. sex - student's sex (binary: "F" - female or "M" - male)
3. age - student's age (numeric: from 15 to 22)
4. address - student's home address type (binary: "U" - urban or "R" - rural)
5. famsize - family size (binary: "LE3" - less or equal to 3 or "GT3" - greater than 3)
6. Pstatus - parent's cohabitation status (binary: "T" - living together or "A" - apart)
7. Medu - mother's education (numeric: 0 - none,  1 - primary education (4th grade), 2 – 5th to 9th grade, 3 – secondary education or 4 – higher education)
8. Fedu - father's education (numeric: 0 - none,  1 - primary education (4th grade), 2 – 5th to 9th grade, 3 – secondary education or 4 – higher education)
9. Mjob - mother's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")

10. Fjob - father's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")
11. reason - reason to choose this school (nominal: close to "home", school "reputation", "course" preference or "other")
12. guardian - student's guardian (nominal: "mother", "father" or "other")
13. traveltime - home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)
14. studytime - weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)
15. failures - number of past class failures (numeric: n if 1<=n<3, else 4)
16. schoolsup - extra educational support (binary: yes or no)
17. famsup - family educational support (binary: yes or no)
18. paid - extra paid classes within the course subject (Math or Portuguese) (binary: yes or no)
19. activities - extra-curricular activities (binary: yes or no)
20. nursery - attended nursery school (binary: yes or no)
21. higher - wants to take higher education (binary: yes or no)
22. internet - Internet access at home (binary: yes or no)
23. romantic - with a romantic relationship (binary: yes or no)
24. famrel - quality of family relationships (numeric: from 1 - very bad to 5 - excellent)
25. freetime - free time after school (numeric: from 1 - very low to 5 - very high)
26. goout - going out with friends (numeric: from 1 - very low to 5 - very high)
27. Dalc - workday alcohol consumption (numeric: from 1 - very low to 5 - very high)
28. Walc - weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)
29. health - current health status (numeric: from 1 - very bad to 5 - very good)
30. absences - number of school absences (numeric: from 0 to 93)

---

**These grades are related with the  Portuguese subject:**

1. G1 - first period grade (numeric: from 0 to 20)
2. G2 - second period grade (numeric: from 0 to 20)
3. G3 - final grade (numeric: from 0 to 20, output target)

# Checking any non-availability (NA) inside the dataset

```
rawdf.isna().sum()
```

```
school        8
sex           3
age           0
address       0
famsize       0
Pstatus       0
Medu          0
Fedu          0
Mjob          0
Fjob          0
reason        0
guardian      0
traveltime    0
```

```
studytime     2
failures      0
schoolsup     0
famsup        0
paid          0
activities    0
nursery       0
higher        0
internet      0
romantic      0
famrel        0
freetime      0
goout         0
Dalc          0
Walc          0
health        0
absences      0
G1            0
G2            0
G3            0
dtype: int64
```

We can see there are variables that contain NA value. We will handle accordingly.

1. student's school (school)
2. student's sex (sex)
3. weekly study time(studytime)

# Imputation

We know that school and sex are in nominal pattern, hence We will handle using imputation by mode.

```
#Impute missing value in school and sex using mode.
Cols = ['school', 'sex']

for Col in Cols:
    rawdf[Col] = rawdf[Col].fillna(rawdf[Col].mode()[0])
    print(rawdf['school'].isna().sum())
```

```
0
0
```

Weekly study time is a numeric data where the dataset has categorized into a range of 1, 2, 3, 4. In this case, we will handle using imputation by median.

```
rawdf['studytime'] = rawdf['studytime'].fillna(rawdf['studytime'].median())
rawdf['studytime'].isna().sum()
```

```
0
```

## Check values pattern consistency

```
for nom_var in rawdf.columns[rawdf.dtypes.eq('object')]:
  print(rawdf[nom_var].value_counts().sort_index())
  print('\n')
```

```
GP    423
MS    226
Name: school, dtype: int64
```

```
F    383
M    266
Name: sex, dtype: int64
```

```
R    197
U    452
Name: address, dtype: int64
```

```
GT3    457
LE3    192
Name: famsize, dtype: int64
```

```
A     80
T    569
Name: Pstatus, dtype: int64
```

```
at_home     135
health       48
other       258
services    136
teacher      72
Name: Mjob, dtype: int64
```

```
at_home      42
health       23
other       367
services    181
teacher      36
Name: Fjob, dtype: int64
```

```
course        285
home          149
other          72
reputation    143
Name: reason, dtype: int64
```

```
father    153
mother    455
other      41
Name: guardian, dtype: int64
```

```
no     581
yes     68
Name: schoolsup, dtype: int64
```

```
no     251
yes    398
Name: famsup, dtype: int64
```

```
no     610
yes     39
Name: paid, dtype: int64
```

```
no     334
yes    315
Name: activities, dtype: int64
```

```
no     128
yes    521
Name: nursery, dtype: int64
```

```
no      69
yes    580
Name: higher, dtype: int64
```

```
no     151
yes    498
Name: internet, dtype: int64
```

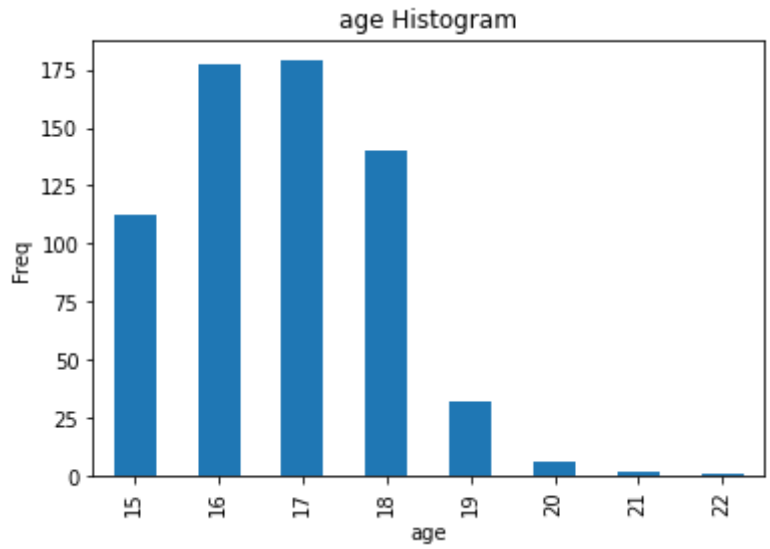
```
no     410
yes    239
Name: romantic, dtype: int64
```

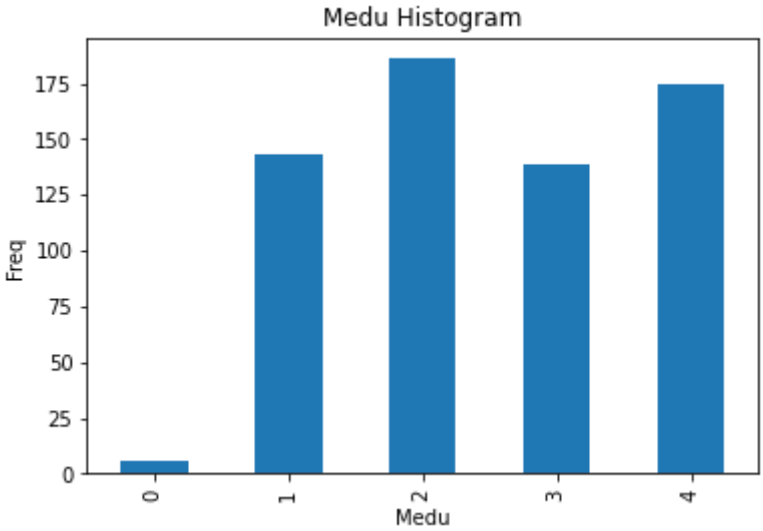
We can see that there are no inconsistent naming of values.

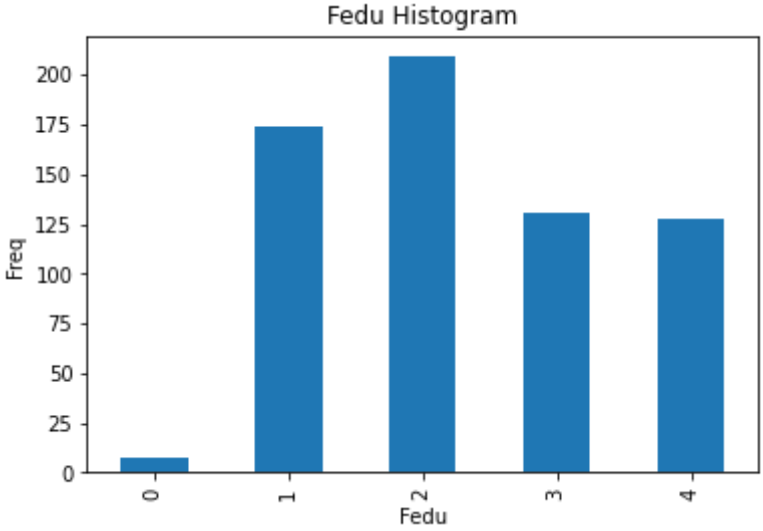
## Visualising distribution of numerical variables:

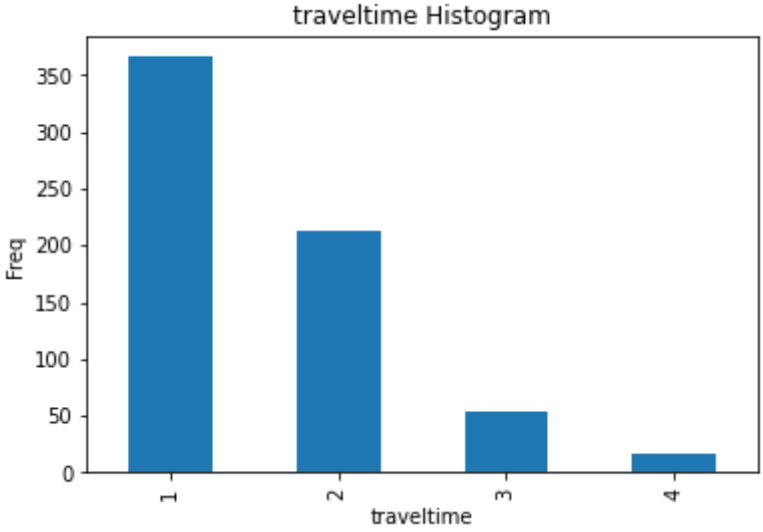
We can use histogram to visualise the distributions of numerical variables:

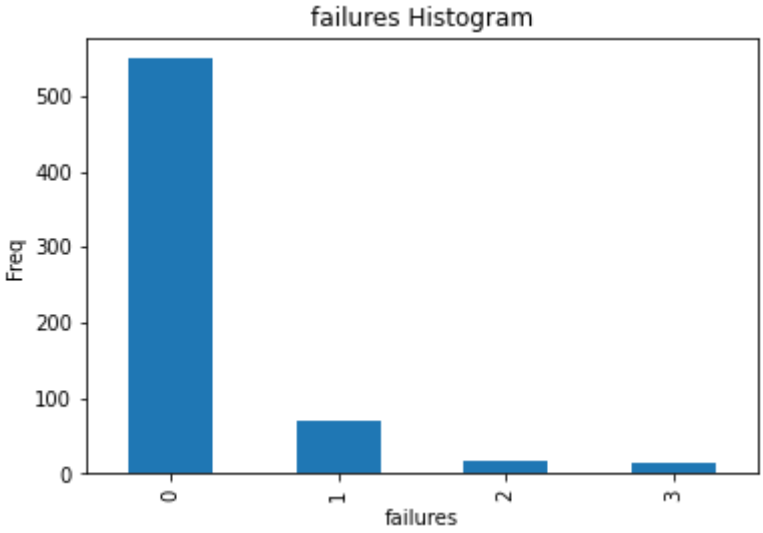
```
for numvar in rawdf.columns[rawdf.dtypes.eq('int64')]:
  plt.title(numvar + " Histogram")
  plt.xlabel(numvar)
  plt.ylabel('Freq')
  rawdf[numvar].value_counts().sort_index().plot(kind="bar")
  plt.show()

```

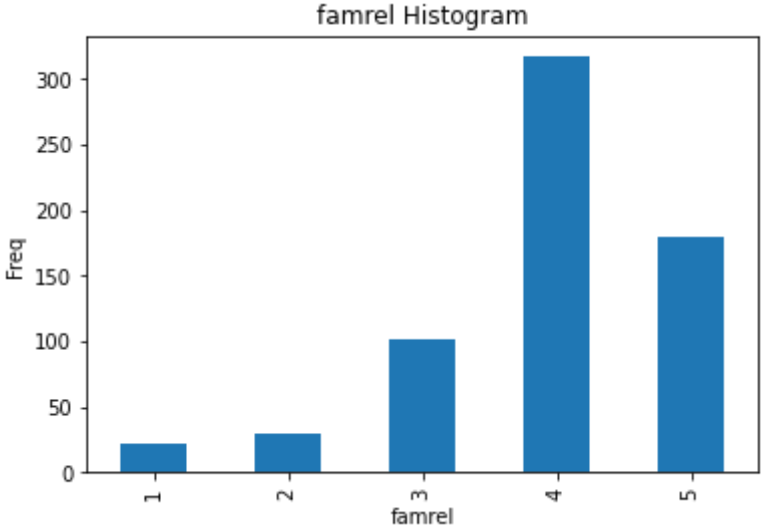


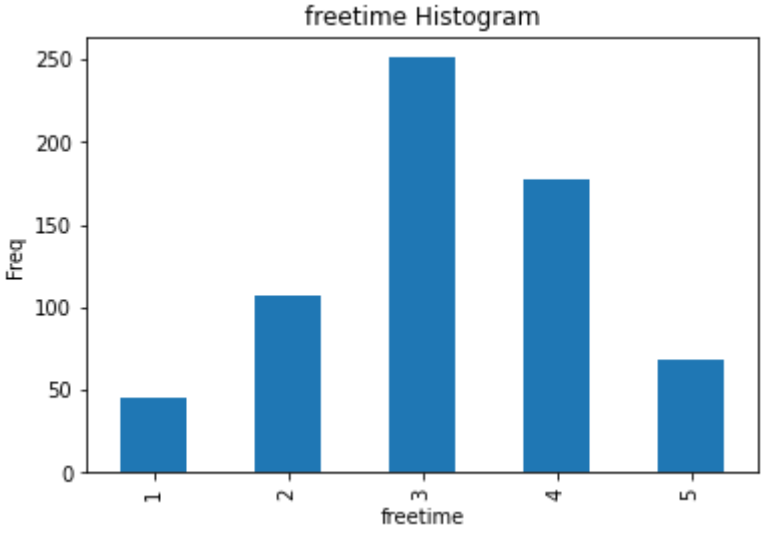

Medu Histogram
Freq
175
150
125
100
75
50
25
0
0
1
2
3
4
Medu

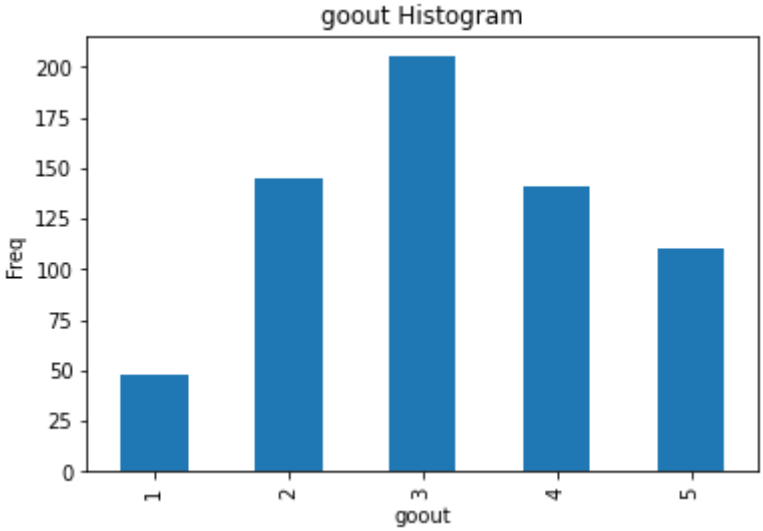

Fedu Histogram
Freq
200
175
150
125
100
75
50
25
0
0
1
2
3
4
Fedu

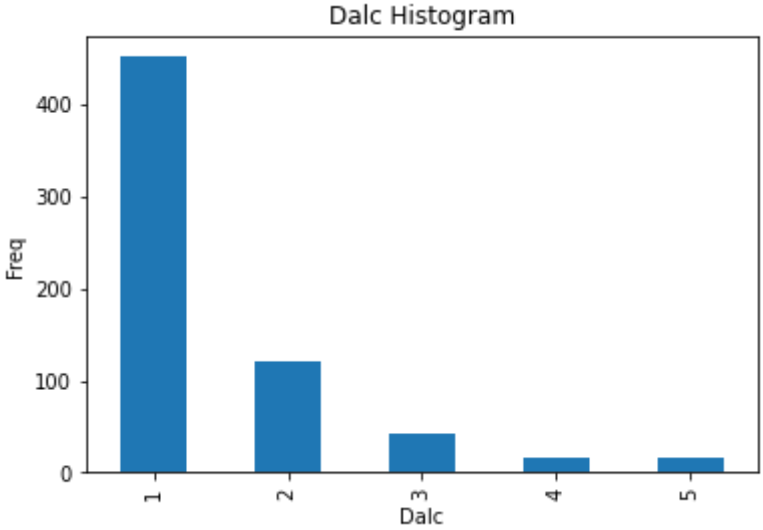

traveltime Histogram
Freq
350
300
250
200
150
100
50
0
1
2
3
4
traveltime

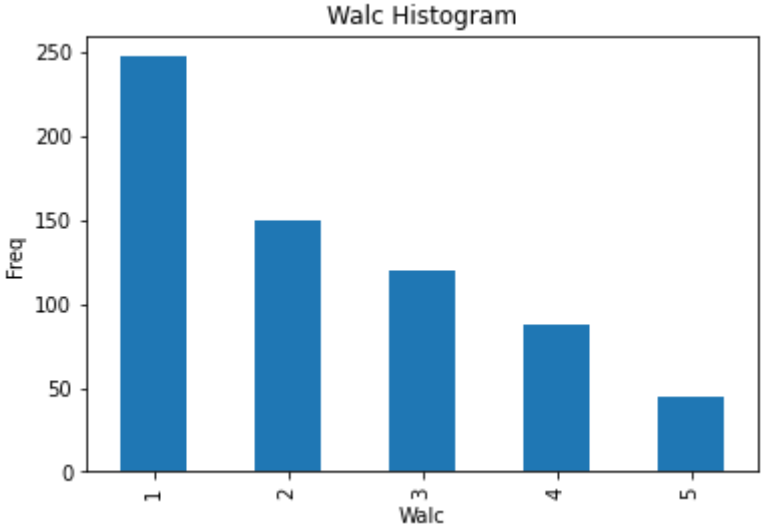

failures Histogram
Freq
500
400
300
200
100
0
0
1
2
3
failures

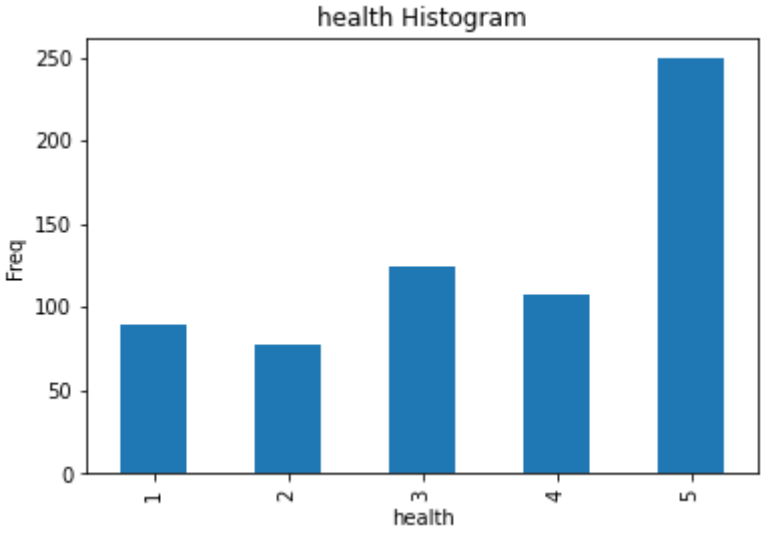

famrel Histogram
Freq
300
250
200
150
100
50
0
1
2
3
4
5
famrel

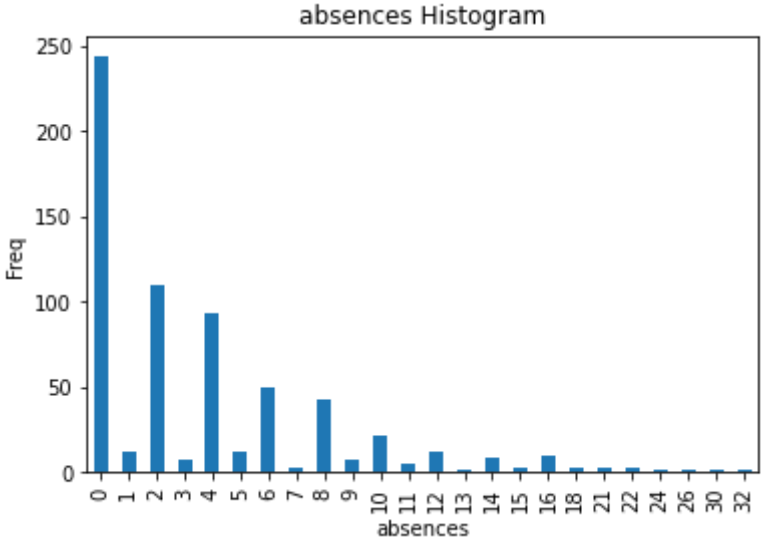

freetime Histogram
Freq
250
200
150
100
50
0
1
2
3
4
5
freetime

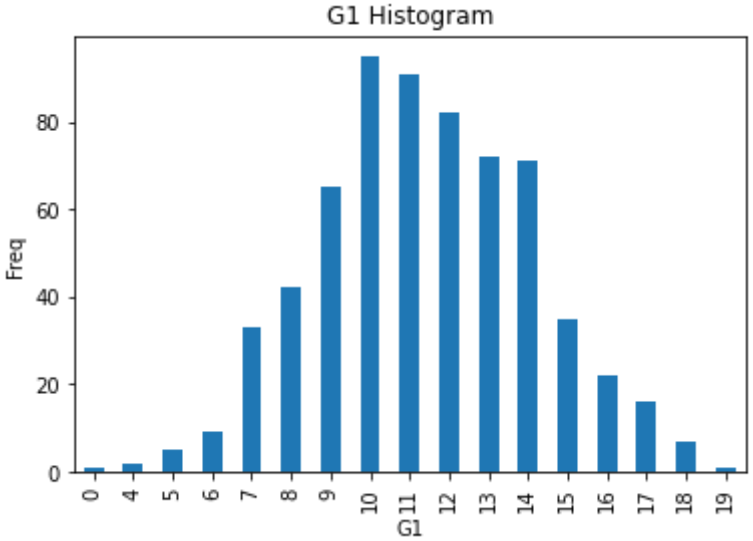

goout Histogram
Freq
goout
0
25
50
75
100
125
150
175
200
1
2
3
4
5

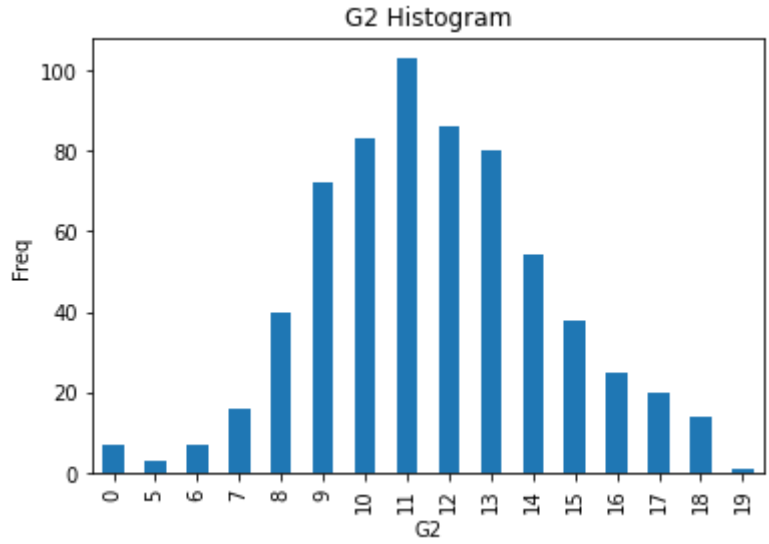

Dalc Histogram
Freq
Dalc
0
100
200
300
400
1
2
3
4
5

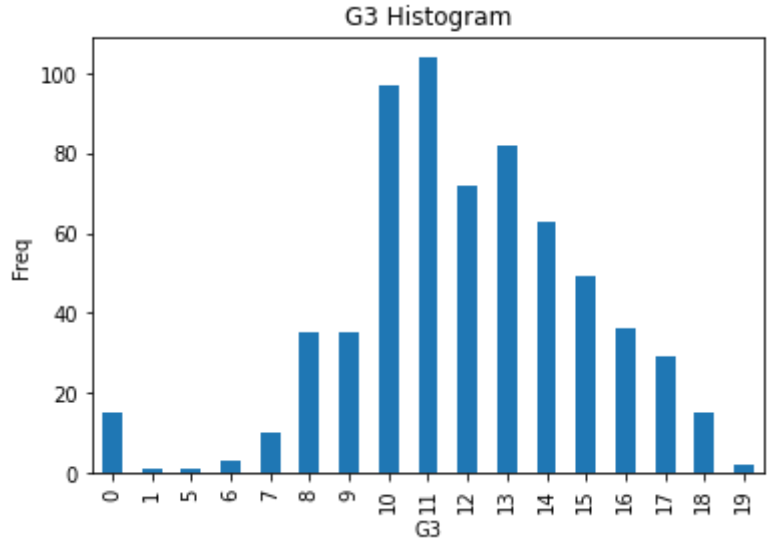

Walc Histogram
Freq
Walc
0
50
100
150
200
250
1
2
3
4
5


health Histogram
Freq
250
200
150
100
50
0
1
2
3
4
5
health


absences Histogram
Freq
250
200
150
100
50
0
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 18 21 22 24 26 30 32
absences


G1 Histogram
Freq
80
60
40
20
0
0 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
G1



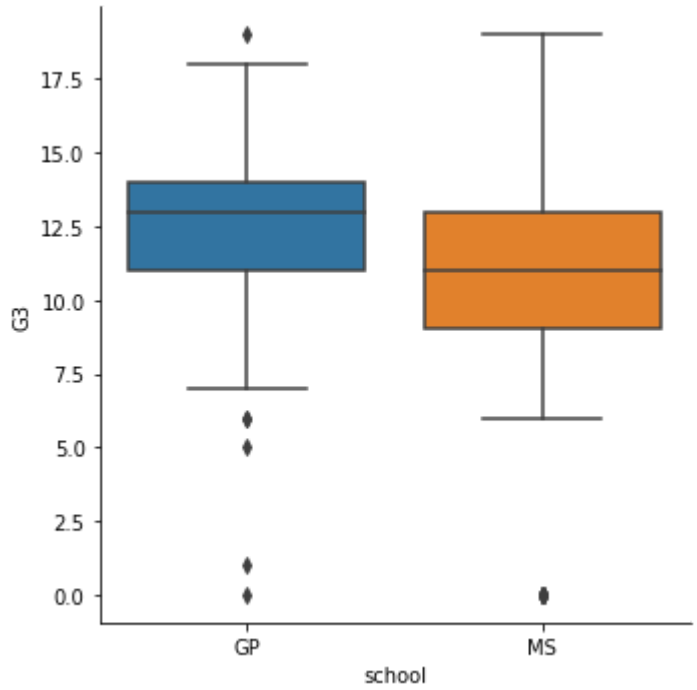

We see that there are very small number of students with Medu=0 or Fedu=0. On the other hand, the number of students with G3=0 is unexpectedly high.

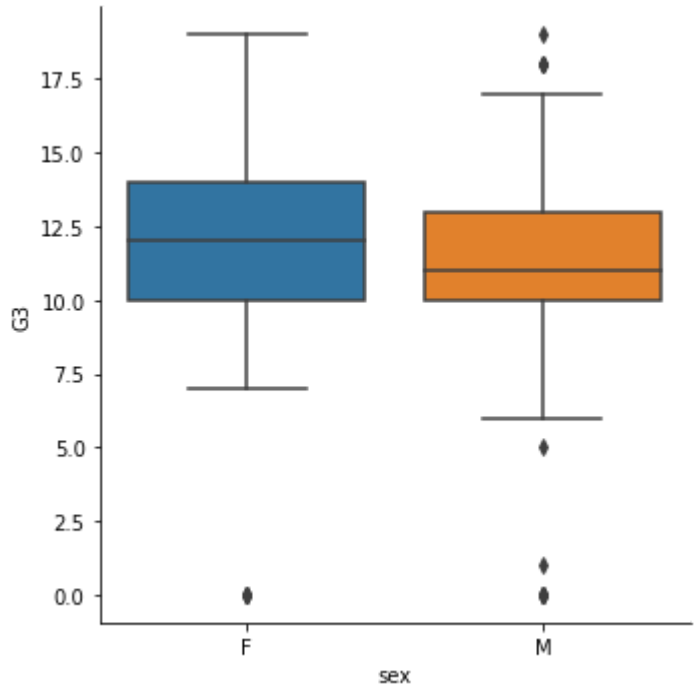
# Feature selection

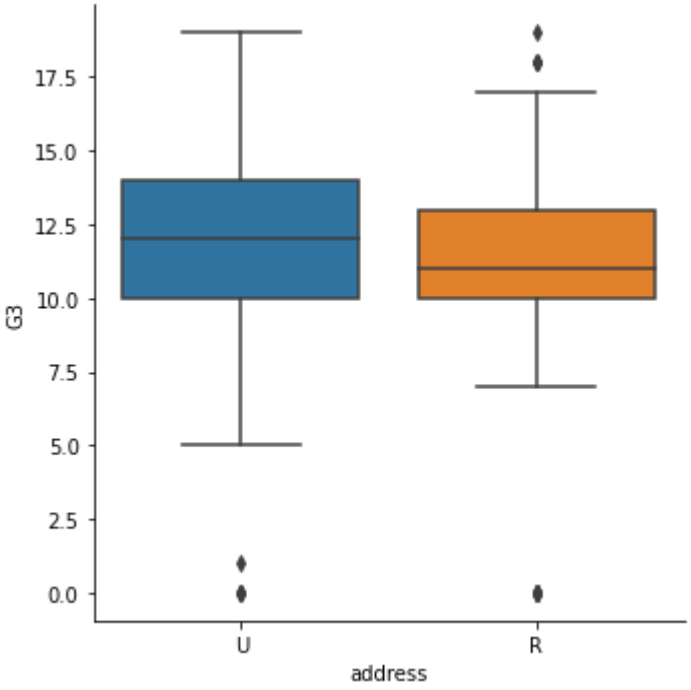
## For categorical variables:

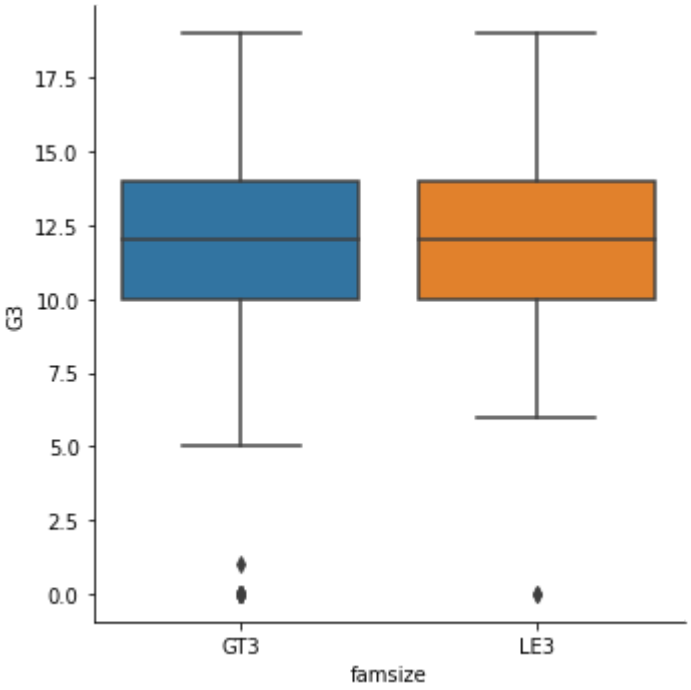
We use a boxplot to visualise the distribution of target G3 for different group of students

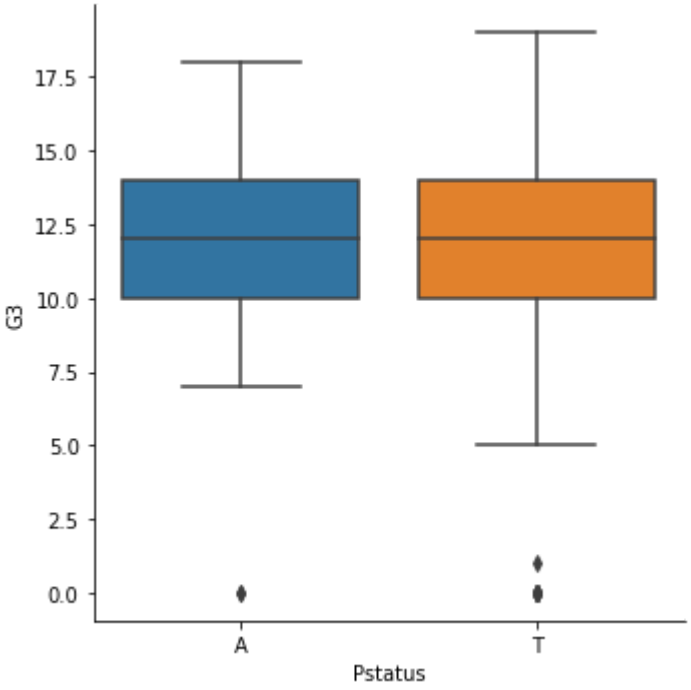
```
#filter out categorical variables
catvar = rawdf.columns[rawdf.dtypes.eq('object')]

# plot box plot
for cvar in catvar:
  sns.catplot(x=cvar, y="G3", kind="box", data=rawdf)
  plt.show()
```

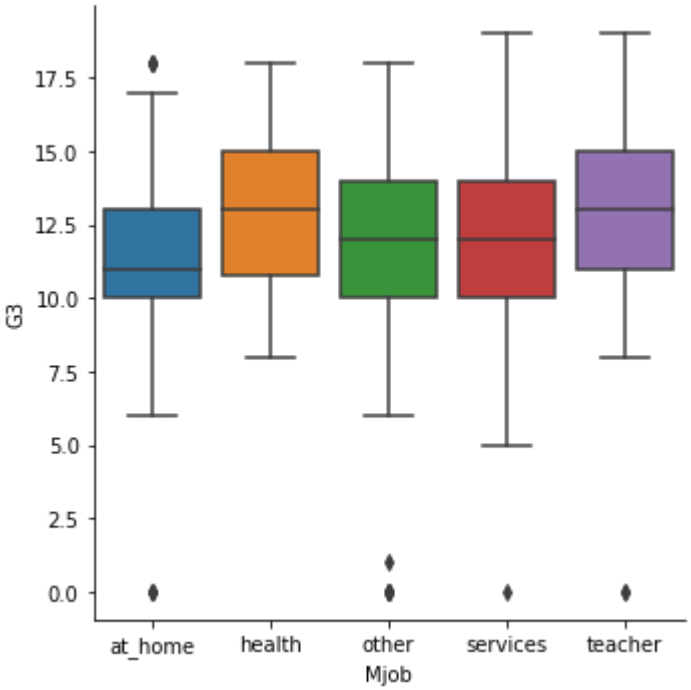

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
GP
MS
school

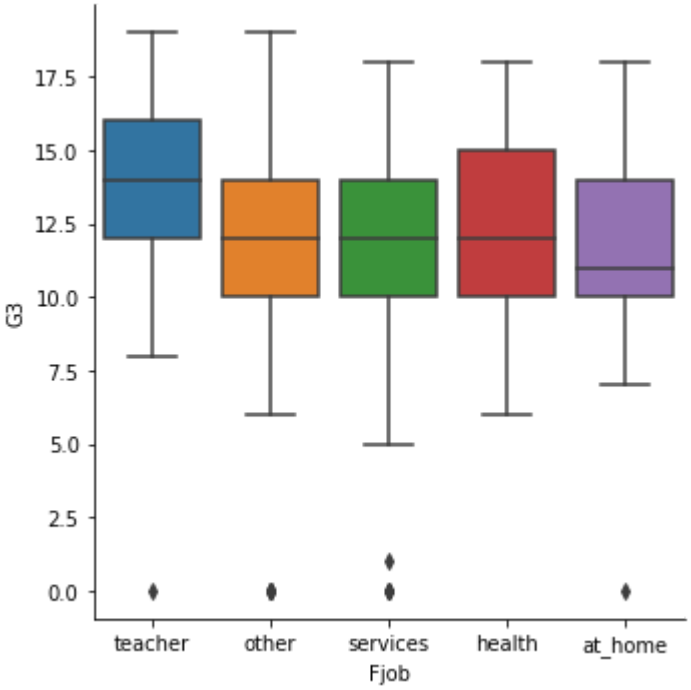

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
F
M
sex

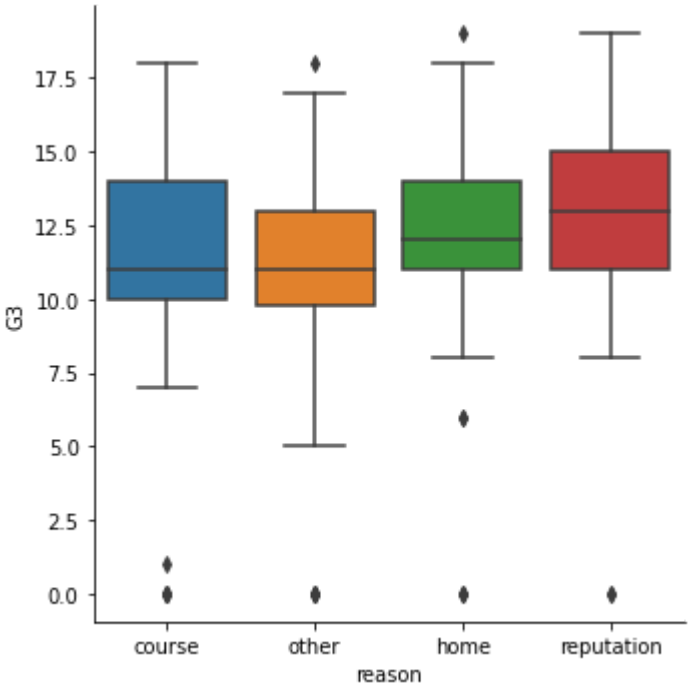

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
U
R
address

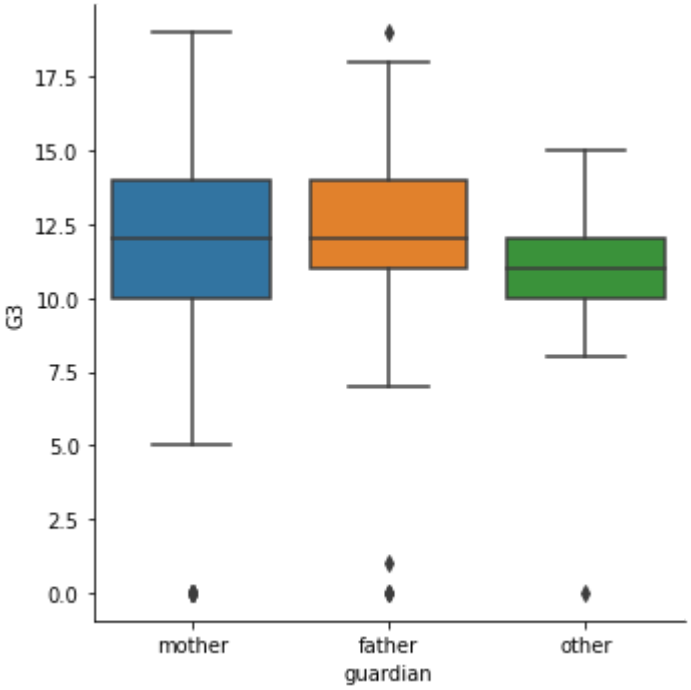

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
GT3
LE3
famsize

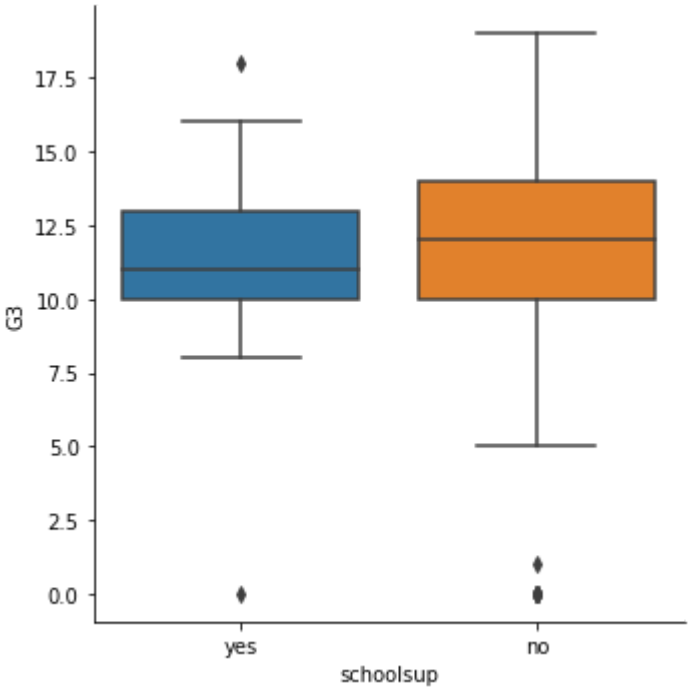

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
A
T
Pstatus

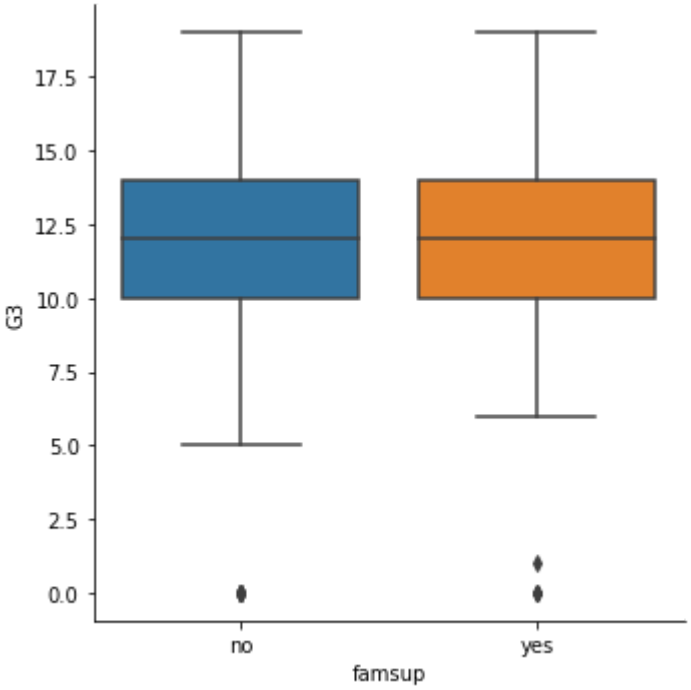

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
at_home
health
other
services
teacher
Mjob

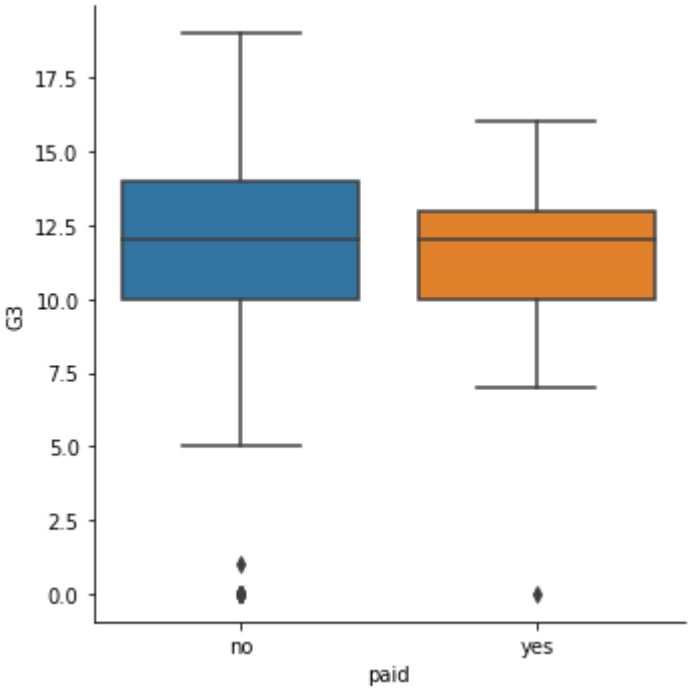

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
teacher
other
services
health
at_home
Fjob

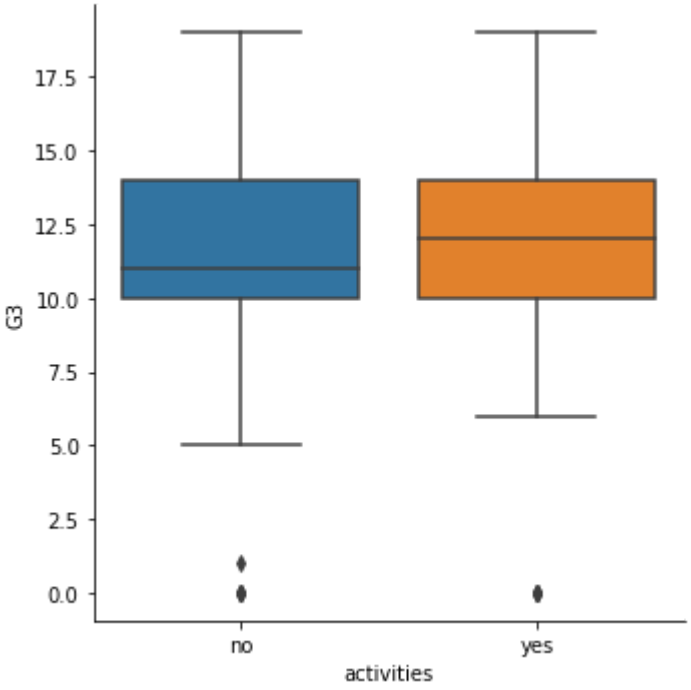

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
course
other
home
reputation
reason

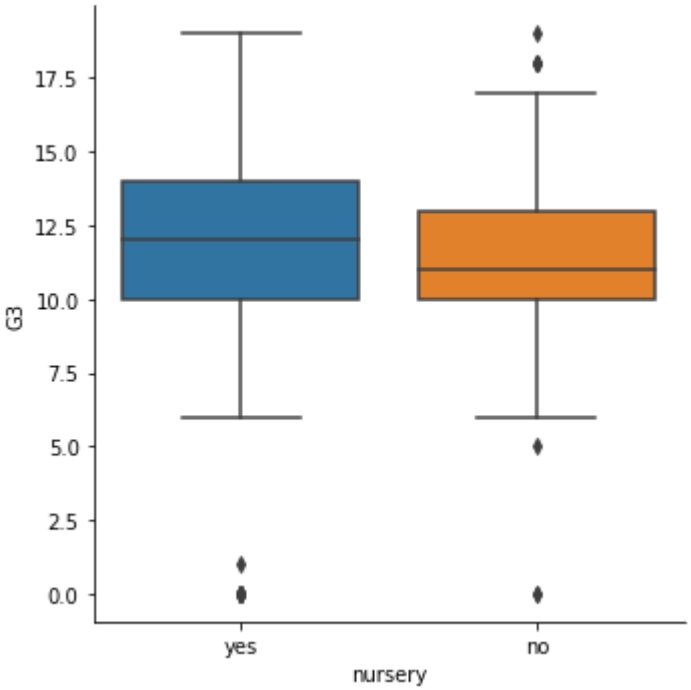

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
mother
father
other
guardian

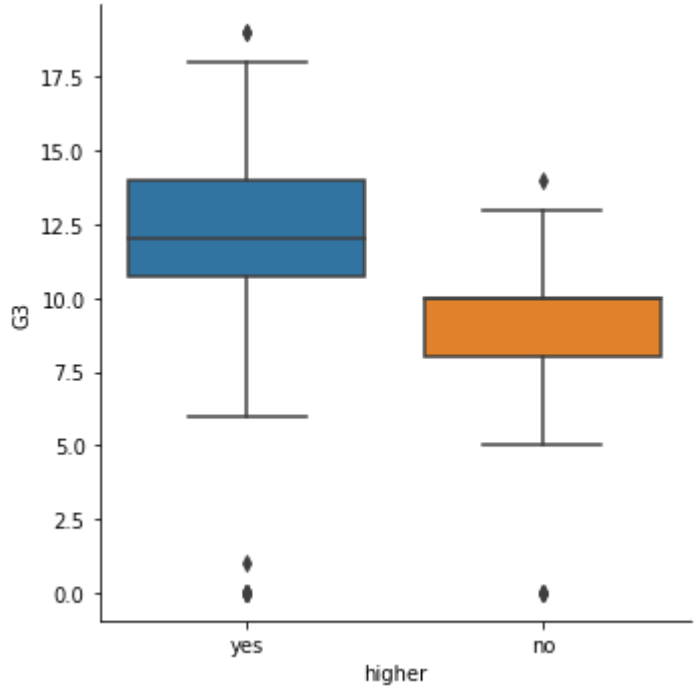

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
yes
no
schoolsup

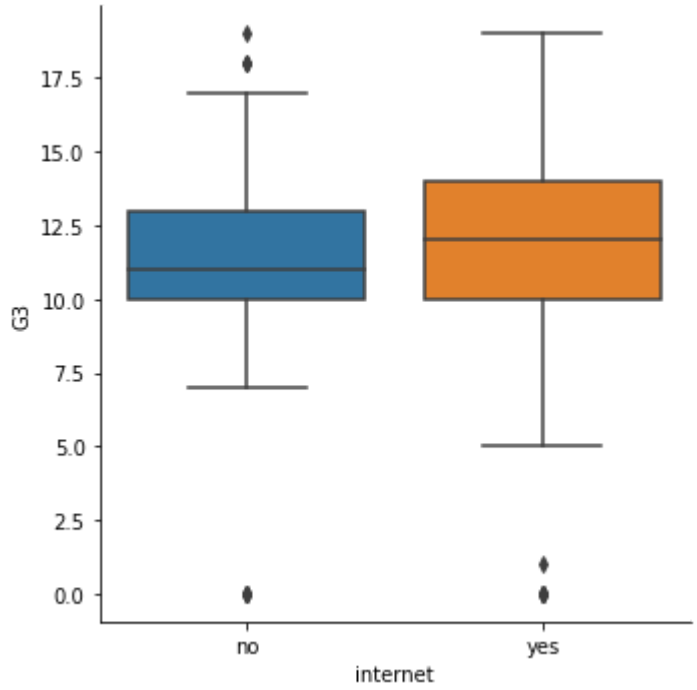

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
no
yes
famsup

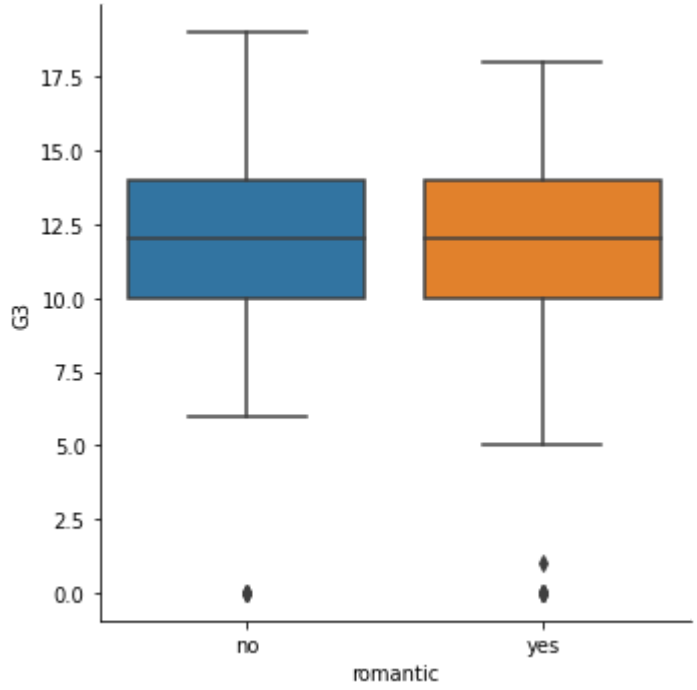

17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
no
yes
paid


17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
no
yes
activities


17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
yes
no
nursery


17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
yes
no
higher


17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
G3
no
yes
internet




From the figures, we can see that there is not much variation across different groups in the following variables:

1. family size (famsize),
2. parent's cohabitation status (Pstatus),
3. family educational support (famsup),
4. extra paid classes within the course subject (paid),
5. with a romantic relationship (romantic).

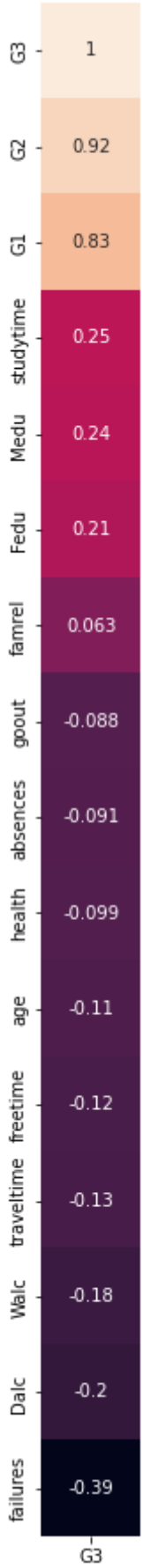
Hence we will consider to drop these variables.

## For numerical variables:

We use a heatmap and scatterplot to investigate on the corelation of each variables with G3

```
plt.figure(figsize=(1, 16))
corrcol = rawdf.corr()[['G3']]
sns.heatmap(corrcol.sort_values(by='G3', ascending=False), annot=True,
cbar=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f46ea6a2050>
```



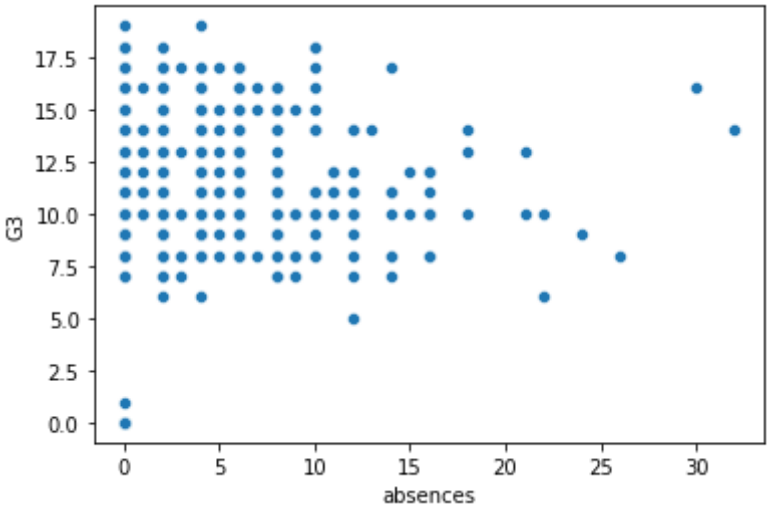

By observing the heatmap, we can identify those variables which show a very weak correlation ($|r| < 0.1$) to our output:

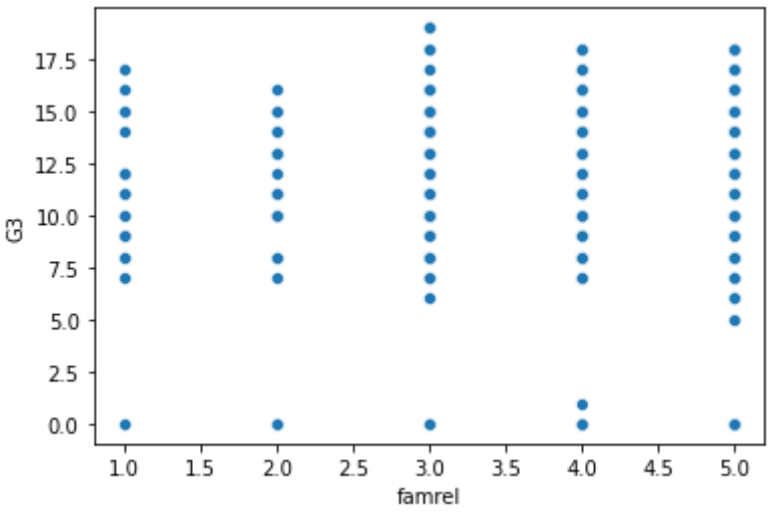
1. number of school absences (absences),
2. quality of family relationships (famrel),

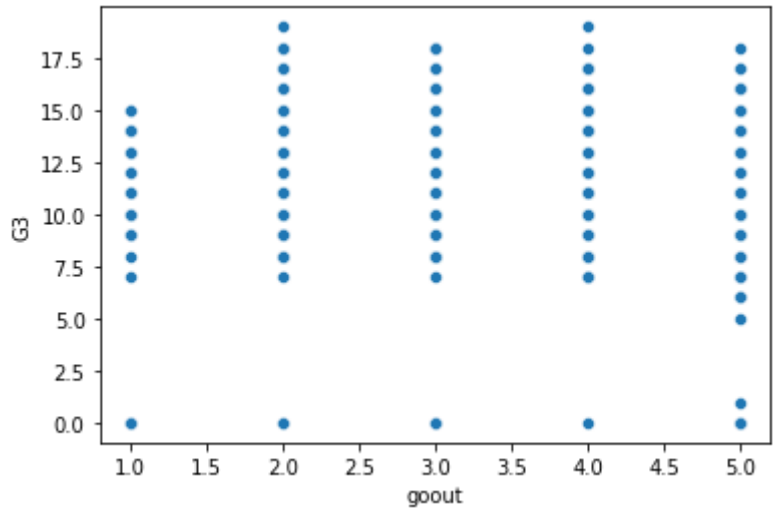
3. going out with friends (goout),
4. current health status (health).

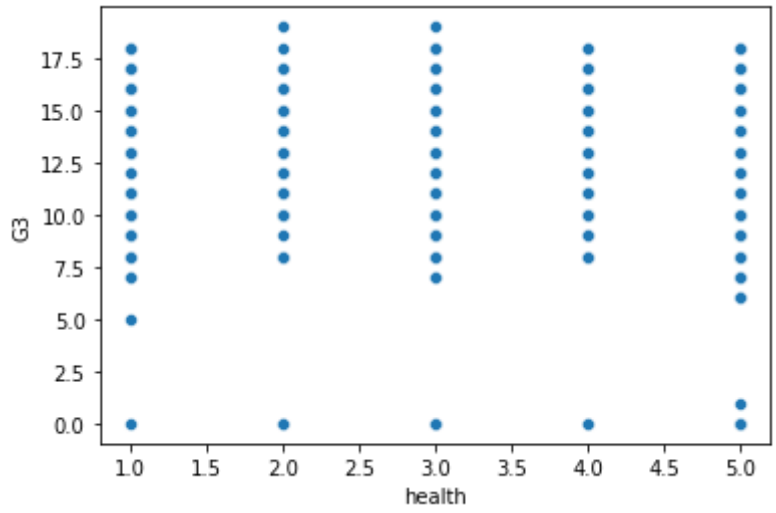
We will use scatterplot to identify if there are any abnormal values that affect the correlation.

```
#Plot G3 versus absences/famrel/goout/health
vars = ['absences', 'famrel','goout','health']

for var in vars:
    sns.scatterplot(x=rawdf[var], y = rawdf.G3)
    plt.show()
```






We can see that there is some extreme condition in absences variable. But it is still worth to keep as it has a negative correlation pattern against G3.

For the remaining variables, data is evenly distributed. Hence, we will keep the absences variable for further analysis and consider to drop the other three.

## Chi2 Test

To support our previous finding, we do a Chi-Squared Test to find features relevancy score to our target.

```
from sklearn.feature_selection import SelectKBest, chi2
from sklearn.preprocessing import LabelEncoder

chi2_df = rawdf.copy()
label_encoder = LabelEncoder()

for nom_var in chi2_df.columns[chi2_df.dtypes.eq('object')]:
  chi2_df[nom_var] = label_encoder.fit_transform(chi2_df[nom_var])

x = chi2_df.drop('G3',axis=1)
y = chi2_df['G3']

```

```
# Select two features with highest chi-squared statistics
chi2_selector = SelectKBest(chi2, k='all')
chi2_selector.fit(x, y)

# Look at scores returned from the selector for each feature
chi2_scores = pd.DataFrame(list(zip(chi2_df.columns,
chi2_selector.scores_)), columns=['features', 'score'])
chi2_scores.sort_values(by='score', ascending=False)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

| | features | score |
|---|---|---|
| **31** | G2 | 418.452113 |
| **30** | G1 | 333.615973 |
| **29** | absences | 294.362769 |
| **14** | failures | 255.779332 |
| **0** | school | 60.035797 |
| **26** | Dalc | 35.491003 |
| **8** | Mjob | 34.888847 |
| **10** | reason | 33.617500 |
| **6** | Medu | 33.567952 |
| **7** | Fedu | 32.368762 |
| **27** | Walc | 25.772026 |
| **15** | schoolsup | 21.756390 |
| **13** | studytime | 19.574536 |
| **28** | health | 14.518698 |
| **17** | paid | 13.252095 |
| **1** | sex | 12.930063 |
| **12** | traveltime | 12.919294 |
| **20** | higher | 11.343756 |
| **18** | activities | 11.131398 |
| **3** | address | 11.106601 |
| **22** | romantic | 10.741687 |
| **25** | goout | 10.105568 |
| **24** | freetime | 9.030872 |
| **4** | famsize | 8.130862 |
| **16** | famsup | 7.533753 |
| **9** | Fjob | 7.277847 |
| **11** | guardian | 6.306938 |
| **21** | internet | 5.694293 |
| **2** | age | 5.223356 |
| **19** | nursery | 3.187132 |

| | features | score |
|---|---|---|
| **23** | famrel | 3.039747 |
| **5** | Pstatus | 0.487380 |

```
cleandf = rawdf.drop(['famsize','Pstatus','famsup','paid','romantic',
'famrel','goout','health'], axis=1)
print(cleandf.columns)

```

```
Index(['school', 'sex', 'age', 'address', 'Medu', 'Fedu', 'Mjob', 'Fjob',
       'reason', 'guardian', 'traveltime', 'studytime', 'failures',
       'schoolsup', 'activities', 'nursery', 'higher', 'internet', 'freetime',
       'Dalc', 'Walc', 'absences', 'G1', 'G2', 'G3'],
      dtype='object')
```

## Dropping of Variables

We will be dropping variables based on the findings from the 3 previous sections. First we use boxplot, correlation and scatter plot to filter out a total of 8 candidates, then we make the final decision with the help of Chi-Squared Test. A critical value of 15 is set, where we find that all the 8 variables have chi2 value below it. Hence, they are all dropped:

1. family size (famsize),
2. parent's cohabitation status (Pstatus),
3. family educational support (famsup),
4. extra paid classes within the course subject (paid),
5. with a romantic relationship (romantic),
6. quality of family relationships (famrel),
7. going out with friends (goout),
8. current health status (health).

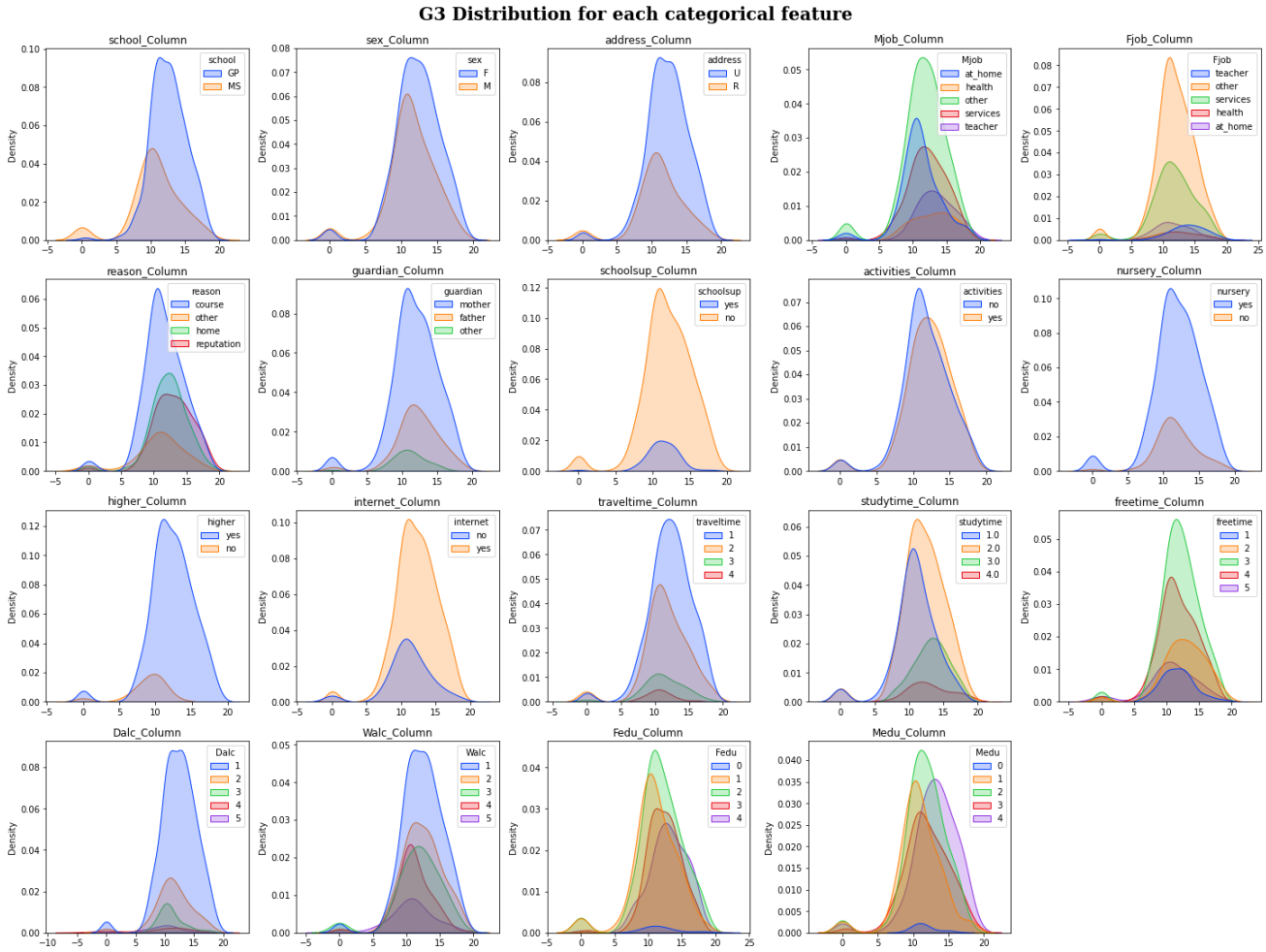
```
cleandf = rawdf.drop(['famsize','Pstatus','famsup','paid','romantic',
'famrel','goout','health'], axis=1)
print(cleandf.columns)

```

```
Index(['school', 'sex', 'age', 'address', 'Medu', 'Fedu', 'Mjob', 'Fjob',
       'reason', 'guardian', 'traveltime', 'studytime', 'failures',
       'schoolsup', 'activities', 'nursery', 'higher', 'internet', 'freetime',
       'Dalc', 'Walc', 'absences', 'G1', 'G2', 'G3'],
      dtype='object')
```

# Exploratory Data Analysis

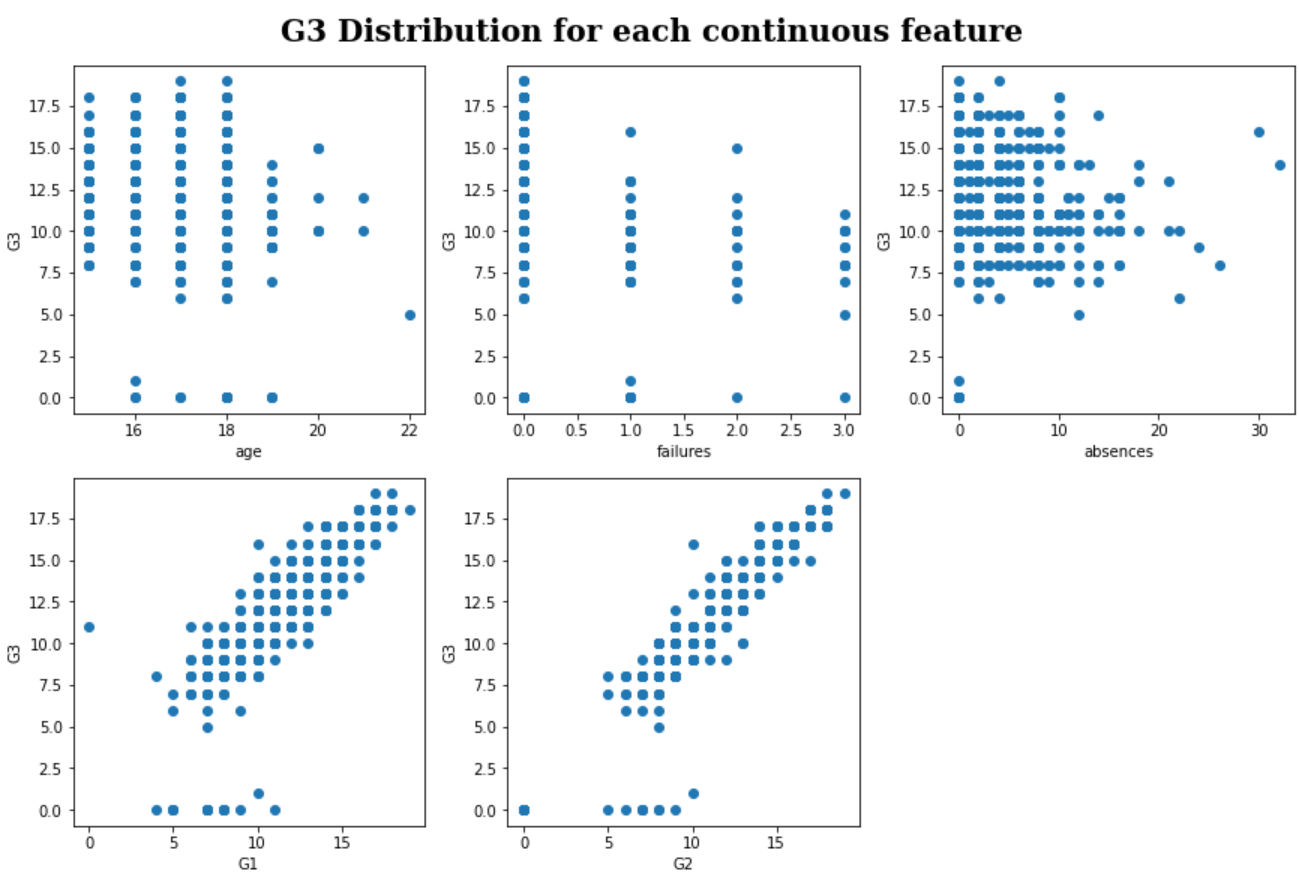
## Categorical Feature

```
nominal_columns = ['school','sex','address','Mjob','Fjob','reason','guardian',
                   'schoolsup','activities','nursery','higher','internet']
ordinal_columns = ['traveltime','studytime','freetime','Dalc','Walc','Fedu','Medu']

fig,axes = plt.subplots(4,5,figsize=(20,15),constrained_layout=True)
fig.suptitle('G3 Distribution for each categorical feature',size=20,
fontweight='bold', fontfamily='serif')
axes=axes.ravel()
axes[-1].remove()#removing non plotted axis

for i in range(len(nominal_columns + ordinal_columns)):
    #Creating plotting data variables
    ax = axes[i]
    col = (nominal_columns + ordinal_columns)[i]
    #plotting function
    sns.kdeplot(x='G3',data=cleandf,hue=col,ax=ax,shade=True,palette='bright')
    #Adjusting plot
    ax.set_xlabel("")
    ax.set_title(col+'_Column')
```




## Continuous Feature

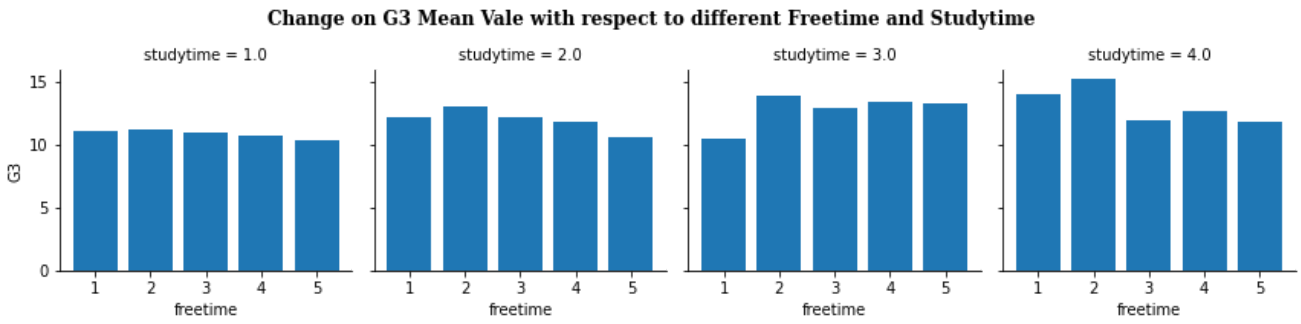
```
#'age','failures','absences','G1','G2','G3'
import seaborn as sns

fig,axs = plt.subplots(2,3,figsize=(12, 8),constrained_layout=True)
fig.suptitle('G3 Distribution for each continuous feature',size=20,
fontweight='bold', fontfamily='serif')

axs[0, 0].scatter(cleandf.age, cleandf.G3)
axs[0, 0].set_xlabel('age')
axs[0, 0].set_ylabel('G3')
axs[0, 1].scatter(cleandf.failures, cleandf.G3)
axs[0, 1].set_xlabel('failures')
axs[0, 1].set_ylabel('G3')
axs[0, 2].scatter(cleandf.absences, cleandf.G3)
axs[0, 2].set_xlabel('absences')
axs[0, 2].set_ylabel('G3')
axs[1, 0].scatter(cleandf.G1, cleandf.G3)
axs[1, 0].set_xlabel('G1')
axs[1, 0].set_ylabel('G3')
axs[1, 1].scatter(cleandf.G2, cleandf.G3)
axs[1, 1].set_xlabel('G2')
axs[1, 1].set_ylabel('G3')
fig.delaxes(axs[1,2])
plt.show()


```




# Multivariant Relationship

Explore how student manage their time can affect their exam score.

```
def exam_mean_facetgrid(df,face_column,x_col,mean_col='G3',title=''):
    plot_data = df.groupby([x_col,face_column]).mean()[mean_col].reset_index()
    g=sns.FacetGrid(data=plot_data,col=face_column,margin_titles=True)
    g.map(plt.bar,x_col,mean_col)
    g.fig.subplots_adjust(top=0.8)
    g.fig.suptitle(f'{title}',size=12, fontweight='bold', fontfamily='serif')

exam_mean_facetgrid(df=cleandf, face_column='studytime',
x_col='freetime',title='Change on G3 Mean Value with respect to different Freetime and Studytime')
```




Obviously more study time will have better result.

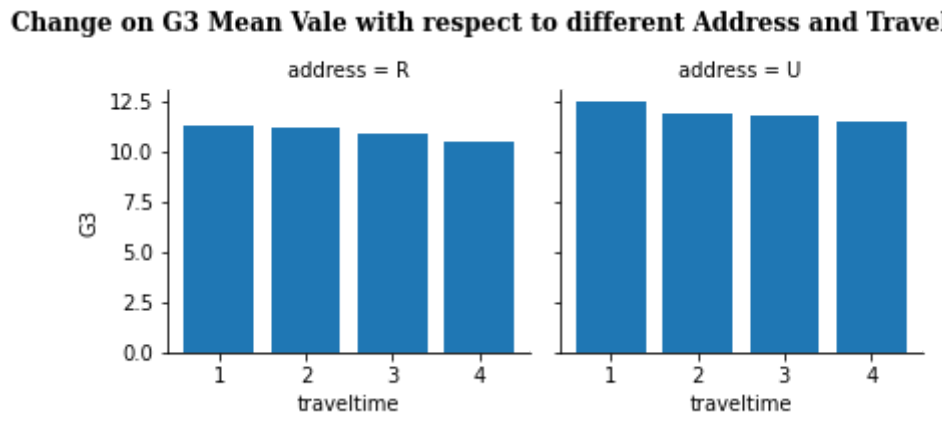
We can also spot that student with studytime = 4.0 (> 10 hours) and freetime = 1&2 perform better than those student with freetime = 3&4&5.

This may infere to following posibility:

1. Student study > 10 hours (probably 16 hours) and make him/her has less free time.
2. Student might has activity that can increase their performance. For example, exercise.

Explore how home address and travel time affect to exam result.

```
exam_mean_facetgrid(df=cleandf, face_column='address',
x_col='traveltime',title='Change on G3 Mean Value with respect to different Address and Traveltime')
```



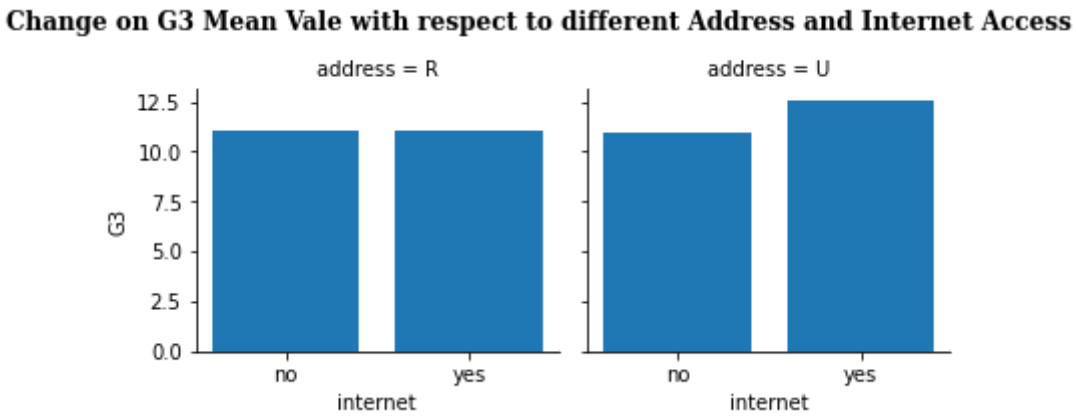

We can see when student stay in urban, they take less time to travel and perform better in exam.

This may infere to following posibility:

1. Urban has a more convenient travel system.
2. Shorter travel time, more energetic to learn (no need to wake up so early).

Explore how home address and internet access affect exam performance.

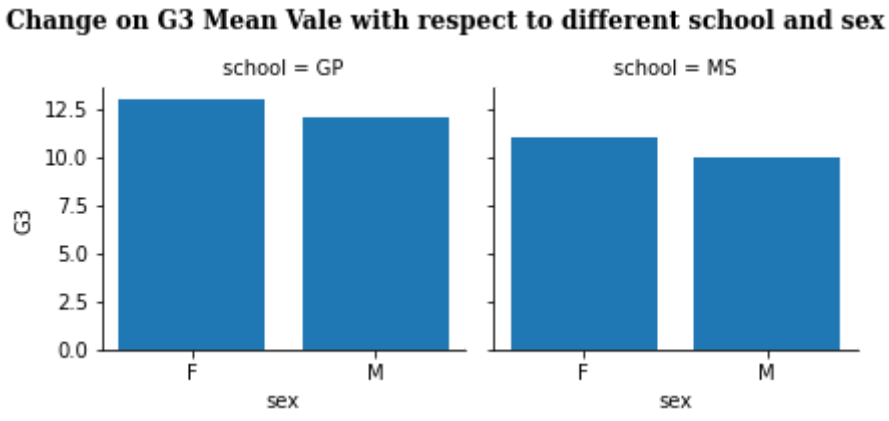
```
exam_mean_facetgrid(df=cleandf, face_column='address', x_col='internet',title='Change on G3 Mean Value with respect to different Address and Internet Access')
```




The students in urban more likely to have internet access and perform better in exam.(urban has more advance internet infrastructure-easier get access, easier to gain extra information for study)

From previous finding, female take higher median exam score in result. Explore how different school affect exam score for different sex.

```
exam_mean_facetgrid(df=cleandf, face_column='school', x_col='sex',title='Change on G3 Mean Value with respect to different school and sex')
```

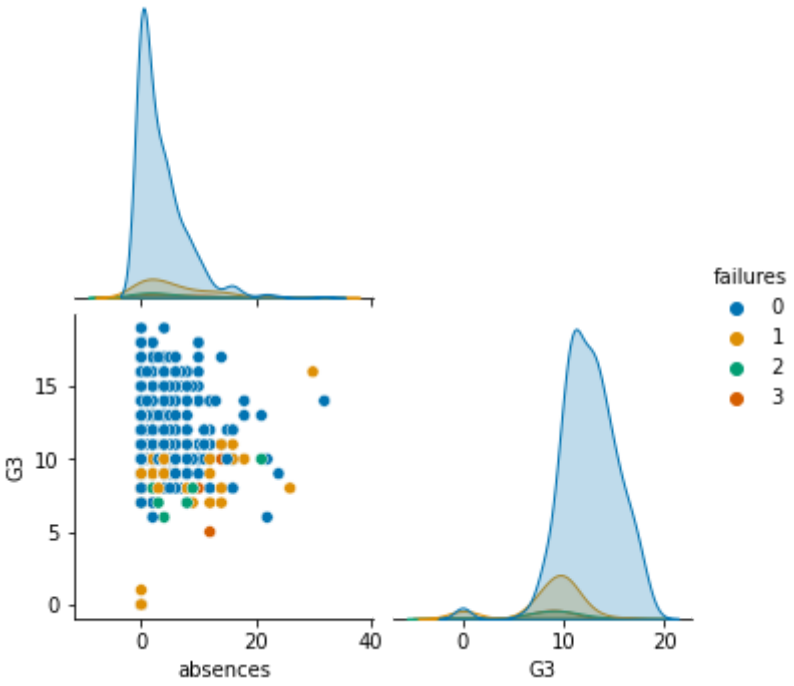



We can see female student contribute better result mo matter which school they belong to.

Explore how absences affect student failures and exam score.

```
pairplot_df = cleandf[['failures','absences','G3']]
sns.pairplot(pairplot_df, hue='failures', palette='colorblind', corner = True)
```

```
<seaborn.axisgrid.PairGrid at 0x7f46e85cad10>
```




From the chart, we can observe that when the absences increase, they are tend to get more failure and lower exam score.

# Data Modelling

## Split into Train and Test Dataset

```
from sklearn.model_selection import train_test_split as tts
import time

# variables
x = cleandf.drop(['G3'], axis=1)

# target
y = cleandf[['G3']]

# Split training data and test data (70%/30%)
x_train, x_test, y_train, y_test = tts(x, y, test_size=0.3, random_state=40)

```

## Feature Transformation and Standardization

```
from sklearn.preprocessing import OrdinalEncoder, StandardScaler
#encode categorical variable into ordinal
catvar = ['school', 'sex', 'address', 'Mjob', 'Fjob',
'reason','guardian','schoolsup','activities','nursery','higher','internet']
ordenc = OrdinalEncoder()
x_train[catvar] = ordenc.fit_transform(x_train[catvar])
```

```
#standardise x_train data
ss_x = StandardScaler()
x_train = pd.DataFrame(ss_x.fit_transform(x_train))
#x_train = pd.DataFrame(x_train)

#standardise y_train data
ss_y = StandardScaler()
y_train = ss_y.fit_transform(y_train)

#encode and standardsise x_test and y_test
x_test[catvar] = ordenc.transform(x_test[catvar])
x_test = pd.DataFrame(ss_x.transform(x_test))
#y_test = ss_y.transform(y_test)
```

# Model One: Random Forest Regressor

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV

start_time = time.time()
#hyper parameter tuning
rfr_param_grid = {'n_estimators':range(50,101)}
rfrReg = GridSearchCV(RandomForestRegressor(random_state = 40), rfr_param_grid, refit=True, verbose=0)
rfrReg.fit(x_train, y_train.ravel())

rfrtime = time.time()-start_time

print(rfrReg.best_estimator_)
```

```
RandomForestRegressor(n_estimators=99, random_state=40)
```

# Model Two: Support Vector Regressor

```
from sklearn import svm
from sklearn.model_selection import GridSearchCV

start_time = time.time()

#hyper parameter tuning
svr_param_grid = {'C': [0.1,1, 10, 100], 'gamma': [1,0.1,0.01,0.001],'kernel': ['rbf', 'poly', 'sigmoid']}
svrReg = GridSearchCV(svm.SVR(),svr_param_grid, refit=True, verbose=0)
svrReg.fit(x_train,y_train.ravel())

svrtime = time.time()-start_time

print(svrReg.best_estimator_)
```

```
SVR(C=100, gamma=0.001, kernel='sigmoid')
```

# Model Three: K Nearest Neighbors Regression Model

```
from sklearn.neighbors import KNeighborsRegressor
from sklearn.model_selection import GridSearchCV

start_time = time.time()

#hyper parameter tuning
knr_param_grid = {'n_neighbors': [3, 5, 7, 13, 15, 17, 19], 'weights':
['uniform','distance']}
knrReg = GridSearchCV(KNeighborsRegressor(),knr_param_grid, refit=True, verbose=0)
knrReg.fit(x_train, y_train.ravel())

knrtime = time.time()-start_time

print(knrReg.best_estimator_)
```

```
KNeighborsRegressor(n_neighbors=13, weights='distance')
```

## Model Four: Decision Tree Regressor

```
from sklearn.tree import DecisionTreeRegressor

start_time = time.time()

DtReg = DecisionTreeRegressor(random_state = 40)
DtReg.fit(x_train, y_train)

dttime = time.time()-start_time
```

## Model Five: Ridge Regressor

```
from sklearn.linear_model import Ridge

start_time = time.time()

rr_param_grid = {'alpha':[0,0.2,0.4,0.6,0.8,1,10,100]}
ridgeReg = GridSearchCV(Ridge(), rr_param_grid, refit=True, verbose=0)
ridgeReg.fit(x_train, y_train)

ridgetime = time.time()-start_time

print(ridgeReg.best_estimator_)
```

```
Ridge(alpha=1)
```

# Results Evalutaion

```
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error

def modelacc(modelname, model, x_test, y_test, ss_y):
  model_y_pred = model.predict(x_test)
  model_y_pred = ss_y.inverse_transform(model_y_pred.reshape(-1,1))

```

```
  modelr2 = r2_score(y_test, model_y_pred)
  modelmse = mean_squared_error(y_test, model_y_pred)
  modelmae = mean_absolute_error(y_test, model_y_pred)

  reseval = pd.DataFrame({
      'Model': [modelname],
      'R2_score': [modelr2],
      'MSE': [modelmse],
      'MAE': [modelmae]
  })

  return(reseval)

######
result = pd.DataFrame(columns= ["Model", "R2_score", "MSE", "MAE"])

#model one: rfr
rfrRegacc = modelacc('RFR', rfrReg, x_test, y_test, ss_y)
result = result.append(rfrRegacc, ignore_index = True)

#model two: svr
svrRegacc = modelacc('SVR', svrReg, x_test, y_test, ss_y)
result = result.append(svrRegacc, ignore_index = True)

#model three: KNR
neighacc = modelacc('KNR', knrReg, x_test, y_test, ss_y)
result = result.append(neighacc, ignore_index = True)

#model four: DT
DtRegacc = modelacc('DT', DtReg, x_test, y_test, ss_y)
result = result.append(DtRegacc, ignore_index = True)

#model five:
ridgeRegacc = modelacc('Ridge', ridgeReg, x_test, y_test, ss_y)
result = result.append(ridgeRegacc, ignore_index = True)

result['Time(s)'] = [rfrtime, svrtime, knrtime, dttime, ridgetime]
print(result)

```

```
   Model  R2_score       MSE       MAE    Time(s)
0    RFR  0.836988  1.967419  0.841388  46.833260
1    SVR  0.813252  2.253889  0.866753   4.999248
2    KNR  0.590293  4.944823  1.478549   0.671243
3     DT  0.620988  4.574359  1.117949   0.006966
4  Ridge  0.821261  2.157229  0.868879   0.333495
```

We see that random forest regressor (RFR) gives the best r2 score, but with a significantly longer time taken (especially when doing hyperparameter tuning). Ridge regressor comes close as second with a much shorter execution time. On the other hand, both k-nearest neighbour and decision tree regressor perform poorly on this set of data.

# Summary

In summary, from 33 variables we have dropped 8 of it which are family size (famsize), parent's cohabitation status (Pstatus), family educational support (famsup), extra paid classes within the course subject (paid), with a romantic relationship (romantic), quality of family relationships (famrel), going out with friends (goout) and current health status (health) due to irrelavent score to the student grade. The criteria for dropping the variables is based on boxplot, scatterplot and pearson correlation coefficient and chi-square test. From exploratory data analysis, we can conclude that study time, travel time, student's home location, their absences and gender had affect their exam grade. Five machine learning-based regressor namely Random Forest Regressor, Support Vector Regressor, K Nearest Neighbors Regression Model, Decision Tree Regressor and Ridge Regressor were implemented in this project. As a result, Random Forest Regressor were significantly better than other regressor.

# Limitation

1. One of the biggest limitation with K-NN is to choose the optimal number of neighbors to be consider while predicting the new data entry. Right number of neighbors is a must to produce the best prediction. Beside that, KNN also bad in large dataset. This is because, in large datasets, the cost of calculating the distance between the new point and each existing points is huge which degrades the performance of the algorithm.
2. Decision tree are inadequate to use when working with continuous variables as it will looses information when the variable are categorizes in different categories. As we can see that Decision Tree Regressor less performed on this dataset.

# Member's Contribution

| Member | Matric No. | Section |
|---|---|---|
| Lim Chin Woon (L) | s2124307 | Feature Selection, EDA , Ridge Model. |
| Muhammad Amirul Daniel bin Badrul Hisham | s2115750 | KNN Model, Limitation, Presentation. |
| Nurul Shahirah binti Sha'ari | s2120876 | Related Work, Summary , Limitation, DT Model. |
| Qiyi Liu | s2106632 | Overview and Motivation, Data Transformation, RFR Model. |
| Yeoh Li Tian | s2120306 | Data Cleaning (except feature selection), Results Evaluation, SVR Model. |

Research Question was by who again?
Maybe Data collection also?