STUDENT PERFORMANCE ANALYSIS

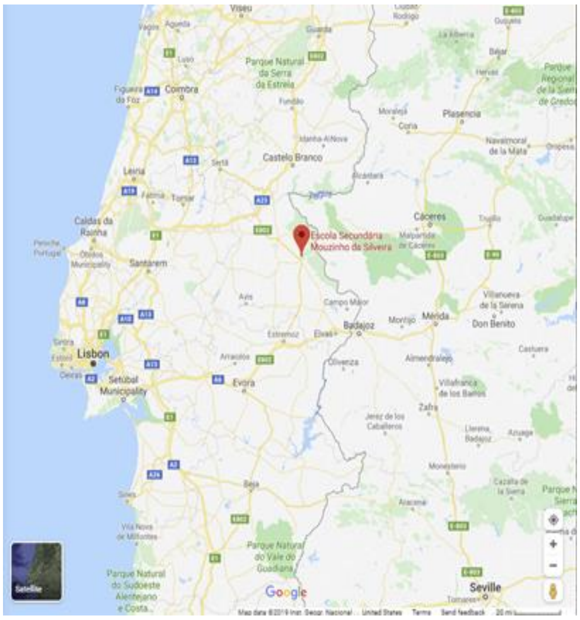
BUAN 6340 - Programming for Data Science

By,
TARONISH MADEKA
VIKRAM ARIKATH
VISHAL CEHKKALA
RAAHUL JAGADEESAN

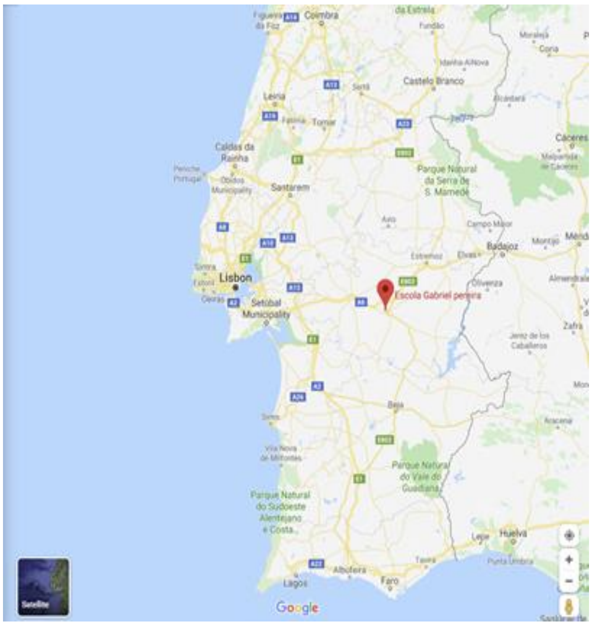
INTRODUCTION

This data set carries information about students' performances, demographic, social and school related features and various other attributes in secondary education of two Portuguese schools namely Escola Gabriel Pereira and Escola Secundária Mouzinho da Silveira. These two schools are roughly 75 miles apart. Data was collected by using school reports and questionnaires. Two datasets are provided regarding the performance in two distinct subjects: Mathematics (mat) and Portuguese language (por). In [Cortez and Silva, 2008], the two datasets were modelled under binary/five-level classification and regression tasks. The target attribute G3 has a strong correlation with attributes G2 and G1. This occurs because G3 is the final year grade, while G1 and G2 correspond to the 1st and 2nd period grades. It is more difficult to predict G3 without G2 and G1, but such prediction is much more valuable.

Education in Portugal is free of cost and mandatory for all until the age of 18. The education is regulated by the State through the Ministry of Education. Both public and private schools are available. Literacy rate in Portugal is 99.44%. According to the Programme for International Student Assessment (PISA) 2015, the average Portuguese 15-year-old student, when rated in terms of reading literacy, mathematics and science knowledge, is placed significantly above the Organisation for Economic Co-operation and Development (OECD) average, at a similar level as those students from Norway, Poland, Denmark and Belgium, with 501 points (493 is the average). The PISA results of the Portuguese students have been continuously improving, surpassing those of a number of other highly developed western countries like the US, Austria, France and Sweden.



Mouzinho da Silveira



Gabriel Pereira

PROJECT DESCRIPTION

The project primarily focuses on analysing the student performance at both the schools in both the Math and Portuguese subjects and understanding the factors that affect their performance.

The main reason to choose this dataset was it has many different and unique attributes. It had multiple categorical columns that could provide interesting and valuable insights.

As a team we wanted to check effects of attributes like alcohol consumption, romantic relationship on students' grades which is one of the aspects we have explored in this project. With this dataset we have done EDA and implemented linear regression, Neural Network, Decision Tree, Random Forest, Logistic Regression, SVM modelling techniques and summarized the best model and the significant factors the affect student performance.

DATA DESCRIPTION

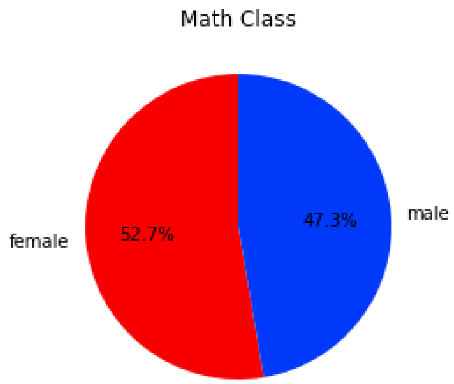
The primary data sets include 2 csv files containing the information of students in Portugal in 2 schools Gabriel Pereira and Escola Secundária Mouzinho da Silveira. Each dataset represents the students enrolled in Math and Portuguese classes respectively. Each data set consists of 34 columns. It has been sourced from the UCI Machine Learning Repository.

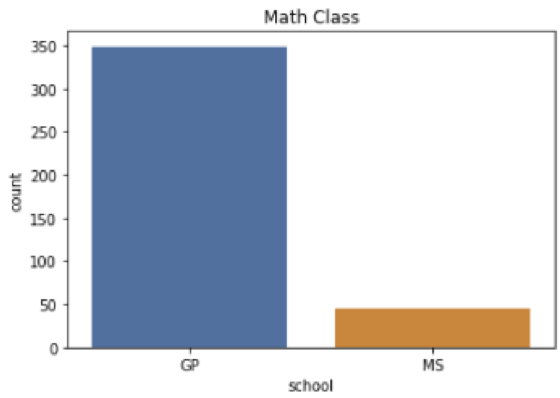
The attributes of the dataset are summarised as followed,

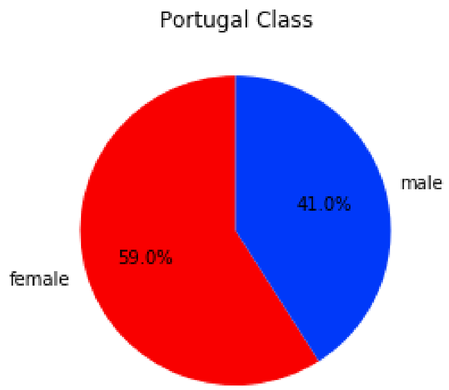
Attributes for both student-mat.csv (Math course) and student-por.csv (Portuguese language course) datasets:
1 school - student's school (binary: 'GP' - Gabriel Pereira or 'MS' - Mousinho da Silveira)
2 sex - student's sex (binary: 'F' - female or 'M' - male)
3 age - student's age (numeric: from 15 to 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 â€“ 5th to 9th grade, 3 â€“ secondary education or 4 â€“ higher education)
8 Fedu - father's education (numeric: 0 - none, 1 - primary education (4th grade), 2 â€“ 5th to 9th grade, 3 â€“ secondary education or 4 â€“ 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)

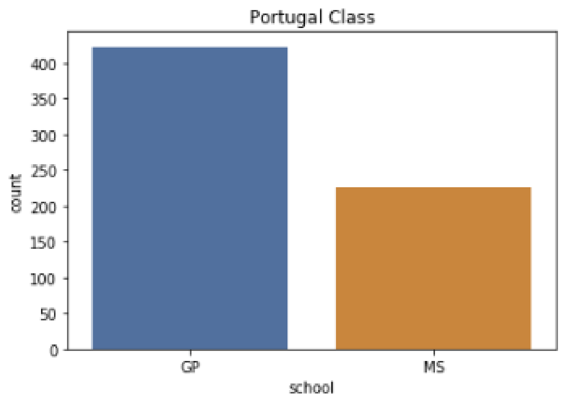
these grades are related with the course subject, Math or Portuguese:
31 G1 - first period grade (numeric: from 0 to 20)
31 G2 - second period grade (numeric: from 0 to 20)
32 G3 - final grade (numeric: from 0 to 20, output target)

EXPLORATORY DATA ANALYSIS







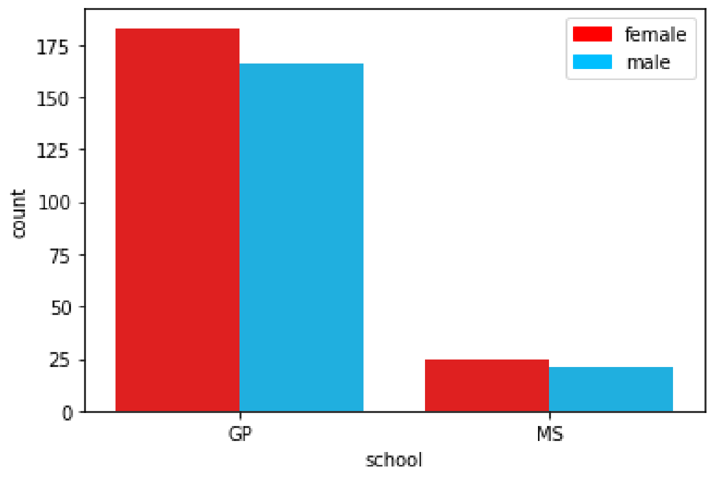


Math Class Observations:

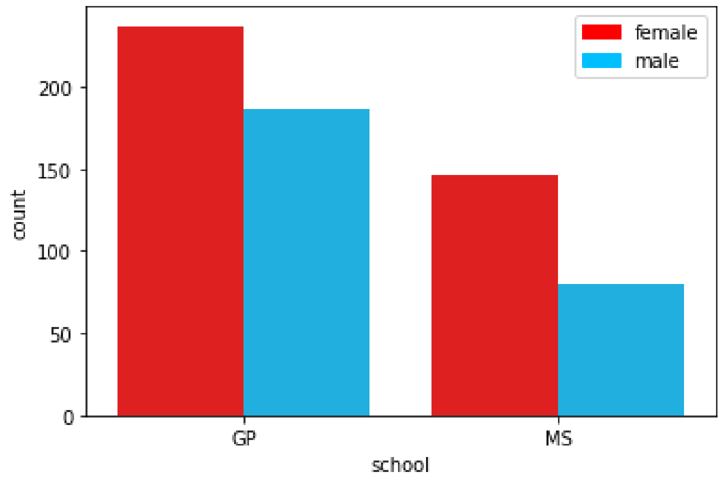
Math class pie chart shows that 52.7% students are female and 47.3% students are male have taken Math as a subject. Math class bar chart shows that around 350 students from Escola Gabriel Pereira and around 50 students from Escola Secundária Mouzinho da Silveira have taken Math class. This also clearly implies that Gabriel Pereira is the bigger school with more students being a part of it. There is also a significant difference, almost close to 300, in the number of students registered for Math class among each school.

Portugal Class Observations:

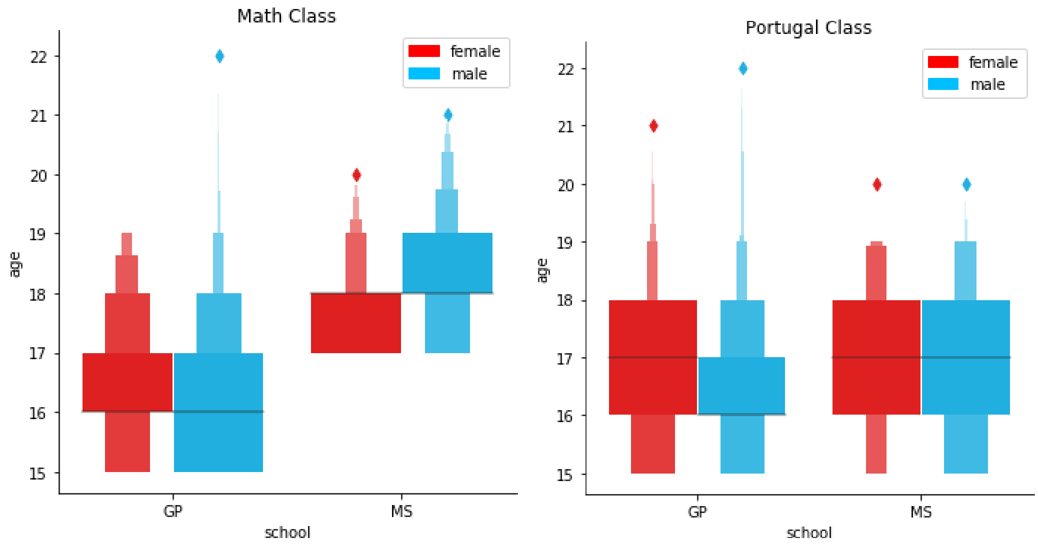
Portugal class pie chart shows that 59% students are female and 41% students are male who have taken Portuguese as a subject. Portugal class bar chart shows that around 420 students are from Escola Gabriel Pereira and 225 students from Escola Secundária Mouzinho da Silveira have taken Portuguese class. Here again it confirms that Gabriel Pereira is the bigger school and in general Portuguese class is enrolled more because it is the native language and students are more confident in enrolling for it.



The above bar graph shows number of female and male students from each school who have taken Math class. From Escola Gabriel Pereira school, around 350 students out of which 175 female and 160 male students. From Escola Secundária Mouzinho da Silveira, around 50 students out of which 26 female and 24 male students. This is a combination of the previous graphs giving us a more accurate visualization

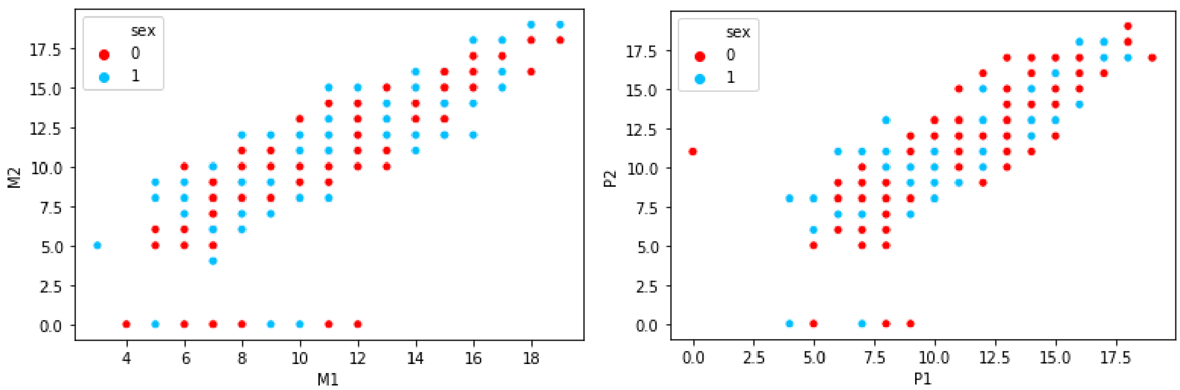


The above bar graph shows number of female and male students from each school who have taken Portuguese class. From Escola Gabriel Pereira school, around 420 students out of which 250 female and 170 male students. From Escola Secundária Mouzinho da Silveira, around 225 students out of which 150 female and 75 male students. This is also a combination of the previous graphs of the Portuguese dataset and there is also a significant increase in the total count since many students enroll for Portuguese as it is their native language. It is also seen that in both Math and Portuguese classes the female students are significantly more than male students.



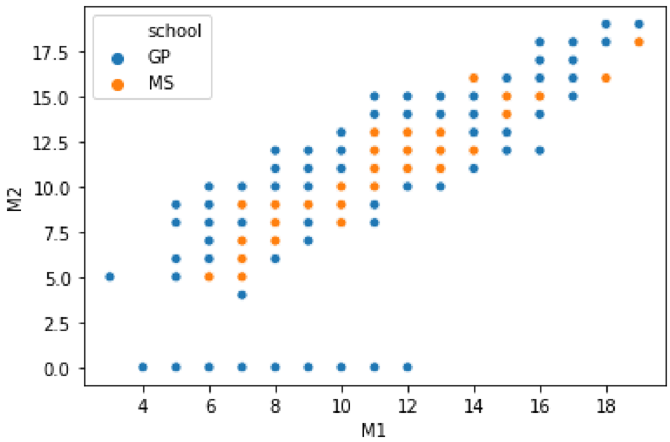
Math class boxen plot indicates that students who have taken Math class in both the schools with comparison to their age. The thickness of the box represents the density of the students in that age interval. Here, we can see that Gabriel Pereira school has a lot of students among the ages of 16 to 17 whereas in the other school we can see that there is more of an elderly population of students among the ages of 17 to 19. We can also see that there are more younger males in Gabriel Pereira whereas the case is reversed in Escola Secundária Mouzinho da Silveira.

Portugal class box plot indicates a very similar age group of students among the ages of 16-18 are more in both the schools. From comparing both the graphs we also see that more of the younger population enroll in the Portuguese class compared to the Math class. We can also see that the eldest in the Gabriel Pereira is a 22-year-old

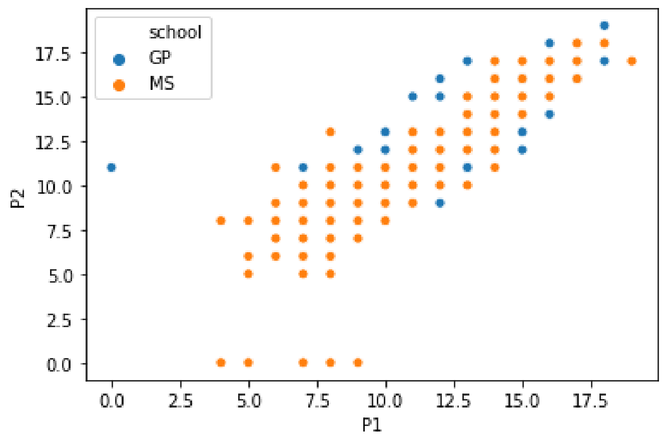


The first scatter represents the marks scored between M1 and M2 with the color scheme legend representing the male and female population. In the math class we see that male students top the class. The students who generally do well in M1 do well in M2 as well as we can see a linear trend while plotting. We can see a few values with either M1 or M2 having a 0. This suggests that either of the class was taken by the student and not both.

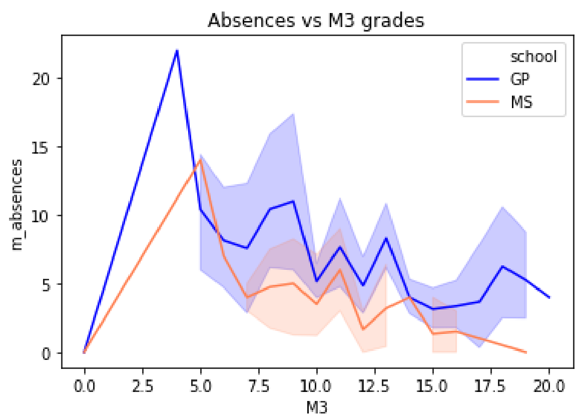
The second scatter plot we see a similar trend of students doing well in both subjects. But we also see that females top the Portuguese class more compared o guys. We also notice a few outliers, similar to the previous graph indicating the students might not have taken one of the subjects.

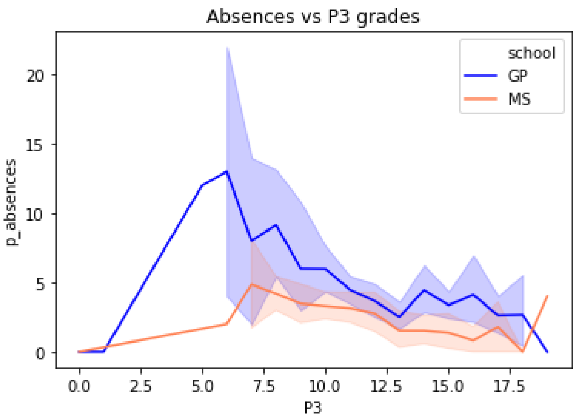


The above scatter plot is the same plot between M1 and M2 but the legend has been changed to represent the different schools on the graphs. It is clearly seen that Gabriel Pereira out performs Escola Secundária Mouzinho da Silveira with majority of the class toppers belonging to the former school. In fact, it is seen that a higher M2 and M1 average is seen among the students of Gabriel Pereira and Escola Secundária Mouzinho da Silveira have students among the mid-range who do decently well in both the subjects.



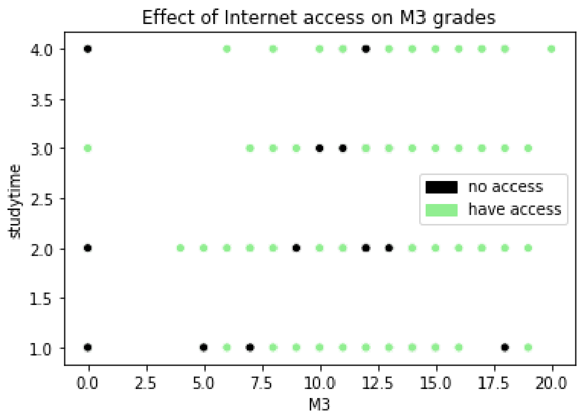
In the P2 vs P1 scatter plot, the schools are used as legends and here we see the Escola Secundária Mouzinho da Silveira has more students doing well in Portuguese as compared to Gabriel Pereira. Although we can see that the topper still belongs to Gabriel Pereira, we can infer from this that the quality of teaching is better in this school since even the topper in the math class belongs to the same school.

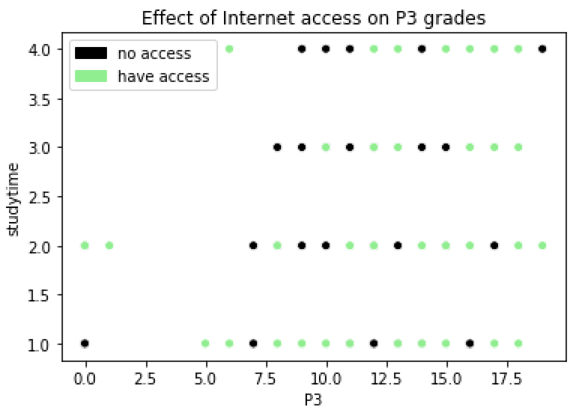




Above Line plots for the both classes indicate the effect of number of absences in class compared to the final grade the students scored. It is clearly seen in both the graphs that as the number of absences decreases the performance of the students increase. This is clearly evident as more involvement in attending classes leads to better understanding of the subject and hence improved grades.

It is also seen that students from Gabriel Pereira end up taking more absences overall compared to students from Escola Secundária Mouzinho da Silveira. This may be because of a more flexible schooling system at Gabriel Pereira where students are allowed to take more holidays.

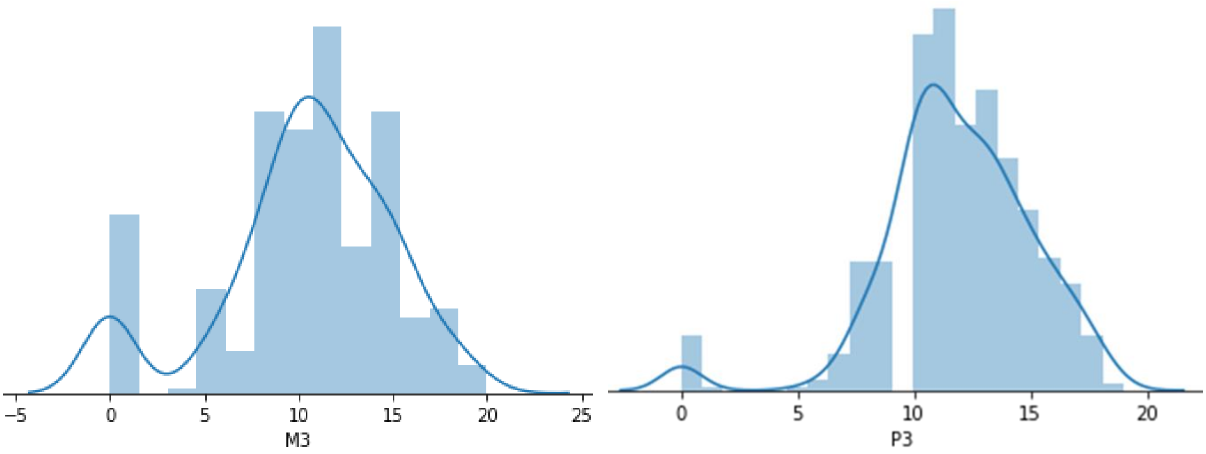




Above Scatter plot graphs are plotted between study time and final grades in Math and Portuguese respectively and color indicates if they have access to internet or not.

First plot shows more students with internet access have performed well in final Math test which proves internet access is very important perform well in Math test. Also, we can see that as the study time increases there is more concentration of higher marks implying that more study-time spend on a subject result in better marks.

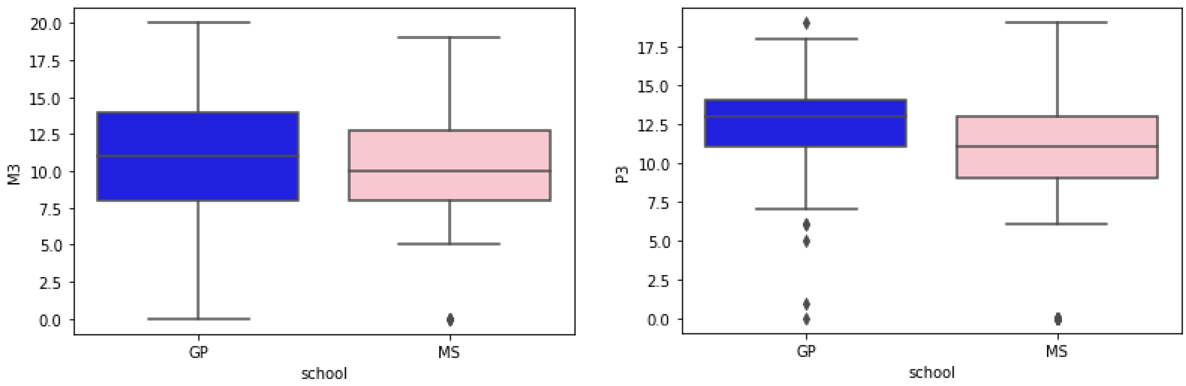
Second plot shows overall students with or without internet access have done equally well in final Portuguese test. There is a concentration of more students without internet access. This is because Portuguese is a native language and it is not essential to have an internet connection to learn more about the subject. However, here also it also seen that there is more concentration of higher marks as the study time increases.



The above graphs indicate the distribution of marks for the final M3 and P3 grades respectively. We can observe that both the graphs follow normal distribution if we ignore a few outliers with the highest concentration of marks in both Math and Portuguese around 10 points.

In both the graphs,

- 34% grades lie between mean and one standard deviation above or below the mean.
- 68% of the grades lies between mean and two standard deviations above or below the mean.
- 99.7% of the grades lies between mean and 3 standard deviations above or below the mean.



Box Plot shows the median, quartiles of final test grades in both Math and Portuguese in Escola Gabriel Pereira and Escola Secundária Mouzinho da Silveira schools.

First box plot shows that median score in final Math test for Escola Gabriel Pereira is around 11 and around 10 for Escola Secundária Mouzinho da Silveira. 75% of the students in both the schools have marks above 7.5 but the higher score in math has been scored by Gabriel Pereira. The interquartile range is more in Gabriel Pereira as compared to Escola Secundária Mouzinho da Silveira.

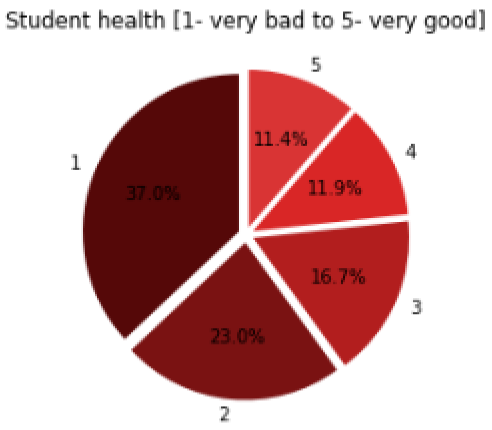
Second plot shows that median score in final Portuguese test for Escola Gabriel Pereira is around 12.7 and around 11 for Escola Secundária Mouzinho da Silveira. It is obvious that they have higher medians since Portuguese is their native language and almost 75% of the students in Gabriel Pereira have scores above 11 and 75% of the students have score above 8.5.

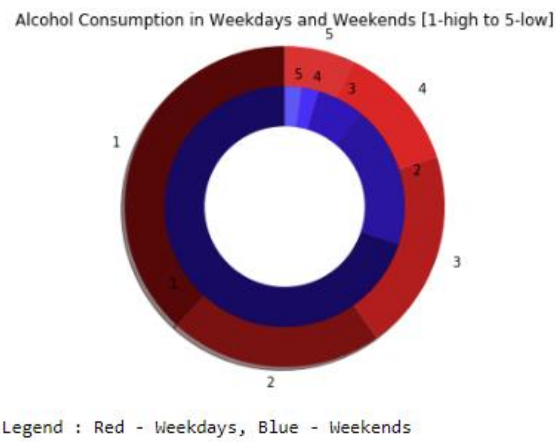
```
count    395.000000
mean      10.415190
std        4.581443
min        0.000000
25%        8.000000
50%       11.000000
75%       14.000000
max       20.000000
Name: M3, dtype: float64
```

```
count    649.000000
mean      11.906009
std        3.230656
min        0.000000
25%       10.000000
50%       12.000000
75%       14.000000
max       19.000000
Name: P3, dtype: float64
```

The above summaries for both the final grades M3 and P3 are listed and seen.

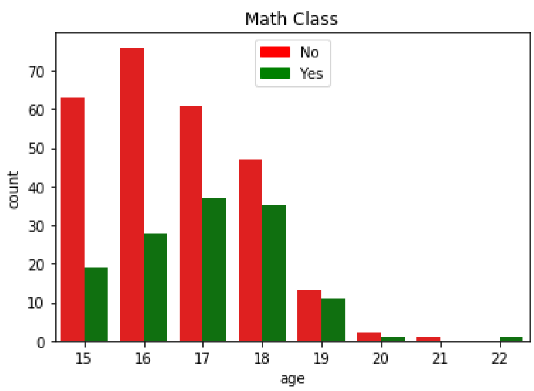
EXPLORATORY DATA ANALYSIS – EXTRA

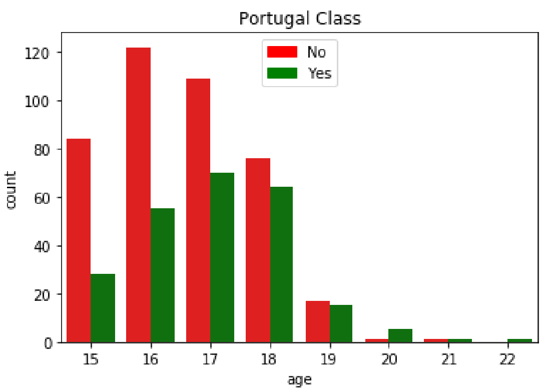




First Plot is a Pie chat with burst option which shows student health as color gradient with percentage of students in each category. More percentage of students have bad health condition and only fewer students are with a good health condition probably reflecting the environment which they stay in.

Second Plot shows students alcohol consumption in weekdays and weekends. The outer circle represents the student’s habits on weekdays and the inner circle represents weekends. Students tend to consume alcohol more during weekends rather than weekdays. This shows that students are more focused during weekdays which is a good sign.





Above bar graphs is a plot between age and number of students with color code as romatic relationship. In Math class and Portuguese class a very similar trend is observed where almost equal number of students from age group 15 to 18 are in a romantic relationships. But students with no romantic relationships are more than the ones who are in a romantic relationships. Also the negative trend as the age increases which indicates that as they grow older they prefer to be in relationships as the number of studnets not in a relationship decreases in both the cases.

MULTIPLE LINEAR REGRESSION

REGRESSION:

In statistical modelling, regression analysis is a set of statistical processes for estimating the relationships between a dependent variable (often called the 'outcome variable') and one or more independent variables (often called 'predictors', 'covariates', or 'features').

MULTIPLE LINEAR REGRESSION:

Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. The goal of multiple linear regression (MLR) is to model the linear relationship between the explanatory (independent) variables and response (dependent) variable.

DATASET FOR REGRESSION MODELS:

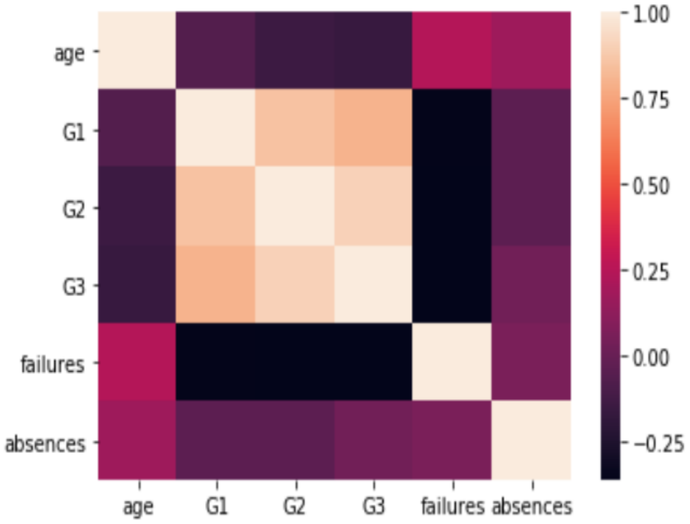
It consists of 33 variables which are continuous, binary and nominal variables. For our multiple linear regression analysis, we have created dummy variables for binary and nominal variables. Thus, dataset to be used consists of 41 independent variables(predictors) and 1 dependent variable.

Out of these 41 variables, we are excluding variables G1(Grade 1) AND G2(Grade 2) in both the Math and Portuguese regression model as they are highly correlated with G3(Grade)

CHECKING CORRELATION BETWEEN NUMERIC VARIABLES:

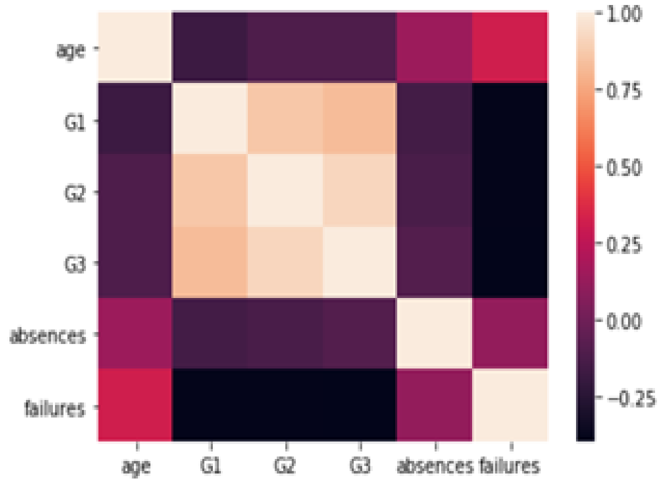
MATH DATASET:

Out[44]: <matplotlib.axes._subplots.AxesSubplot at 0x125844f4710>



PORTUGUESE DATASET:

Out[55]: <matplotlib.axes._subplots.AxesSubplot at 0x22db269f2e8>



MODEL 1

MATH REGRESSION MODEL:

In the Math Regression model, we have used 39 variables as predictors and 1 variable (G3) as the target variable.

RESULTS:

OLS Regression Results

Dep. Variable:	G3	**R-squared:**	0.276
Model:	OLS	**Adj. R-squared:**	0.196
Method:	Least Squares	**F-statistic:**	3.463
Date:	Mon, 04 Nov 2019	**Prob (F-statistic):**	3.32e-10
Time:	22:44:51	**Log-Likelihood:**	-1097.5
No. Observations:	395	**AIC:**	2275.
Df Residuals:	355	**BIC:**	2434.
Df Model:	39		
Covariance Type:	nonrobust		

SIGNIFICANT VARIABLES:

We have considered variables to be significant whose p values is less than 0.05(95% confidence level)

	Coefficient	**p value**
Intercept	14.077	0.002
Failures	-1.72	0.0
Goout	-0.59	0.009
Sex_M	1.26	0.012
Schoolsup_yes	-1.35	0.044
Romantic_yes	-1.09	0.020

INTERPRETATION:

Failures: For every increase in failure by 1 unit, there will be an average decrease of 1.72 points in G3(Grade 3)

Goout: For every increase in Goout by 1 unit, there will be an average decrease of 0.59 points in G3(Grade 3)

Sex_M: Compared to Females, on an average, Males will score 1.26 points higher in G3(Grade 3) than females

Schoolsup_yes: Compared to students who had no education support, on an average, students with education support will score 1.35 points lesser in G3(Grade 3)

Romantic_yes: Compared to students who were not in relationship, on an average, students who were in relationship will score 1.09 points lesser in G3(Grade 3)

MODEL 2

MATH MODEL WITH INTERACTION TERMS:

In this model, we included interaction terms to assess the interaction effects along with the main effects. This model consists of variables which were significant in the previous Math model.

RESULTS:

OLS Regression Results

Dep. Variable:	G3	**R-squared:**	0.197
Model:	OLS	**Adj. R-squared:**	0.166
Method:	Least Squares	**F-statistic:**	6.210
Date:	Wed, 27 Nov 2019	**Prob (F-statistic):**	9.59e-12
Time:	16:54:45	**Log-Likelihood:**	-1117.8
No. Observations:	395	**AIC:**	2268.
Df Residuals:	379	**BIC:**	2331.
Df Model:	15		
Covariance Type:	nonrobust		

SIGNIFICANT VARIABLES:

	Coefficient	**p value**
Intercept	12.80	0
sex_M	1.18	0.377
failures	**-2.99**	**0.001**
failures*sex_M	**-1.22**	**0.04**
failures*schoolsup_yes	**2.05**	**0.02**
Schoolsup_yes	- 0.49	0.79

INTERPRETATION:

It is seen that there are no main effects of variable ‘Schoolsup_yes’ and ‘sex_M’ on G3, but they both are significant when it interacts with ‘Failures’ variable

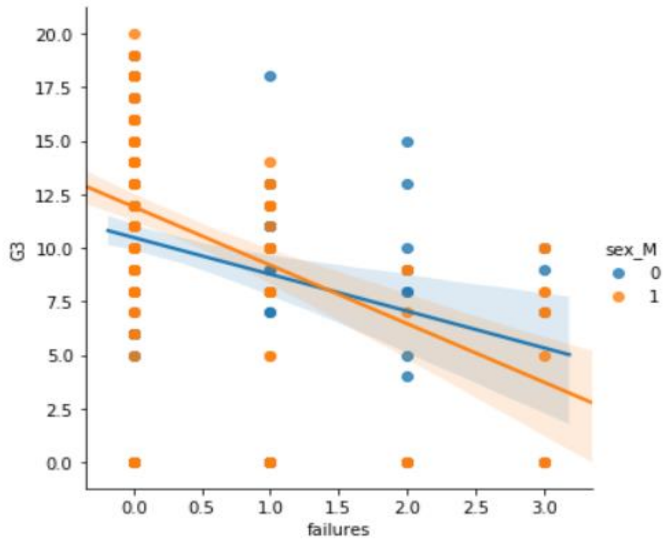
Failures: For every increase in failure by 1 unit, there will be an average decrease of 2.99 points in G3(Grade 3)

Failures*sex_M: For every increase in failure, females will score additional (-2.99), while males will score additional ‘-4.21’ points (-2.99-1.22). Thus, females who have failed before will score higher than Males who have failed before.

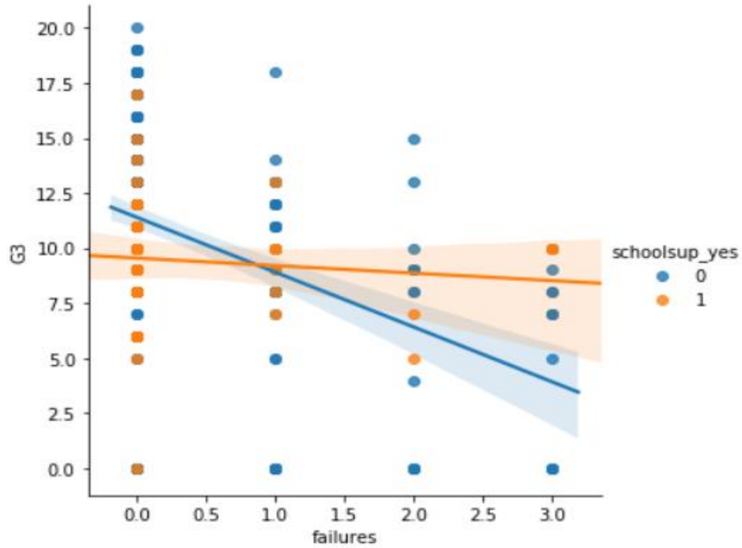
Failures * schoolsup_yes: For every increase in failure, students with no education support will score additional ‘less marks’ (-2.99), while a student with education support will score additional ‘-0.94’ points (-2.99 + 2.05). Thus, students who have failed before and have received education support will score better than students who did not receive education support

VISUALIZATION OF INTERACTION EFFECTS

Failures*sex_M:



Failures * schoolsup_yes:



MODEL 3

PORTUGUESE REGRESSION MODEL:

In the Portuguese Regression model, we have used 39 variables as predictors and 1 variable (G3) as the target variable.

RESULTS:

OLS Regression Results

Dep. Variable:	G3	**R-squared:**	0.360
Model:	OLS	**Adj. R-squared:**	0.319
Method:	Least Squares	**F-statistic:**	8.797
Date:	Mon, 04 Nov 2019	**Prob (F-statistic):**	3.27e-38
Time:	13:26:10	**Log-Likelihood:**	-1536.5
No. Observations:	649	**AIC:**	3153.
Df Residuals:	609	**BIC:**	3332.
Df Model:	39		
Covariance Type:	nonrobust		

SIGNIFICANT VARIABLES:

	Coefficient	**p value**
Intercept	8.68	0
Studytime	0.4	0.004
Failures	-1.41	0.00
Health	- 0.1874	0.015
Schools_MS	-1.2	0.00
Sex_M	- 0.63	0.012
Schoolsup_yes	- 1.31	0.00
Higher_yes	1.733	0.00

INTERPRETATION:

Studytime: For every increase in studytime by 1 unit, there will be an average increase of 0.4 points in G3(Grade 3)

Failures: For every increase in failure by 1 unit, there will be an average decrease of 1.41 points in G3(Grade 3)

Health: For every increase in health by 1 unit (as it improves), there will be an average decrease of 0.1874 points in G3(Grade 3)

School_MS: Compared to students who attended School_GP, Students who attended School_MS will score, on an average of 1.2 points lesser in G3

Sex_M: Compared to Females, on an average, Males will score 0.63 points lesser in G3 than females

Schoolsup_yes: Compared to students who had no education support, on an average, students with education support scored 1.31 points lesser in G3

Higher_yes: Compared to students who do not want to take higher educations, students who want to take higher education will score, on an average of 1.733 points higher in G3

MODEL 4

PORTUGUESE MODEL WITH INTERACTION TERMS:

In this model, we included interaction terms to assess the interaction effects along with the main effects. This model consists of variables which were significant in the previous Portuguese model

RESULTS:

OLS Regression Results

Dep. Variable:	G3	**R-squared:**	0.331
Model:	OLS	**Adj. R-squared:**	0.302
Method:	Least Squares	**F-statistic:**	11.37
Date:	Mon, 04 Nov 2019	**Prob (F-statistic):**	7.88e-39
Time:	18:54:25	**Log-Likelihood:**	-1551.1
No. Observations:	649	**AIC:**	3158.
Df Residuals:	621	**BIC:**	3284.
Df Model:	27		
Covariance Type:	nonrobust		

SIGNIFICANT VARIABLES:

It is seen that there is no main effect of variable 'Schoolsup_yes' on G3, but it is significant when it interacts with 'Failures' variable

	Coefficient	p value
Intercept	**9.62**	**0.00**
Failures	**-1.93**	**0.03**
Higher_yes	**3.66**	**0.02**
Schoolsup_yes	-1.53	0.3
failures*schoolsup_yes	**1.56**	**0.013**

INTERPRETATION:

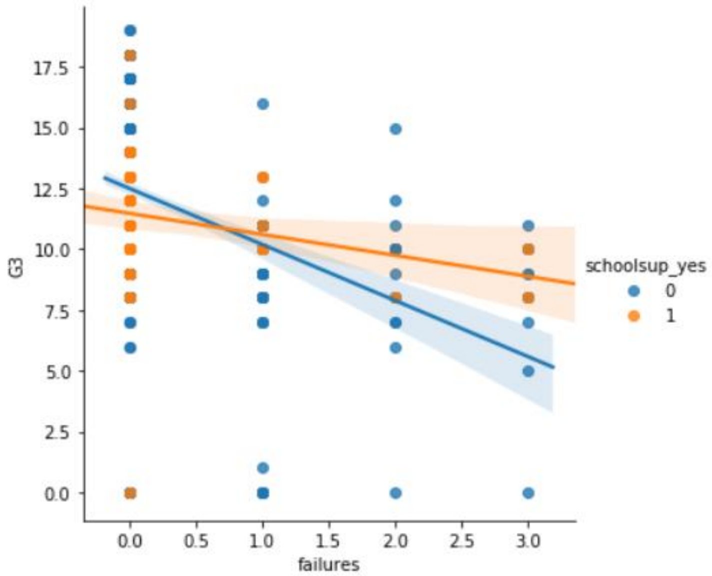
Failures: For every increase in failure by 1 unit, there will be an average decrease of 1.93 points in G3(Grade 3)

Higher_yes: Compared to students who do not want to take higher educations, students who wants to take higher education will score, on an average of 3.66 points higher in G3

Failures*schoolsup_yes: For every increase in failure, student with no education support will score additional 'less marks' (-1.93), while a student with education support will score additional '-0.37' points (-1.93+1.56) in G3. Thus, students who have failed before and have received education support will score better than students who did not receive education support

VISUALIZATION OF INTERACTION EFFECTS:

Failures*schoolsup_yes:



PREDICTIVE MODELLING WITH NEURAL NETWORKS

Models

Four models were created to predict if a student would pass of fail a class using the features present in the dataset. These models include a binary classification neural network, a decision tree optimized by cross-validated grid search, a random forest optimized by cross-validated grid search, and a logistic regression model.

Model Data Preparation

One-hot encoding was employed to handle categorical variables with no ordinality. The process of one-hot encoding consists of transforming a categorical value into a group of bits in which there is only a single high value. In other words, the binary categorical variable "sex" possesses 1 for male and 0 for female. One-hot encoding this variable would transform female to [1,0] and male to [0,1]. This process was carried out for each categorical variable with no ordinality such as sex, paid, higher, internet, and romantic. Categorical variables which do show ordinality such as mother's education and father's education were not one-hot-encoded to preserve their properties.

Labels were created using the final grade each student achieved in the course. In Portugal, grades range from 0-20. A grade below 9.5 is considered a failing grade and a grade above 9.5 is passing.

Handling Skewed Classes

Since more students passed their courses, the math and Portuguese data was skewed. 33% of students failed math and 15% of students failed Portuguese. Skewed labels have a detrimental effect on the performance of neural networks and other algorithms. Therefore, the minority classes were up sampled to produce two equal size classes. In other words, 50% of the data held the label "pass" and the other 50% of the data held the label "fail."

Feature Selection

A combination of demographic, behavioural, and socioeconomic features were selected for use in the prediction models. The ordinal variables included mother's and father's education levels, how frequently the student goes out with friends, weekday alcohol consumption levels, and weekend alcohol consumption levels. One-hot encoded features (non-ordinal) include whether or not students had extra paid coursework, whether or not students had hopes of attending university, whether or not students were in a romantic relationship, and whether or not they passed the first grading period. Finally, the two continuous variables included were number of past failures and the student's grade from the first grading period of the year. These features were selected using insights obtained from regressions and through trial and error to most effectively predict student performance.

Metrics

True Positives – model identifies a failing student as a failing student.
False Positives – model identifies a passing student as a failing student.
False Negatives – model identifies a failing student as a passing student.
True Negatives – model identifies a passing student as a passing student.

A combination of metrics was utilized for each model to most effectively measure a model's performance. Accuracy is the fraction of predictions the model was able to correctly classify. In other words, accuracy is the proportion of students the model correctly predicted would pass or fail. Precision is the proportion of positive identifications (students failing) made by the model that were actually correct. Precision can be thought of as a measure of how frequently a false positive is predicted. Recall is the proportion of failing students that were identified correctly by the model. In this project, recall is the most vital metric because the cost of a student's failure is high for both the student and the school. Incorrectly classifying a few passing students as failing does not carry as high of a cost. Therefore, the models created for this project will be judged heavily on how effectively failing students are identified.

Neural Network

A neural network using TensorFlow's high level API, keras, was built to predict whether a student would pass or fail their course. This binary classification neural network is composed of an input layer, 2 hidden layers, and an output layer. The two hidden layers hold 128 nodes each. The overall shape of the network was determined through a trial and error process. Only 1 hidden layer led to underfitting the data because the network was too simple and more than 2 hidden layers showed no improvement in performance. Similarly, fewer than 128 nodes in each hidden layer underfit the training data and greater than 128 nodes did not improve performance substantially. The 2 hidden layers and the output layer all shared a sigmoid activation function which is responsible for squishing the node values into a number between 0 and 1 to allow the network to more efficiently work with the data. The sigmoid activation function is the standard activation function used for binary classification tasks. Finally, the training data was passed through the neural network 36 times during the training phase to allow the network to better learn the patterns present in the data. All reported metrics are average values of 5 different random state iterations.

The neural network's performance metrics are summarized in the following table:

MATH						
Trial	1	2	3	4	5	Avg
Precision	0.824	0.836	0.838	0.866	0.827	0.838
Recall	0.924	0.849	0.864	0.891	0.939	0.893
Accuracy	0.865	0.842	0.850	0.876	0.872	0.861
F1	0.871	0.842	0.851	0.879	0.879	0.864

PORTUGUESE						
Trial	1	2	3	4	5	Avg
Precision	0.866	0.876	0.903	0.846	0.823	0.863
Recall	0.891	0.971	0.942	0.957	0.978	0.948
Accuracy	0.876	0.916	0.920	0.891	0.884	0.897
F1	0.879	0.921	0.922	0.898	0.894	0.903

Decision Tree with Cross Validated Grid Search

A binary classification decision tree was created to predict whether a student would pass or fail their course. Hyperparameters were determined through sklearn's grid search algorithm. The best hyperparameters for the math and Portuguese decision trees were identical with a max depth of 1, 2 max leaf nodes, and a minimum of 2 samples splits. These parameters stayed constant through every iteration. The splitting measure Gini Index was used in this model because it works best with a binary task.

The decision tree's performance metrics are summarized in the following table:

MATH						
Trial	1	2	3	4	5	Avg
Precision	0.805	0.765	0.747	0.798	0.805	0.784
Recall	0.939	0.939	0.939	0.955	0.985	0.952
Accuracy	0.857	0.827	0.812	0.857	0.872	0.845
F1	0.867	0.844	0.832	0.869	0.886	0.860

PORTUGUESE						
Trial	1	2	3	4	5	Avg
Precision	0.848	0.858	0.838	0.761	0.867	0.834
Recall	0.848	0.927	0.942	1.000	0.854	0.914
Accuracy	0.847	0.887	0.880	0.844	0.862	0.864
F1	0.848	0.891	0.887	0.864	0.860	0.870

Random Forest (Tuned)

A random forest is an ensemble classifier made of a large amount of individual decision trees. Each individual tree classifies a student as passing and failing using the patterns in the training data and the class with the greatest number of votes becomes the model's output. A random forest is a powerful model because a large number of uncorrelated trees operating together outperform a single decision tree. The random forest's hyperparameters were determined through sklearn's grid search algorithm. The hyperparameters did not stay consistent through the 5 trials. Again, the splitting measure Gini Index was used in this model because it works best with a binary task. 100 different trees made up the random forest.

The random forest's performance metrics are summarized in the following table:

MATH						
Trial	1	2	3	4	5	Avg
Precision	0.864	0.853	0.857	0.781	0.773	0.825
Recall	0.851	0.879	0.818	0.970	0.879	0.879
Accuracy	0.857	0.865	0.842	0.850	0.812	0.845
F1	0.857	0.866	0.837	0.865	0.823	0.850

PORTUGUESE						
Trial	1	2	3	4	5	Avg
Precision	0.866	0.838	0.862	0.857	0.843	0.853
Recall	0.898	0.905	0.906	0.876	0.855	0.888
Accuracy	0.880	0.866	0.880	0.866	0.847	0.868
F1	0.882	0.870	0.883	0.866	0.849	0.870

Logistic Regression

Logistic regression is a popular supervised learning method for tackling a classification problem. It is predominantly used to solve binary prediction tasks because the output of a logistic regression is the probability of a specific event occurring. With respect to this project, the logistic regression model will determine the probability of a student failing their course given a set of independent variables.

The logistic regression's performance metrics are summarized in the following table:

MATH						
Trial	1	2	3	4	5	Avg
Precision	0.778	0.857	0.807	0.844	0.831	0.823
Recall	0.849	0.818	0.746	0.818	0.818	0.810
Accuracy	0.805	0.842	0.782	0.835	0.827	0.818
F1	0.812	0.837	0.775	0.831	0.824	0.816

PORTUGUESE						
Trial	1	2	3	4	5	Avg
Precision	0.843	0.890	0.868	0.887	0.887	0.875
Recall	0.855	0.877	0.906	0.906	0.913	0.891
Accuracy	0.847	0.884	0.884	0.895	0.898	0.881
F1	0.849	0.883	0.887	0.896	0.900	0.883

Model Comparison

Recall is the proportion of failing students that were identified correctly by the model. This is the most important metric to use when comparing all the classification models that were built because the cost of missing a failing student is high. A failing student would not receive the extra attention they require if the model classifies them as passing. These students would be forced to repeat the course which could be costly to the Portuguese school system. The cost to the students would also be high in terms because they would fall behind their peers. The model with the highest average recall value when using the math dataset is the decision tree with tuned hyperparameters. The model with the highest average recall value when using the Portuguese dataset is the dense neural network. The best model to use to classify students in the math course is the decision tree with tuned hyperparameters. The best model to use to classify students in the Portuguese course is the dense neural network.

In [1]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

In [2]:
```
import pandas as pd
import keras.utils as keras
from sklearn import preprocessing
import numpy as np
```

```
Using TensorFlow backend.
```

In [3]:
```
## Read data from csv file 'student-mat.csv'
math_data = pd.read_csv('student-mat.csv', sep=';')

## Read data from csv file 'student-por.csv'
por_data = pd.read_csv('student-por.csv', sep=';')

## Two datasets are similar except for the final 4 columns.
math_data.head()
por_data.head()

## shape
math_data.shape
por_data.shape
```

Out[3]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

Out[3]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

Out[3]: (395, 33)

Out[3]: (649, 33)

In [4]:
```
## Rename columns.
math_data = math_data.rename(columns= {'absences': 'm_absences',
                                       'G1': 'M1',
                                       'G2': 'M2',
                                       'G3': 'M3'})


por_data = por_data.rename(columns= {'absences': 'p_absences',
                                     'G1': 'P1',
                                     'G2': 'P2',
                                     'G3': 'P3'})
```

In [5]:
```
math_data.head()
por_data.head()
```

Out[5]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

Out[5]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

In [6]:
```
math_data['school'].value_counts()
por_data['school'].value_counts()
```

Out[6]:
```
GP    349
MS     46
Name: school, dtype: int64
```

Out[6]:
```
GP    423
MS    226
Name: school, dtype: int64
```

In [22]:
```
math_data['Mjob'].value_counts()
math_data['Fjob'].value_counts()
por_data['Mjob'].value_counts()
por_data['Fjob'].value_counts()
```

Out[22]:
```
other       141
services    103
at_home      59
teacher      58
health       34
Name: Mjob, dtype: int64
```

Out[22]:
```
other       217
services    111
teacher      29
at_home      20
health       18
Name: Fjob, dtype: int64
```

Out[22]:
```
other       258
services    136
at_home     135
teacher      72
health       48
Name: Mjob, dtype: int64
```

Out[22]:
```
other       367
services    181
at_home      42
teacher      36
health       23
Name: Fjob, dtype: int64
```

In [8]:
```
## Missing data check.
math_missing_data = math_data.isnull().sum()
print(math_missing_data)
## No missing data in math_data.

print()

por_missing_data = por_data.isnull().sum()
print(por_missing_data)
## No missing data in por_data
```

```
school        0
sex           0
age           0
address       0
famsize       0
Pstatus       0
Medu          0
Fedu          0
Mjob          0
Fjob          0
reason        0
guardian      0
traveltime    0
studytime     0
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
m_absences    0
M1            0
M2            0
M3            0
dtype: int64

school        0
sex           0
age           0
address       0
famsize       0
Pstatus       0
Medu          0
Fedu          0
Mjob          0
Fjob          0
reason        0
guardian      0
traveltime    0
studytime     0
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
p_absences    0
P1            0
P2            0
P3            0
dtype: int64
```

In [9]:

```
## Label Encode Sex. 0 = female, 1 = male
le_math = preprocessing.LabelEncoder()
le_math.fit(math_data['sex'])
math_sex_array = le_math.transform(math_data['sex'])
math_data['sex'] = math_sex_array
math_data.head()

le_por = preprocessing.LabelEncoder()
le_por.fit(por_data['sex'])
por_sex_array = le_por.transform(por_data['sex'])
por_data['sex'] = por_sex_array
por_data.head()
```

Out[9]: LabelEncoder()

Out[9]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	0	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	0	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	0	15	U	GT3	T	4	2	health	services	...	3	
4	GP	0	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

Out[9]: LabelEncoder()

Out[9]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	0	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	0	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	0	15	U	GT3	T	4	2	health	services	...	3	
4	GP	0	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

In [10]:
```
## Label Encode Parental Status. 0 = apart 1 = together
le_math_Pstatus = preprocessing.LabelEncoder()
le_math_Pstatus.fit(math_data['Pstatus'])
math_Pstatus_array = le_math_Pstatus.transform(math_data['Pstatus'])
math_data['Pstatus'] = math_Pstatus_array
math_data.head()

le_por_Pstatus = preprocessing.LabelEncoder()
le_por_Pstatus.fit(por_data['Pstatus'])
por_Pstatus_array = le_por_Pstatus.transform(por_data['Pstatus'])
por_data['Pstatus'] = por_Pstatus_array
por_data.head()
```

Out[10]: LabelEncoder()

Out[10]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	U	GT3	0	4	4	at_home	teacher	...	4	
1	GP	0	17	U	GT3	1	1	1	at_home	other	...	5	
2	GP	0	15	U	LE3	1	1	1	at_home	other	...	4	
3	GP	0	15	U	GT3	1	4	2	health	services	...	3	
4	GP	0	16	U	GT3	1	3	3	other	other	...	4	

5 rows × 33 columns

Out[10]: LabelEncoder()

Out[10]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	U	GT3	0	4	4	at_home	teacher	...	4	
1	GP	0	17	U	GT3	1	1	1	at_home	other	...	5	
2	GP	0	15	U	LE3	1	1	1	at_home	other	...	4	
3	GP	0	15	U	GT3	1	4	2	health	services	...	3	
4	GP	0	16	U	GT3	1	3	3	other	other	...	4	

5 rows × 33 columns

In [11]:
```python
## Label Encode Address (Urban or Rural). 1 = Urban 0 = Rural
le_math_address = preprocessing.LabelEncoder()
le_math_address.fit(math_data['address'])
math_address_array = le_math_address.transform(math_data['address'])
math_data['address'] = math_address_array
math_data.head()

le_por_address = preprocessing.LabelEncoder()
le_por_address.fit(por_data['address'])
por_address_array = le_por_address.transform(por_data['address'])
por_data['address'] = por_address_array
por_data.head()
```

Out[11]: LabelEncoder()

Out[11]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	1	GT3	0	4	4	at_home	teacher	...	4	
1	GP	0	17	1	GT3	1	1	1	at_home	other	...	5	
2	GP	0	15	1	LE3	1	1	1	at_home	other	...	4	
3	GP	0	15	1	GT3	1	4	2	health	services	...	3	
4	GP	0	16	1	GT3	1	3	3	other	other	...	4	

5 rows × 33 columns

Out[11]: LabelEncoder()

Out[11]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	1	GT3	0	4	4	at_home	teacher	...	4	
1	GP	0	17	1	GT3	1	1	1	at_home	other	...	5	
2	GP	0	15	1	LE3	1	1	1	at_home	other	...	4	
3	GP	0	15	1	GT3	1	4	2	health	services	...	3	
4	GP	0	16	1	GT3	1	3	3	other	other	...	4	

5 rows × 33 columns

In [12]:
```
## Label Encode schoolsup (extra educational support) 1 = yes, 0 = no.
le_math_schoolsup = preprocessing.LabelEncoder()
le_math_schoolsup.fit(math_data['schoolsup'])
math_schoolsup_array = le_math_schoolsup.transform(math_data['schoolsup'])
math_data['schoolsup'] = math_schoolsup_array
math_data['schoolsup'].head()

le_por_schoolsup = preprocessing.LabelEncoder()
le_por_schoolsup.fit(por_data['schoolsup'])
por_schoolsup_array = le_por_schoolsup.transform(por_data['schoolsup'])
por_data['schoolsup'] = por_schoolsup_array
por_data['schoolsup'].head()
```

Out[12]:
```
LabelEncoder()
```

Out[12]:
```
0    1
1    0
2    1
3    0
4    0
Name: schoolsup, dtype: int64
```

Out[12]:
```
LabelEncoder()
```

Out[12]:
```
0    1
1    0
2    1
3    0
4    0
Name: schoolsup, dtype: int64
```

In [13]:
```
## Label Encode famsup (Family Support) 1 = yes, 0 = no
le_math_famsup = preprocessing.LabelEncoder()
le_math_famsup.fit(math_data['famsup'])
math_famsup_array = le_math_famsup.transform(math_data['famsup'])
math_data['famsup'] = math_famsup_array
math_data['famsup'].head()

le_por_famsup = preprocessing.LabelEncoder()
le_por_famsup.fit(por_data['famsup'])
por_famsup_array = le_por_famsup.transform(por_data['famsup'])
por_data['famsup'] = por_famsup_array
por_data['famsup'].head()
```

Out[13]:
```
LabelEncoder()
```

Out[13]:
```
0    0
1    1
2    0
3    1
4    1
Name: famsup, dtype: int64
```

Out[13]:
```
LabelEncoder()
```

Out[13]:
```
0    0
1    1
2    0
3    1
4    1
Name: famsup, dtype: int64
```

In [14]:
```
## Label Encode paid (extra paid classes within the course subject (Math or Po
rtuguese)) 1 = yes, 0 = no
le_math_paid = preprocessing.LabelEncoder()
le_math_paid.fit(math_data['paid'])
math_paid_array = le_math_paid.transform(math_data['paid'])
math_data['paid'] = math_paid_array
math_data['paid'].head()

le_por_paid= preprocessing.LabelEncoder()
le_por_paid.fit(por_data['paid'])
por_paid_array = le_por_paid.transform(por_data['paid'])
por_data['paid'] = por_paid_array
por_data['paid'].head()
```

Out[14]:
```
LabelEncoder()
```

Out[14]:
```
0    0
1    0
2    1
3    1
4    1
Name: paid, dtype: int64
```

Out[14]:
```
LabelEncoder()
```

Out[14]:
```
0    0
1    0
2    0
3    0
4    0
Name: paid, dtype: int64
```

In [15]:
```
## Label Encode activities (extra-curricular activities) 1 = yes, 0 = no
le_math_activities = preprocessing.LabelEncoder()
le_math_activities.fit(math_data['activities'])
math_activities_array = le_math_activities.transform(math_data['activities'])
math_data['activities'] = math_activities_array
math_data['activities'].head()

le_por_activities= preprocessing.LabelEncoder()
le_por_activities.fit(por_data['activities'])
por_activities_array = le_por_activities.transform(por_data['activities'])
por_data['activities'] = por_activities_array
por_data['activities'].head()
```

Out[15]:
```
LabelEncoder()
```

Out[15]:
```
0    0
1    0
2    0
3    1
4    0
Name: activities, dtype: int64
```

Out[15]:
```
LabelEncoder()
```

Out[15]:
```
0    0
1    0
2    0
3    1
4    0
Name: activities, dtype: int64
```

In [16]:

```
## Label Encode nursery (attended nursery school) 1 = yes, 0 = no
le_math_nursery = preprocessing.LabelEncoder()
le_math_nursery.fit(math_data['nursery'])
math_nursery_array = le_math_nursery.transform(math_data['nursery'])
math_data['nursery'] = math_nursery_array
math_data['nursery'].head()

le_por_nursery= preprocessing.LabelEncoder()
le_por_nursery.fit(por_data['nursery'])
por_nursery_array = le_por_nursery.transform(por_data['nursery'])
por_data['nursery'] = por_nursery_array
por_data['nursery'].head()
```

Out[16]:

```
LabelEncoder()
```

Out[16]:

```
0    1
1    0
2    1
3    1
4    1
Name: nursery, dtype: int64
```

Out[16]:

```
LabelEncoder()
```

Out[16]:

```
0    1
1    0
2    1
3    1
4    1
Name: nursery, dtype: int64
```

In [17]:
```
## Label Encode higher (wants to take higher education) 1 = yes, 0 = no
le_math_higher = preprocessing.LabelEncoder()
le_math_higher.fit(math_data['higher'])
math_higher_array = le_math_higher.transform(math_data['higher'])
math_data['higher'] = math_higher_array
math_data['higher'].head()

le_por_higher= preprocessing.LabelEncoder()
le_por_higher.fit(por_data['higher'])
por_higher_array = le_por_higher.transform(por_data['higher'])
por_data['higher'] = por_higher_array
por_data['higher'].head()
```

Out[17]:
```
LabelEncoder()
```

Out[17]:
```
0    1
1    1
2    1
3    1
4    1
Name: higher, dtype: int64
```

Out[17]:
```
LabelEncoder()
```

Out[17]:
```
0    1
1    1
2    1
3    1
4    1
Name: higher, dtype: int64
```

In [18]:
```
## Label Encode internet (Internet access at home) 1 = yes, 2 = no
le_math_internet = preprocessing.LabelEncoder()
le_math_internet.fit(math_data['internet'])
math_internet_array = le_math_internet.transform(math_data['internet'])
math_data['internet'] = math_internet_array
math_data['internet'].head()

le_por_internet= preprocessing.LabelEncoder()
le_por_internet.fit(por_data['internet'])
por_internet_array = le_por_internet.transform(por_data['internet'])
por_data['internet'] = por_internet_array
por_data['internet'].head()
```

Out[18]:
```
LabelEncoder()
```

Out[18]:
```
0    0
1    1
2    1
3    1
4    0
Name: internet, dtype: int64
```

Out[18]:
```
LabelEncoder()
```

Out[18]:
```
0    0
1    1
2    1
3    1
4    0
Name: internet, dtype: int64
```

In [19]:
```
## Label Encode romantic (in a romantic relationship) 1 = yes, 2 = no
le_math_romantic = preprocessing.LabelEncoder()
le_math_romantic.fit(math_data['romantic'])
math_romantic_array = le_math_romantic.transform(math_data['romantic'])
math_data['romantic'] = math_romantic_array
math_data['romantic'].head()

le_por_romantic= preprocessing.LabelEncoder()
le_por_romantic.fit(por_data['romantic'])
por_romantic_array = le_por_romantic.transform(por_data['romantic'])
por_data['romantic'] = por_romantic_array
por_data['romantic'].head()
```

Out[19]:
```
LabelEncoder()
```

Out[19]:
```
0    0
1    0
2    0
3    1
4    0
Name: romantic, dtype: int64
```

Out[19]:
```
LabelEncoder()
```

Out[19]:
```
0    0
1    0
2    0
3    1
4    0
Name: romantic, dtype: int64
```

In [21]:
```
## Saving the new encoded datset as a CSV
export_math = math_data.to_csv('encoded_math_data.csv')
export_por = por_data.to_csv('encoded_por_data.csv')
```

In [ ]:
```

```

In [7]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
math=pd.read_csv('encoded_math_data.csv',sep=',')
por=pd.read_csv('encoded_por_data.csv',sep=',')
```

In [8]:
```
math
```

Out[8]:

	Unnamed: 0	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	...	famrel
0	0	GP	0	18	1	GT3	0	4	4	at_home	...	4
1	1	GP	0	17	1	GT3	1	1	1	at_home	...	5
2	2	GP	0	15	1	LE3	1	1	1	at_home	...	4
3	3	GP	0	15	1	GT3	1	4	2	health	...	3
4	4	GP	0	16	1	GT3	1	3	3	other	...	4
5	5	GP	1	16	1	LE3	1	4	3	services	...	5
6	6	GP	1	16	1	LE3	1	2	2	other	...	4
7	7	GP	0	17	1	GT3	0	4	4	other	...	4
8	8	GP	1	15	1	LE3	0	3	2	services	...	4
9	9	GP	1	15	1	GT3	1	3	4	other	...	5
10	10	GP	0	15	1	GT3	1	4	4	teacher	...	3
11	11	GP	0	15	1	GT3	1	2	1	services	...	5
12	12	GP	1	15	1	LE3	1	4	4	health	...	4
13	13	GP	1	15	1	GT3	1	4	3	teacher	...	5
14	14	GP	1	15	1	GT3	0	2	2	other	...	4
15	15	GP	0	16	1	GT3	1	4	4	health	...	4
16	16	GP	0	16	1	GT3	1	4	4	services	...	3
17	17	GP	0	16	1	GT3	1	3	3	other	...	5
18	18	GP	1	17	1	GT3	1	3	2	services	...	5
19	19	GP	1	16	1	LE3	1	4	3	health	...	3
20	20	GP	1	15	1	GT3	1	4	3	teacher	...	4
21	21	GP	1	15	1	GT3	1	4	4	health	...	5
22	22	GP	1	16	1	LE3	1	4	2	teacher	...	4
23	23	GP	1	16	1	LE3	1	2	2	other	...	5
24	24	GP	0	15	0	GT3	1	2	4	services	...	4
25	25	GP	0	16	1	GT3	1	2	2	services	...	1
26	26	GP	1	15	1	GT3	1	2	2	other	...	4
27	27	GP	1	15	1	GT3	1	4	2	health	...	2
28	28	GP	1	16	1	LE3	0	3	4	services	...	5
29	29	GP	1	16	1	GT3	1	4	4	teacher	...	4
...	...	...	...	...	...	...	...	...	...	...	...	...
365	365	MS	1	18	0	GT3	1	1	3	at_home	...	3
366	366	MS	1	18	1	LE3	1	4	4	teacher	...	4
367	367	MS	0	17	0	GT3	1	1	1	other	...	5
368	368	MS	0	18	1	GT3	1	2	3	at_home	...	5

	Unnamed: 0	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	...	famrel
369	369	MS	0	18	0	GT3	1	4	4	other	...	3
370	370	MS	0	19	1	LE3	1	3	2	services	...	3
371	371	MS	1	18	0	LE3	1	1	2	at_home	...	4
372	372	MS	0	17	1	GT3	1	2	2	other	...	3
373	373	MS	0	17	0	GT3	1	1	2	other	...	3
374	374	MS	0	18	0	LE3	1	4	4	other	...	5
375	375	MS	0	18	0	GT3	1	1	1	other	...	4
376	376	MS	0	20	1	GT3	1	4	2	health	...	5
377	377	MS	0	18	0	LE3	1	4	4	teacher	...	5
378	378	MS	0	18	1	GT3	1	3	3	other	...	4
379	379	MS	0	17	0	GT3	1	3	1	at_home	...	4
380	380	MS	1	18	1	GT3	1	4	4	teacher	...	3
381	381	MS	1	18	0	GT3	1	2	1	other	...	4
382	382	MS	1	17	1	GT3	1	2	3	other	...	4
383	383	MS	1	19	0	GT3	1	1	1	other	...	4
384	384	MS	1	18	0	GT3	1	4	2	other	...	5
385	385	MS	0	18	0	GT3	1	2	2	at_home	...	5
386	386	MS	0	18	0	GT3	1	4	4	teacher	...	4
387	387	MS	0	19	0	GT3	1	2	3	services	...	5
388	388	MS	0	18	1	LE3	1	3	1	teacher	...	4
389	389	MS	0	18	1	GT3	1	1	1	other	...	1
390	390	MS	1	20	1	LE3	0	2	2	services	...	5
391	391	MS	1	17	1	LE3	1	3	1	services	...	2
392	392	MS	1	21	0	GT3	1	1	1	other	...	5
393	393	MS	1	18	0	LE3	1	3	2	services	...	4
394	394	MS	1	19	1	LE3	1	1	1	other	...	3

395 rows × 34 columns

In [10]:
```
math['M3'].describe()
```

Out[10]:
```
count    395.000000
mean      10.415190
std        4.581443
min        0.000000
25%        8.000000
50%       11.000000
75%       14.000000
max       20.000000
Name: M3, dtype: float64
```

In [11]:
```
por['P3'].describe()
```

Out[11]:
```
count    649.000000
mean      11.906009
std        3.230656
min        0.000000
25%       10.000000
50%       12.000000
75%       14.000000
max       19.000000
Name: P3, dtype: float64
```

In [12]:
```
math.shape
```

Out[12]:
```
(395, 34)
```

In [13]:
```
por.shape
```

Out[13]:
```
(649, 34)
```

In [14]:
```
math.info()
por.info()
```

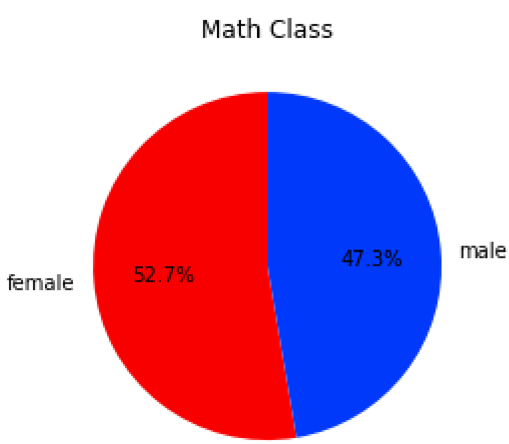
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 395 entries, 0 to 394
Data columns (total 34 columns):
Unnamed: 0    395 non-null int64
school        395 non-null object
sex           395 non-null int64
age           395 non-null int64
address       395 non-null int64
famsize       395 non-null object
Pstatus       395 non-null int64
Medu          395 non-null int64
Fedu          395 non-null int64
Mjob          395 non-null object
Fjob          395 non-null object
reason        395 non-null object
guardian      395 non-null object
traveltime    395 non-null int64
studytime     395 non-null int64
failures      395 non-null int64
schoolsup     395 non-null int64
famsup        395 non-null int64
paid          395 non-null int64
activities    395 non-null int64
nursery       395 non-null int64
higher        395 non-null int64
internet      395 non-null int64
romantic      395 non-null int64
famrel        395 non-null int64
freetime      395 non-null int64
goout         395 non-null int64
Dalc          395 non-null int64
Walc          395 non-null int64
health        395 non-null int64
m_absences    395 non-null int64
M1            395 non-null int64
M2            395 non-null int64
M3            395 non-null int64
dtypes: int64(28), object(6)
memory usage: 105.0+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 649 entries, 0 to 648
Data columns (total 34 columns):
Unnamed: 0    649 non-null int64
school        649 non-null object
sex           649 non-null int64
age           649 non-null int64
address       649 non-null int64
famsize       649 non-null object
Pstatus       649 non-null int64
Medu          649 non-null int64
Fedu          649 non-null int64
Mjob          649 non-null object
Fjob          649 non-null object
reason        649 non-null object
guardian      649 non-null object
traveltime    649 non-null int64
studytime     649 non-null int64
```

```
failures       649 non-null int64
schoolsup      649 non-null int64
famsup         649 non-null int64
paid           649 non-null int64
activities     649 non-null int64
nursery        649 non-null int64
higher         649 non-null int64
internet       649 non-null int64
romantic       649 non-null int64
famrel         649 non-null int64
freetime       649 non-null int64
goout          649 non-null int64
Dalc           649 non-null int64
Walc           649 non-null int64
health         649 non-null int64
p_absences     649 non-null int64
P1             649 non-null int64
P2             649 non-null int64
P3             649 non-null int64
dtypes: int64(28), object(6)
memory usage: 172.5+ KB
```

In [17]:

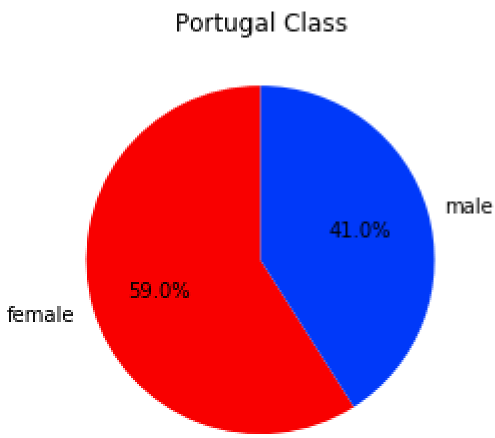
```
colors = ['#F90000','#0039F9']
plt.pie(math['sex'].value_counts(),startangle=90, labels=['female','male'], co
lors = colors,  autopct='%1.1f%%')
plt.title('Math Class')
```

Out[17]: `Text(0.5, 1.0, 'Math Class')`



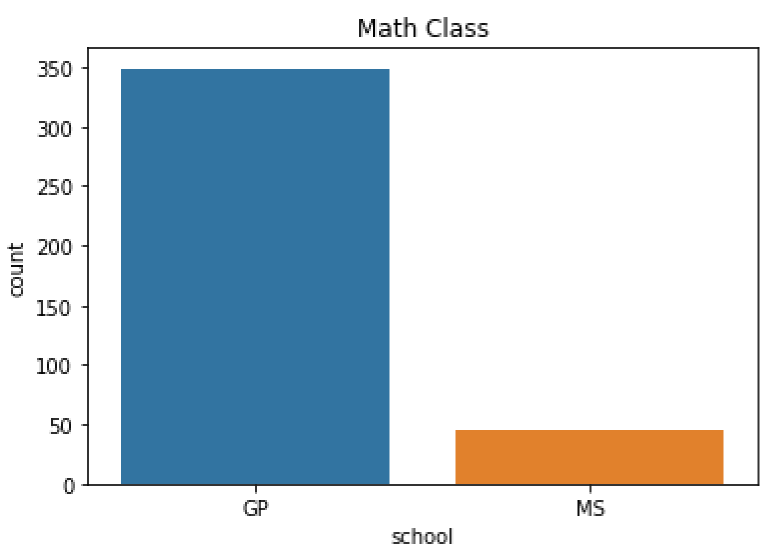
In [18]:
```
colors = ['#F90000', '#0039F9']
plt.pie(por['sex'].value_counts(),startangle=90, labels=['female','male'], col
ors = colors,  autopct='%1.1f%%')
plt.title('Portugal Class')
```

Out[18]: Text(0.5, 1.0, 'Portugal Class')



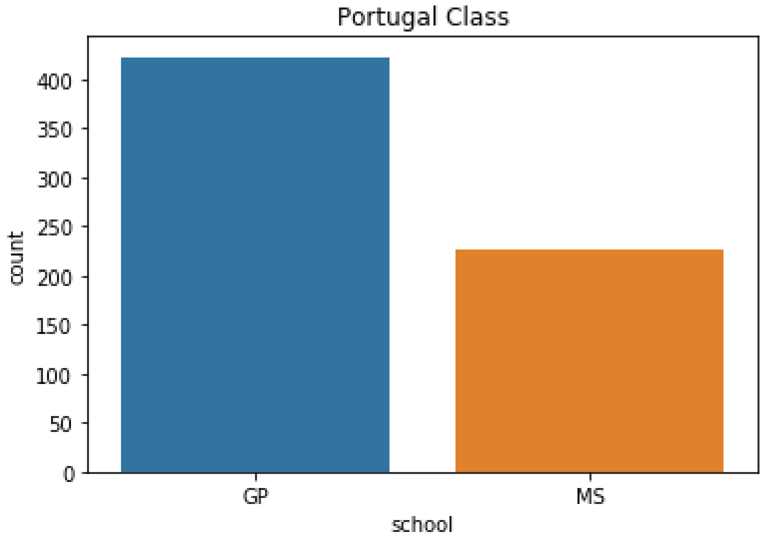
In [19]:
```
sns.countplot(data = math, x = 'school')
plt.title('Math Class')
```

Out[19]: Text(0.5, 1.0, 'Math Class')



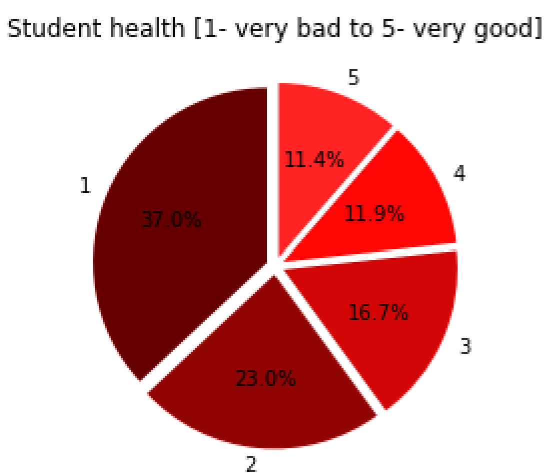
In [20]:
```
sns.countplot(data = por, x = 'school')
plt.title('Portugal Class')
```

Out[20]: Text(0.5, 1.0, 'Portugal Class')

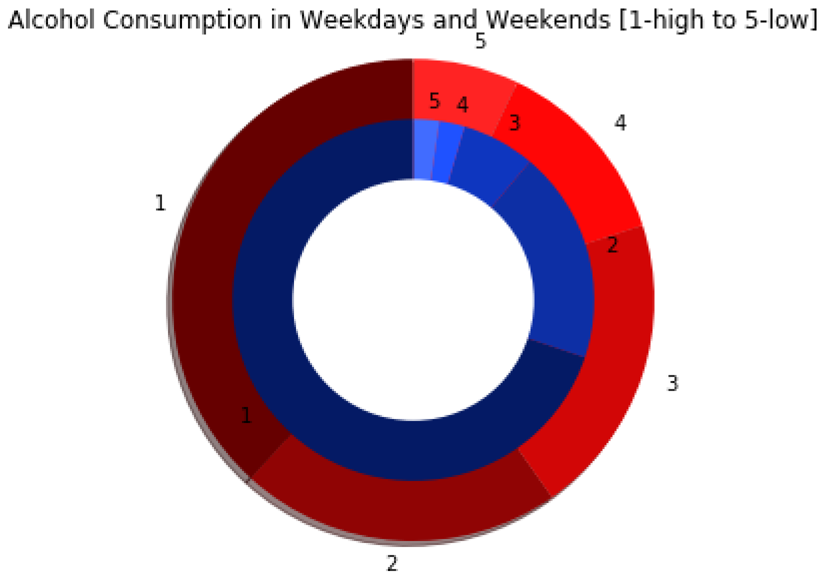


In [21]:
```
colors = ['#660000','#900404','#D20606','#FF0606', '#FF2323']
explode = (0.05,0.05,0.05,0.05,0.05)
plt.pie(math['health'].value_counts(),startangle=90, labels =['1','2','3','4',
'5'], colors=colors, explode = explode, autopct='%1.1f%%')
plt.title('Student health [1- very bad to 5- very good]')
```

Out[21]: Text(0.5, 1.0, 'Student health [1- very bad to 5- very good]')



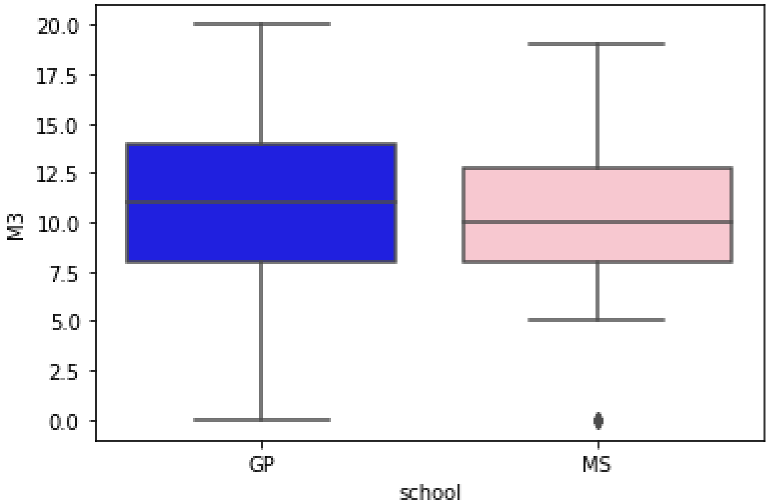
In [22]:
```
# Data to plot
labels_walc = ['1', '2', '3', '4', '5']
sizes_walc = [504, 337, 415, 280]
labels_dalc = ['1','2','3','4','5']
sizes_dalc = [315,189,125,212,270]
colors_walc = ['#660000','#900404','#D20606','#FF0606', '#FF2323']
colors_dalc = ['#041A65', '#0D2FA5', '#0E36BF', '#1F52FF', '#416CFF']

# Plot
plt.pie(math['Walc'].value_counts(), labels=labels_walc, colors=colors_walc, s
tartangle=90,frame=True, shadow=True)
plt.pie(math['Dalc'].value_counts(), labels=labels_dalc, colors=colors_dalc ,r
adius=0.75, startangle=90, shadow=True)
centre_circle = plt.Circle((0,0),0.5,color='black', fc='white',linewidth=0)
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

plt.axis('equal')
plt.tight_layout()
plt.title('Alcohol Consumption in Weekdays and Weekends [1-high to 5-low]')
plt.show()
print('Legend : Red - Weekdays, Blue - Weekends')
```




```
Legend : Red - Weekdays, Blue - Weekends
```

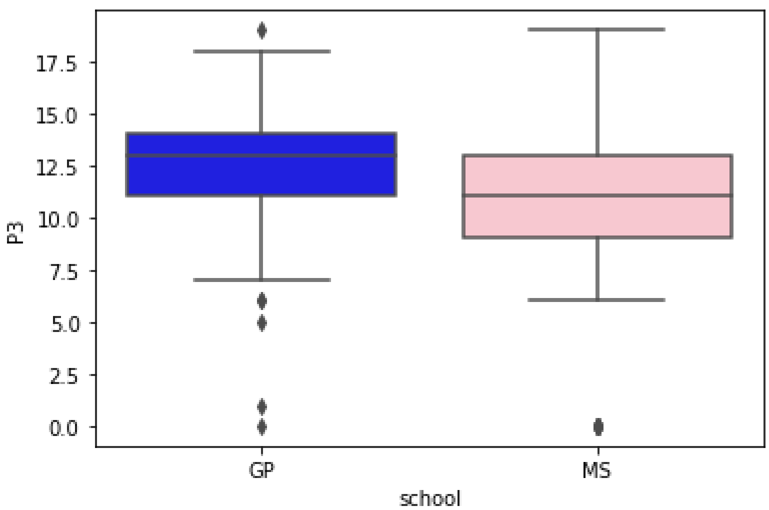
In [23]:
```
sns.boxplot(data = math,palette=["blue", "pink"], x='school', y='M3')
```

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x256b130acf8>



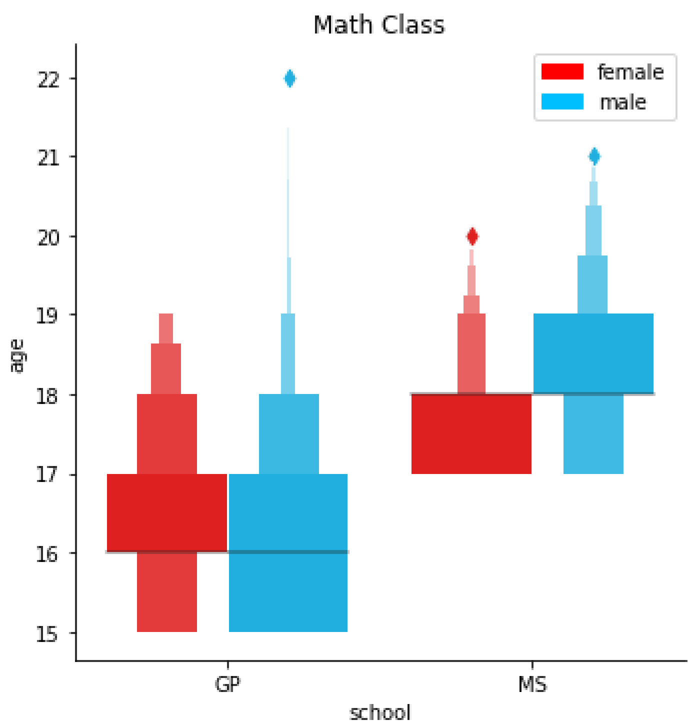
In [24]:
```
sns.boxplot(data = por ,palette=["blue", "pink"], x='school', y='P3')
```

Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0x256b13f9f98>



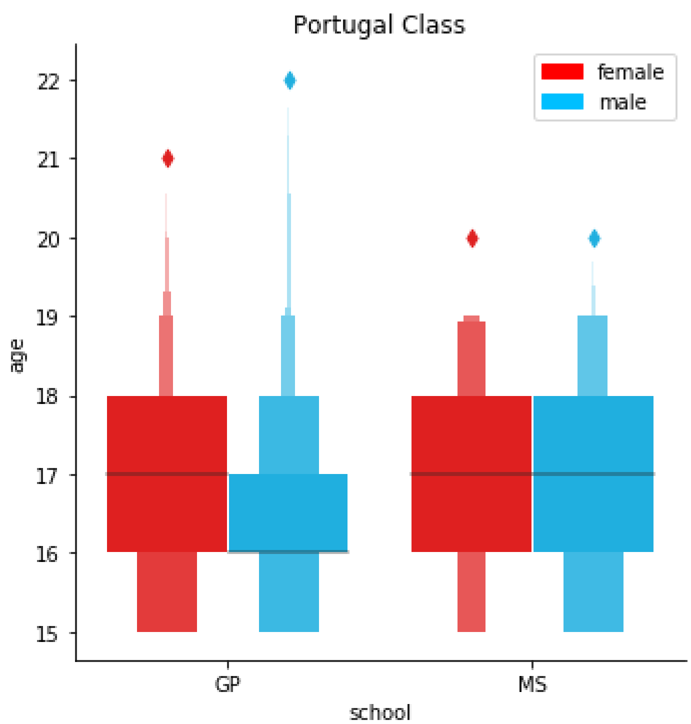
In [25]:
```
import matplotlib.patches as mpatches

sns.catplot(x="school", y="age", hue="sex", kind="boxen", palette=["r", "deepskyblue"], data= math, legend_out = False);

red_patch = mpatches.Patch(color='r', label='female')
cyan_patch = mpatches.Patch(color='deepskyblue', label='male')
plt.legend(handles=[red_patch, cyan_patch])
plt.title('Math Class')
```

Out[25]: Text(0.5, 1, 'Math Class')

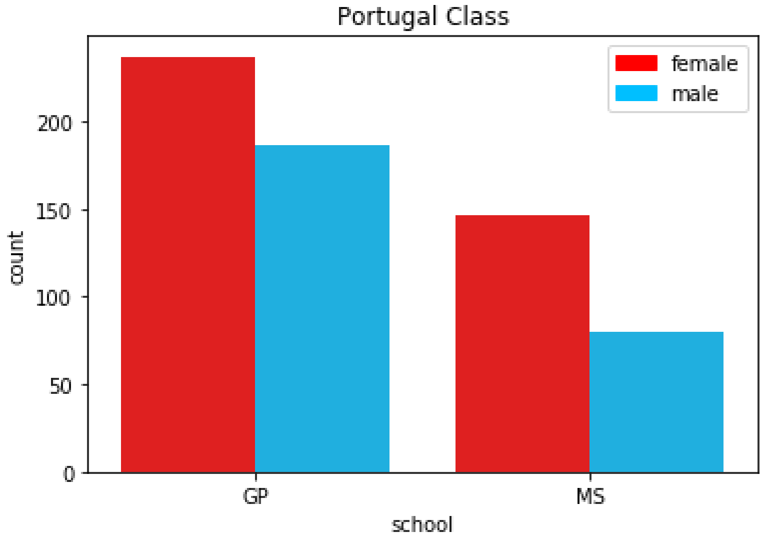


In [26]:
```
import matplotlib.patches as mpatches

sns.catplot(x="school", y="age", hue="sex", kind="boxen", palette=["r", "deepskyblue"], data= por, legend_out = False);

red_patch = mpatches.Patch(color='r', label='female')
cyan_patch = mpatches.Patch(color='deepskyblue', label='male')
plt.legend(handles=[red_patch, cyan_patch])
plt.title('Portugal Class')
```

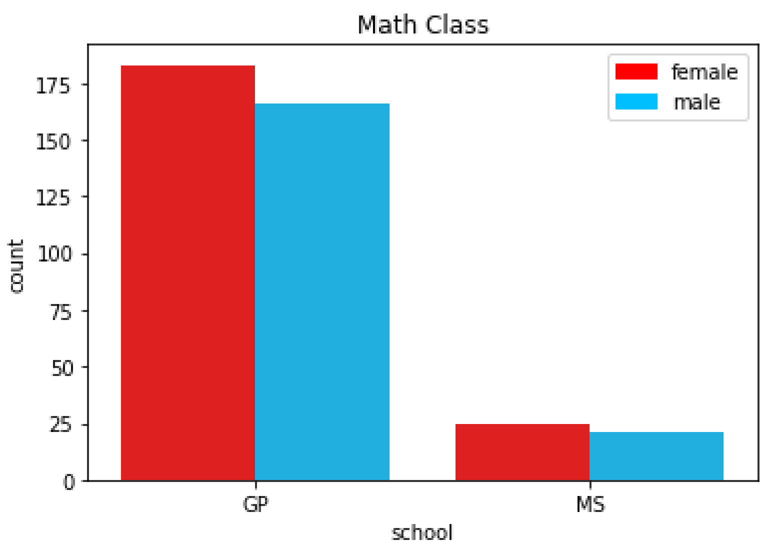
Out[26]:
```
Text(0.5, 1, 'Portugal Class')
```



In [49]:
```
sns.countplot(x="school", hue="sex", data=por, palette=["r", "deepskyblue"])
red_patch = mpatches.Patch(color='r', label='female')
cyan_patch = mpatches.Patch(color='deepskyblue', label='male')
plt.legend(handles=[red_patch, cyan_patch])
plt.title('Portugal Class')
plt.show()
```

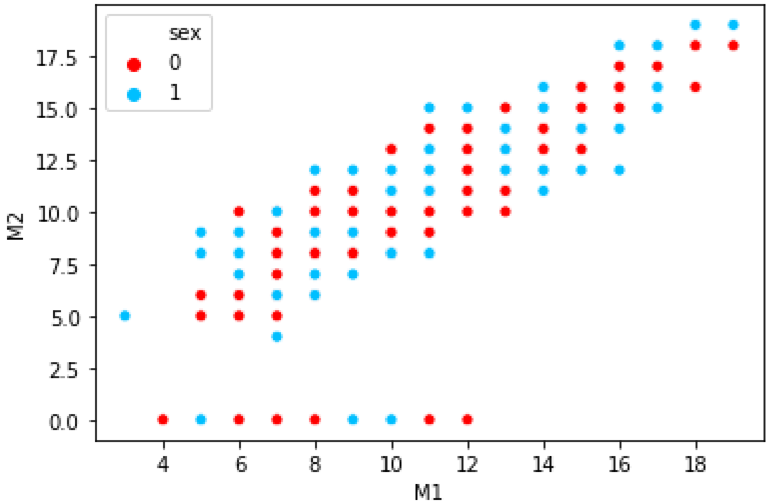


In [50]:
```
sns.countplot(x="school", hue="sex", data=math, palette=["r", "deepskyblue"])

red_patch = mpatches.Patch(color='r', label='female')
cyan_patch = mpatches.Patch(color='deepskyblue', label='male')
plt.legend(handles=[red_patch, cyan_patch])

plt.title('Math Class')
plt.show()
```



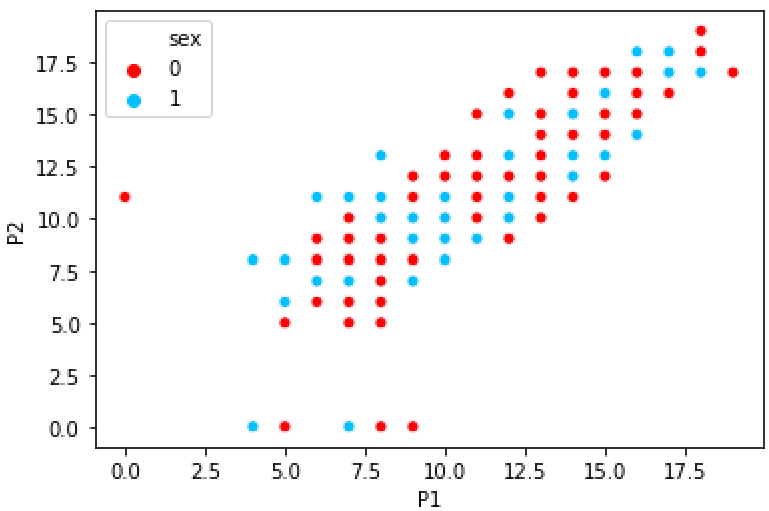
In [51]:
```
sns.scatterplot(x="M1", y="M2", hue="sex", size=None, data=math, legend='brie
f', palette=["r", "deepskyblue"])
```

Out[51]: <matplotlib.axes._subplots.AxesSubplot at 0x256b18c57f0>



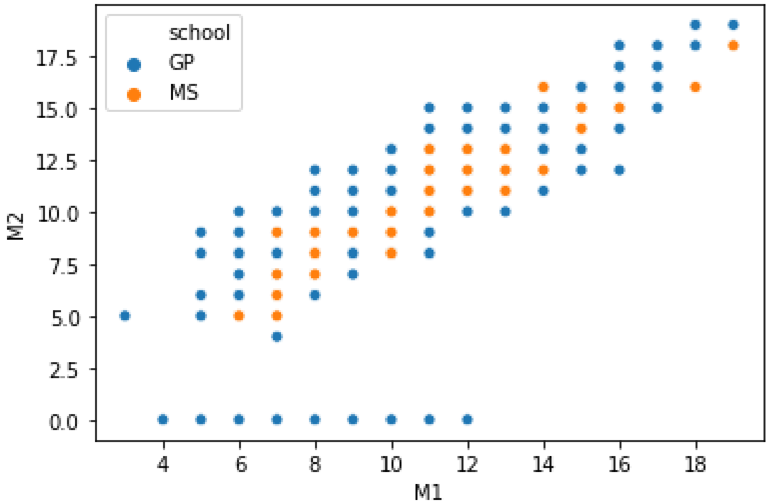
In [52]:
```
sns.scatterplot(x="P1", y="P2", hue="sex", size=None, data=por,  palette=["r",
"deepskyblue"])
```

Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0x256b2a5dcc0>



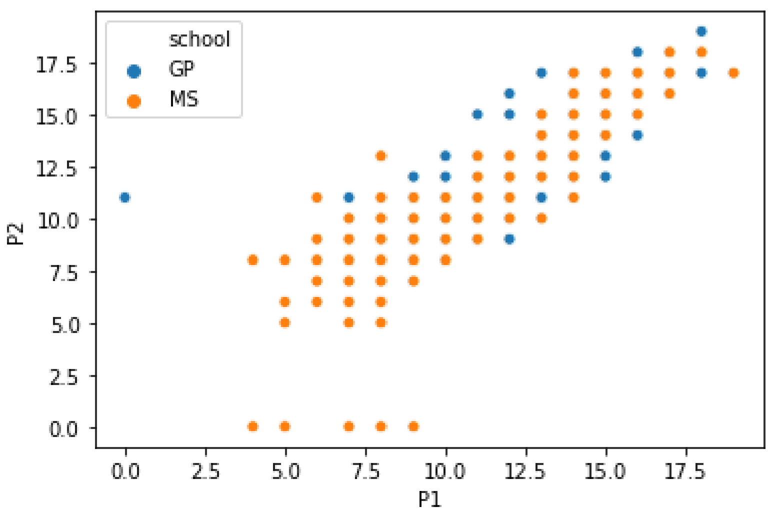
In [53]:
```
sns.scatterplot(x="M1", y="M2", hue="school", size=None, data=math, legend='brief')
```

Out[53]: <matplotlib.axes._subplots.AxesSubplot at 0x256b2e39438>

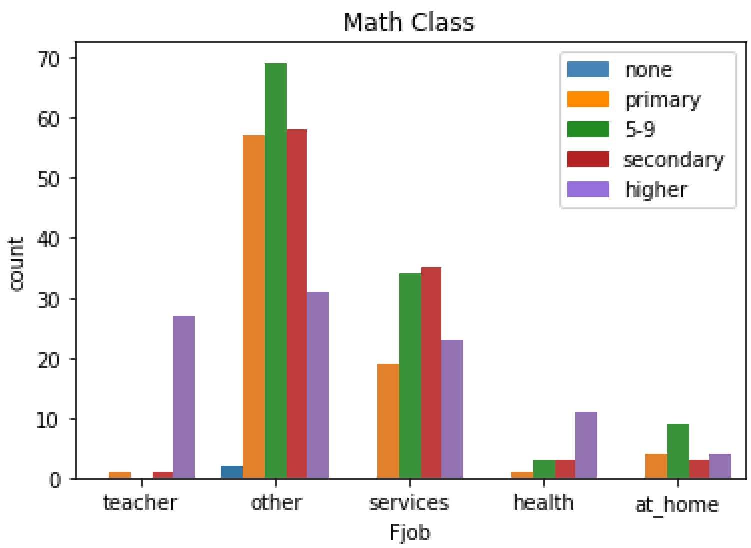


In [54]:
```
sns.scatterplot(x="P1", y="P2", hue="school", size=None, data=por, legend='brief')
```

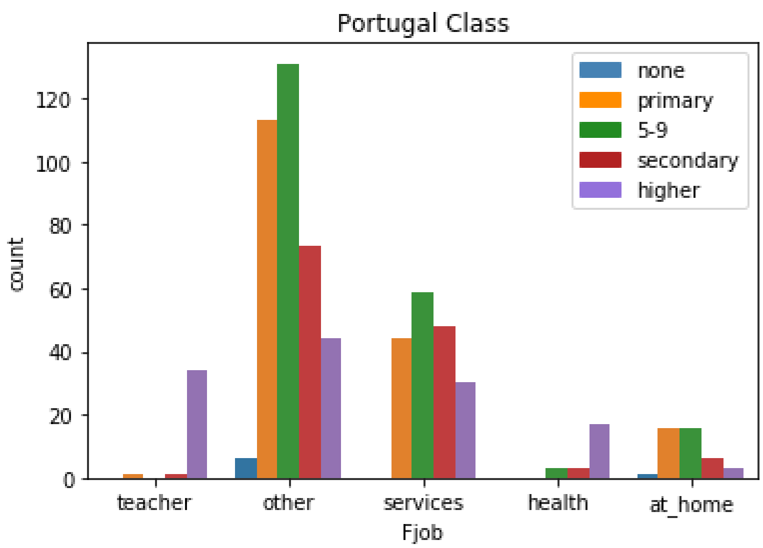
Out[54]: <matplotlib.axes._subplots.AxesSubplot at 0x256b2dcaf28>



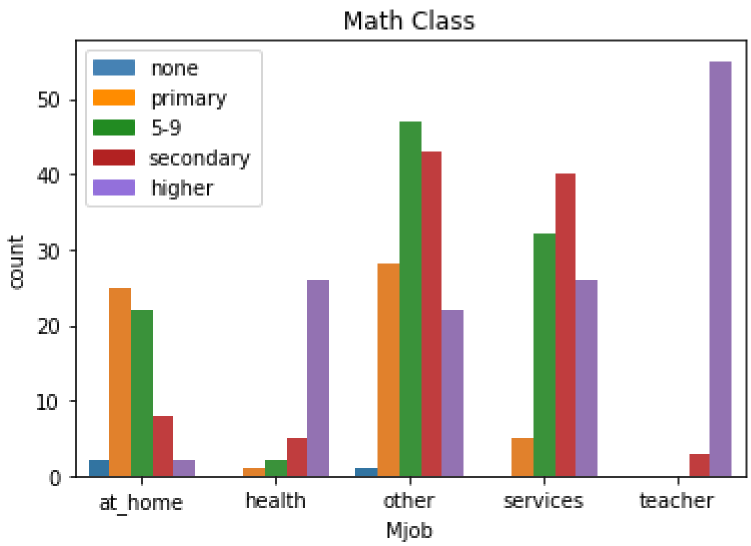
In [55]:
```
sns.countplot(x="Fjob", hue="Fedu", data=math)
plt.title("Father's job compared to their education level")
s_patch = mpatches.Patch(color='steelblue', label='none')
d_patch = mpatches.Patch(color='darkorange', label='primary')
f_patch = mpatches.Patch(color='forestgreen', label='5-9')
fi_patch = mpatches.Patch(color='firebrick', label='secondary')
m_patch = mpatches.Patch(color='mediumpurple', label='higher')
plt.legend(handles=[s_patch, d_patch, f_patch, fi_patch, m_patch])
plt.title('Math Class')
plt.show()
```



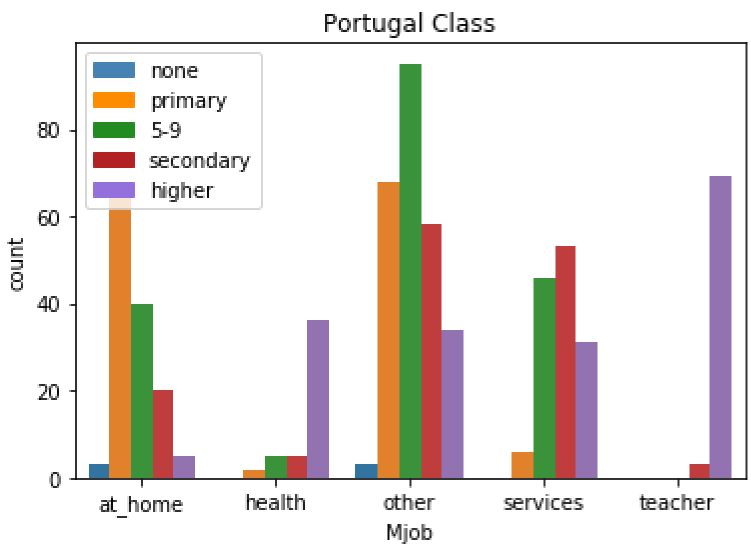
In [56]:
```
s = sns.countplot(x="Fjob", hue="Fedu", data=por)
plt.title("Father's job compared to their education level")
s.legend_.remove()
s_patch = mpatches.Patch(color='steelblue', label='none')
d_patch = mpatches.Patch(color='darkorange', label='primary')
f_patch = mpatches.Patch(color='forestgreen', label='5-9')
fi_patch = mpatches.Patch(color='firebrick', label='secondary')
m_patch = mpatches.Patch(color='mediumpurple', label='higher')
plt.legend(handles=[s_patch, d_patch, f_patch, fi_patch, m_patch])
plt.title('Portugal Class')
plt.show()
```



In [57]:
```
sns.countplot(x="Mjob", hue="Medu", data=math)
plt.title("Mother's job compared to their education level")
s_patch = mpatches.Patch(color='steelblue', label='none')
d_patch = mpatches.Patch(color='darkorange', label='primary')
f_patch = mpatches.Patch(color='forestgreen', label='5-9')
fi_patch = mpatches.Patch(color='firebrick', label='secondary')
m_patch = mpatches.Patch(color='mediumpurple', label='higher')
plt.legend(handles=[s_patch, d_patch, f_patch, fi_patch, m_patch])
plt.title('Math Class')
plt.show()
```

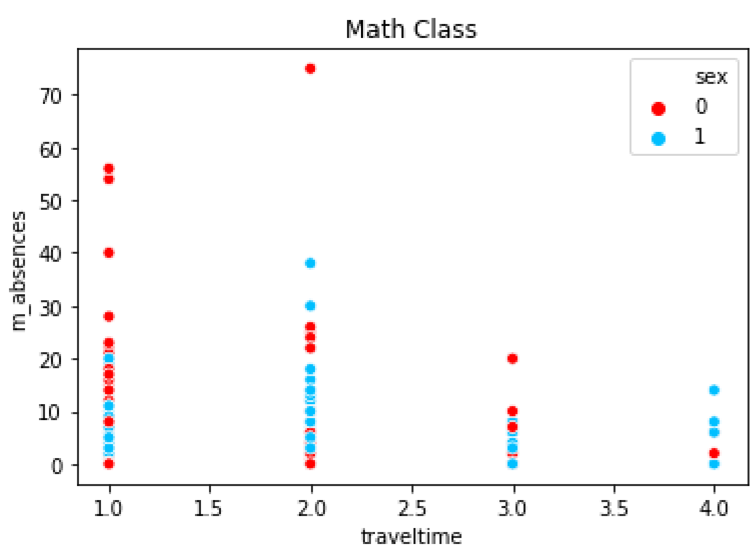


In [58]:
```
r = sns.countplot(x="Mjob", hue="Medu", data=por)
plt.title("Mother's job compared to their education level")
r.legend_.remove()
s_patch = mpatches.Patch(color='steelblue', label='none')
d_patch = mpatches.Patch(color='darkorange', label='primary')
f_patch = mpatches.Patch(color='forestgreen', label='5-9')
fi_patch = mpatches.Patch(color='firebrick', label='secondary')
m_patch = mpatches.Patch(color='mediumpurple', label='higher')
plt.legend(handles=[s_patch, d_patch, f_patch, fi_patch, m_patch])
plt.title('Portugal Class')
plt.show()
```



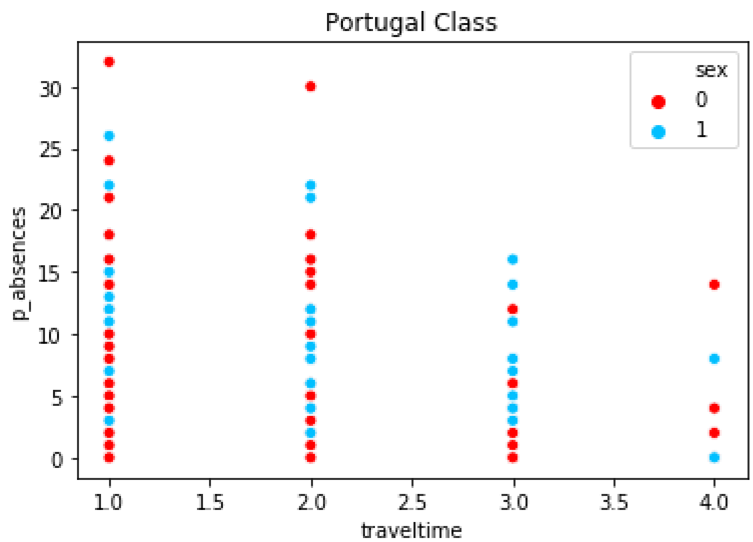
In [59]:
```
sns.scatterplot(x="traveltime", y="m_absences", hue="sex" , size=None, palette
=["r", "deepskyblue"], data=math, legend='brief')
plt.title('Math Class')
```

Out[59]: Text(0.5, 1.0, 'Math Class')



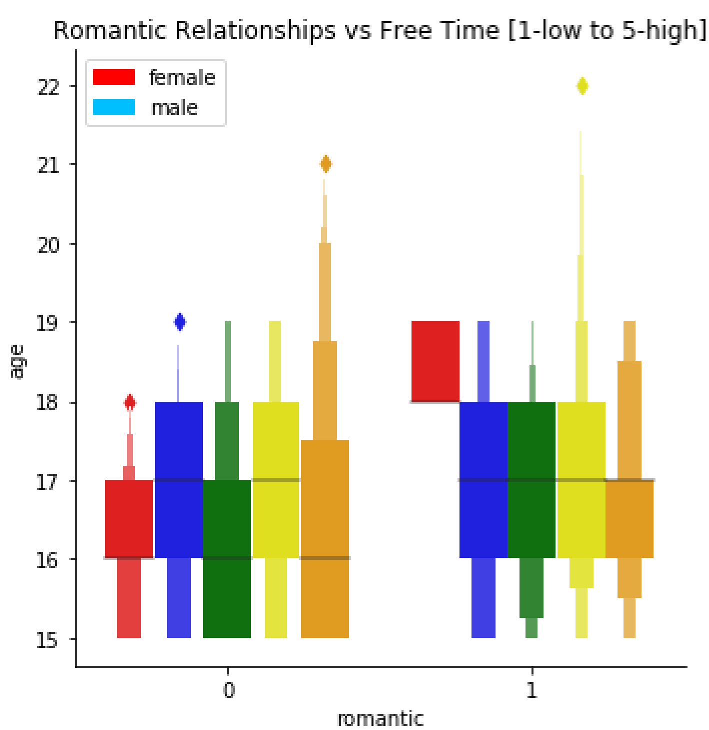
In [60]:
```
sns.scatterplot(x="traveltime", y="p_absences", hue="sex" , size=None, palette
=["r", "deepskyblue"], data=por, legend='brief')
plt.title('Portugal Class')
```

Out[60]: `Text(0.5, 1.0, 'Portugal Class')`

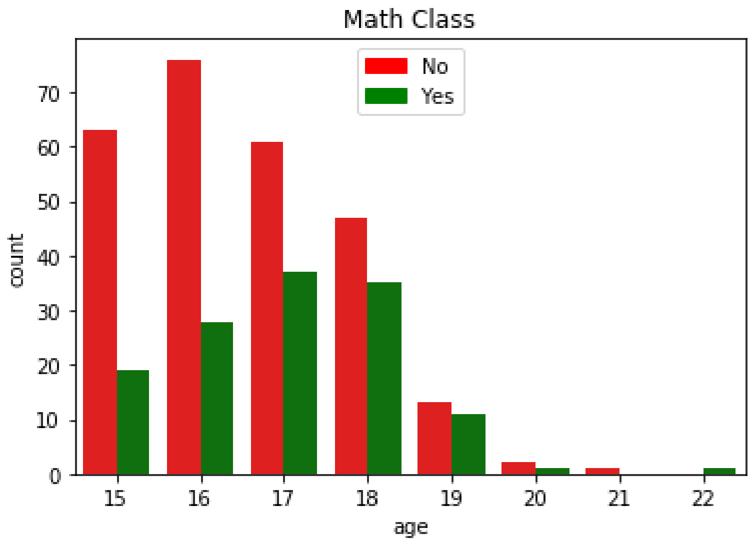


In [32]:
```
sns.catplot(x="romantic", y="age", hue="freetime", kind="boxen", palette=["re
d","blue","green","yellow","orange"], data= math, legend_out = False);

red_patch = mpatches.Patch(color='r', label='female')
blue_patch = mpatches.Patch(color='deepskyblue', label='male')
plt.legend(handles=[red_patch, cyan_patch])

plt.title("Romantic Relationships vs Free Time [1-low to 5-high]")
```

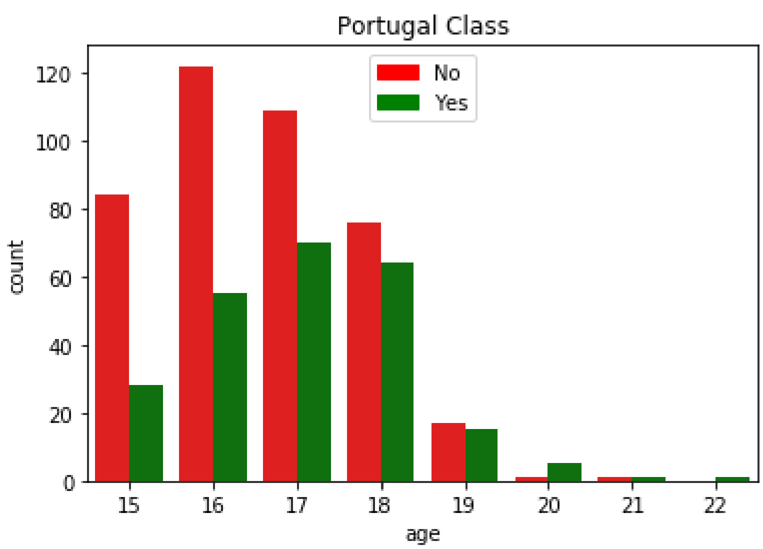
Out[32]: `Text(0.5, 1, 'Romantic Relationships vs Free Time [1-low to 5-high]')`



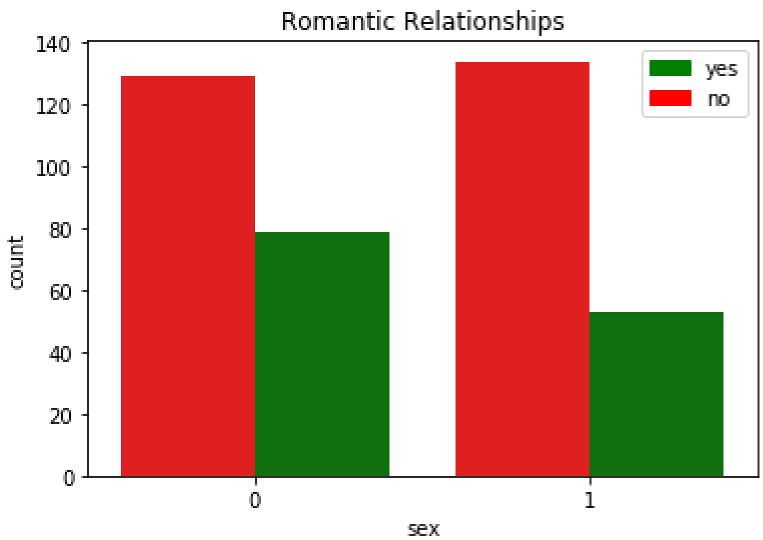
In [39]:
```
sns.countplot(x="age", hue="romantic", data=math, palette=["red","green"])
plt.title("Romantic Relationships")
s_patch = mpatches.Patch(color='red', label='No')
d_patch = mpatches.Patch(color='green', label='Yes')
plt.legend(handles=[s_patch, d_patch])
plt.title('Math Class')

plt.show()
```



In [40]:
```
sns.countplot(x="age", hue="romantic", data=por, palette=["red","green"])
plt.title("Romantic Relationships")
s_patch = mpatches.Patch(color='red', label='No')
d_patch = mpatches.Patch(color='green', label='Yes')

plt.legend(handles=[s_patch, d_patch])
plt.title('Portugal Class')

plt.show()
```

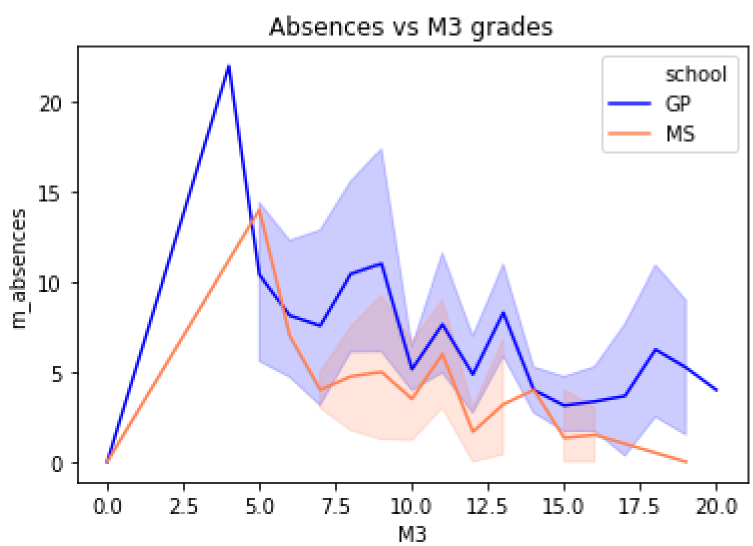


In [41]:
```
sns.countplot(x="sex", hue="romantic", data=math, palette=["red","green"])
plt.title("Romantic Relationships")
s_patch = mpatches.Patch(color='green', label='yes')
d_patch = mpatches.Patch(color='red', label='no')

plt.legend(handles=[s_patch, d_patch])

plt.show()
```



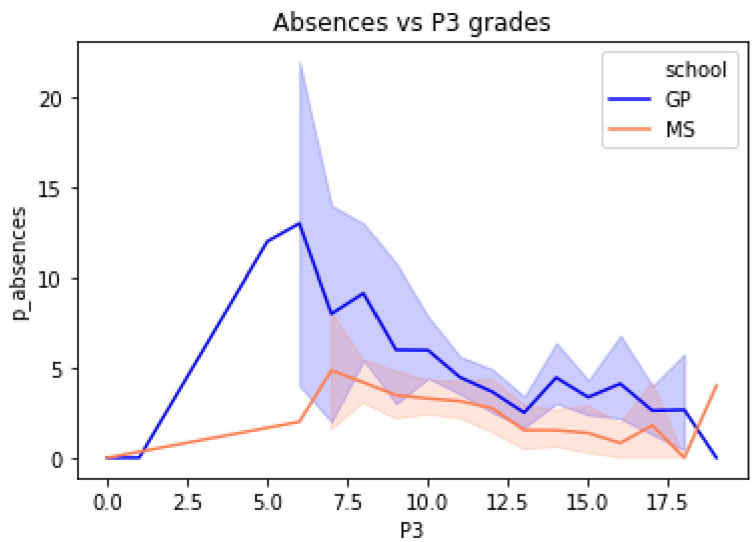
In [42]:
```
sns.lineplot(x="M3", y="m_absences",hue="school",size=None, palette=["blue",
"coral"], data=math, legend= 'brief')
plt.title("Absences vs M3 grades")
```

Out[42]: `Text(0.5, 1.0, 'Absences vs M3 grades')`



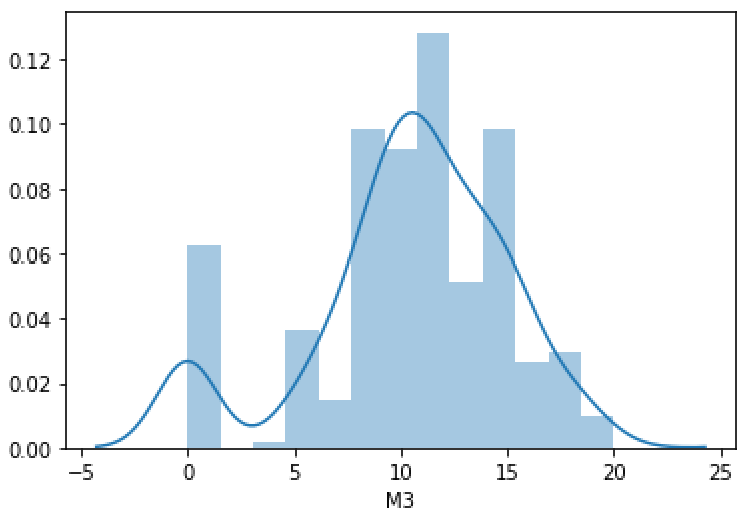
In [43]:
```
sns.lineplot(x="P3", y="p_absences",hue="school",size=None, palette=["blue",
"coral"], data=por, legend= 'brief')
plt.title("Absences vs P3 grades")
```

Out[43]:
```
Text(0.5, 1.0, 'Absences vs P3 grades')
```



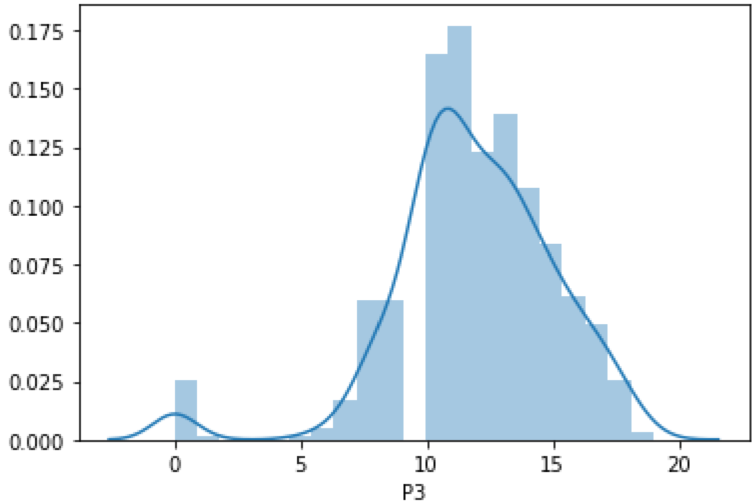
In [44]:
```
M3 = math["M3"]
sns.distplot(M3)
```

Out[44]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x256b2bfb5f8>
```

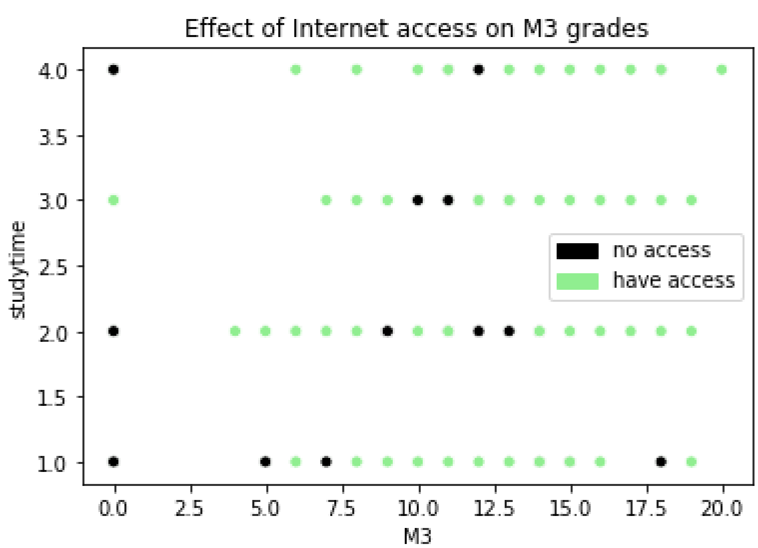


In [45]:
```
P3 = por["P3"]
sns.distplot(P3)
```

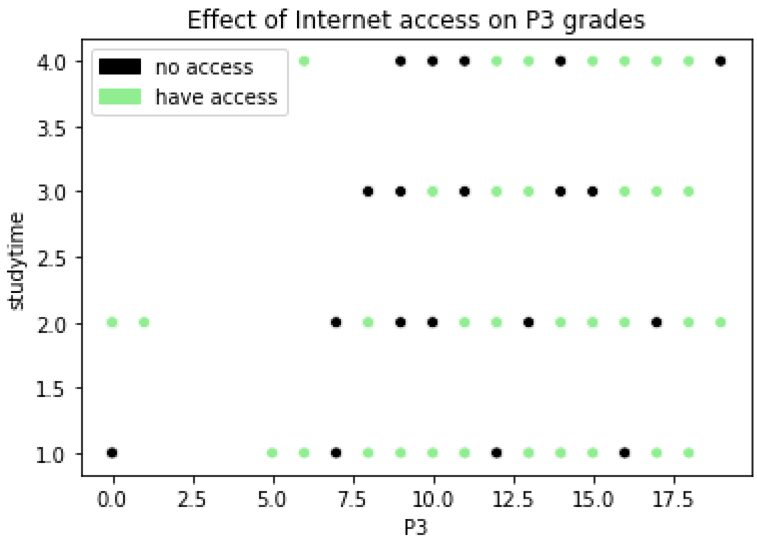
Out[45]: <matplotlib.axes._subplots.AxesSubplot at 0x256b2cdb898>



In [46]:
```
#internet access and grades
sns.scatterplot(x="M3", y="studytime", hue="internet", size=None, data=math, p
alette=["black", "lightgreen"], legend= False)

r_patch = mpatches.Patch(color='black', label='no access')
blue_patch = mpatches.Patch(color='lightgreen', label='have access')
plt.legend(handles=[r_patch, blue_patch])

plt.title("Effect of Internet access on M3 grades")

plt.show()
```



In [47]:
```
#internet access and grades
sns.scatterplot(x="P3", y="studytime", hue="internet", size=None, data=por, pa
lette=["black", "lightgreen"], legend= False)

r_patch = mpatches.Patch(color='black', label='no access')
blue_patch = mpatches.Patch(color='lightgreen', label='have access')
plt.legend(handles=[r_patch, blue_patch])
plt.title("Effect of Internet access on P3 grades")

plt.show()
```



In [2]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

In [3]:
```
%matplotlib inline
import numpy as np
import pandas as pd
import sklearn

import warnings
warnings.filterwarnings('ignore')
```

In [4]:
```
## Read data from csv file 'student-mat.csv'
math_data = pd.read_csv('student-mat.csv', sep=';')

## Read data from csv file 'student-por.csv'
port_data = pd.read_csv('student-por.csv', sep=';')
```

In [5]:
```
math_data.head()
port_data.head()
```

Out[5]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

Out[5]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

In [6]:
```
math_data.shape
port_data.shape
```

Out[6]: (395, 33)

Out[6]: (649, 33)

In [7]:
```
# Making dummy variables in math data and saving as mathdata_dummy

mathdata_dummy = pd.get_dummies(math_data, columns=['school','sex','address',
'famsize','Pstatus','Mjob','Fjob','reason','guardian','schoolsup','famsup','pa
id','activities','nursery','higher','internet','romantic'], drop_first=True)


mathdata_dummy.head()
```

Out[7]:

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetime	goout	Dalc	...	guardian_
0	18	4	4	2	2	0	4	3	4	1	...	
1	17	1	1	1	2	0	5	3	3	1	...	
2	15	1	1	1	2	3	4	3	2	2	...	
3	15	4	2	1	3	0	3	2	2	1	...	
4	16	3	3	1	2	0	4	3	2	1	...	

5 rows × 42 columns

In [11]:
```
# Starting Regression


# Creating MX AND MX1
# MX - selecting only the predictor variables and not the response variable G3
including G1 and G2

# MX1 - Selecting all the predictor variables including G1 and G2

MX = mathdata_dummy[['age',
 'Medu',
 'Fedu',
 'traveltime',
 'studytime',
 'failures',
 'famrel',
 'freetime',
 'goout',
 'Dalc',
 'Walc',
 'health',
 'absences',
 'school_MS',
 'sex_M',
 'address_U',
 'famsize_LE3',
 'Pstatus_T',
 'Mjob_health',
 'Mjob_other',
 'Mjob_services',
 'Mjob_teacher',
 'Fjob_health',
 'Fjob_other',
 'Fjob_services',
 'Fjob_teacher',
 'reason_home',
 'reason_other',
 'reason_reputation',
 'guardian_mother',
 'guardian_other',
 'schoolsup_yes',
 'famsup_yes',
 'paid_yes',
 'activities_yes',
 'nursery_yes',
 'higher_yes',
 'internet_yes',
 'romantic_yes']]


MX.head()

print(MX.shape)
```

Out[11]:

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetime	goout	Dalc	...	guardian_
0	18	4	4	2	2	0	4	3	4	1	...	
1	17	1	1	1	2	0	5	3	3	1	...	
2	15	1	1	1	2	3	4	3	2	2	...	
3	15	4	2	1	3	0	3	2	2	1	...	
4	16	3	3	1	2	0	4	3	2	1	...	

5 rows × 39 columns

```
(395, 39)
```

In [12]:
```
#Listing the column names of math data

list(mathdata_dummy.columns)
```

Out[12]:
```
['age',
 'Medu',
 'Fedu',
 'traveltime',
 'studytime',
 'failures',
 'famrel',
 'freetime',
 'goout',
 'Dalc',
 'Walc',
 'health',
 'absences',
 'G1',
 'G2',
 'G3',
 'school_MS',
 'sex_M',
 'address_U',
 'famsize_LE3',
 'Pstatus_T',
 'Mjob_health',
 'Mjob_other',
 'Mjob_services',
 'Mjob_teacher',
 'Fjob_health',
 'Fjob_other',
 'Fjob_services',
 'Fjob_teacher',
 'reason_home',
 'reason_other',
 'reason_reputation',
 'guardian_mother',
 'guardian_other',
 'schoolsup_yes',
 'famsup_yes',
 'paid_yes',
 'activities_yes',
 'nursery_yes',
 'higher_yes',
 'internet_yes',
 'romantic_yes']
```

In [13]:

```
# Y dependent variable of mathdata_dummy

MY = mathdata_dummy['G3']

MY.head()
```

Out[13]:

```
0     6
1     6
2    10
3    15
4    10
Name: G3, dtype: int64
```

In [29]:

```
MXGrade = mathdata_dummy[['age','G1','G2','G3','failures','absences']]
```

In [30]:

```
correlation1 = MXGrade.corr()
```

In [31]:

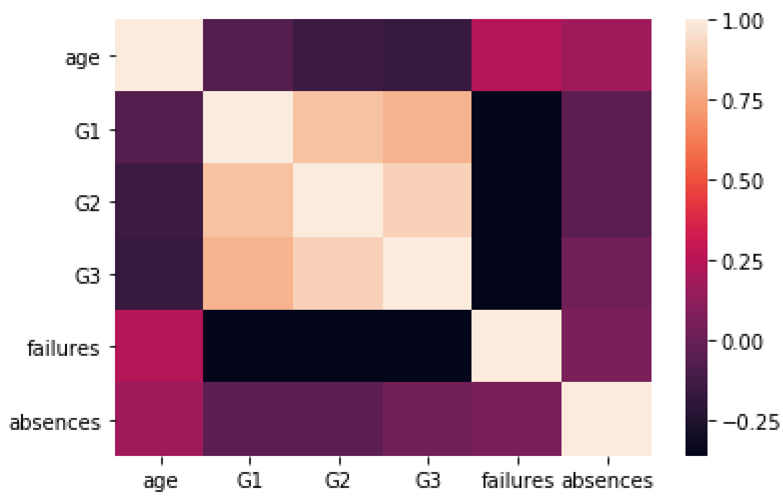
```
#checking correlation between age g1 g2 g3

correlation1
import seaborn
seaborn.heatmap(correlation1)
```

Out[31]:

	age	G1	G2	G3	failures	absences
age	1.000000	-0.064081	-0.143474	-0.161579	0.243665	0.175230
G1	-0.064081	1.000000	0.852118	0.801468	-0.354718	-0.031003
G2	-0.143474	0.852118	1.000000	0.904868	-0.355896	-0.031777
G3	-0.161579	0.801468	0.904868	1.000000	-0.360415	0.034247
failures	0.243665	-0.354718	-0.355896	-0.360415	1.000000	0.063726
absences	0.175230	-0.031003	-0.031777	0.034247	0.063726	1.000000

Out[31]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x269fdaffa90>
```



In [14]:
```
#Regression Model Excluding G1 and G2
import statsmodels.api as sb

MX = sb.add_constant(MX)

mod1 = sb.OLS(MY,MX)

fii1 = mod1.fit()
```

In [15]:
```
fii1
```

Out[15]:
```
<statsmodels.regression.linear_model.RegressionResultsWrapper at 0x269f8f62f9
8>
```

In [16]:
```
som1 = fii1.summary()
```

In [17]:
```
som1
```

Out[17]:

OLS Regression Results

Dep. Variable:	G3	**R-squared:**	0.276
Model:	OLS	**Adj. R-squared:**	0.196
Method:	Least Squares	**F-statistic:**	3.463
Date:	Wed, 27 Nov 2019	**Prob (F-statistic):**	3.32e-10
Time:	16:50:35	**Log-Likelihood:**	-1097.5
No. Observations:	395	**AIC:**	2275.
Df Residuals:	355	**BIC:**	2434.
Df Model:	39		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	14.0777	4.481	3.142	0.002	5.265	22.890
age	-0.3752	0.217	-1.727	0.085	-0.802	0.052
Medu	0.4569	0.323	1.414	0.158	-0.179	1.092
Fedu	-0.1046	0.278	-0.377	0.707	-0.651	0.441
traveltime	-0.2403	0.339	-0.709	0.479	-0.907	0.426
studytime	0.5495	0.288	1.910	0.057	-0.016	1.115
failures	-1.7240	0.333	-5.179	0.000	-2.379	-1.069
famrel	0.2316	0.246	0.942	0.347	-0.252	0.715
freetime	0.3024	0.237	1.274	0.203	-0.164	0.769
goout	-0.5937	0.225	-2.644	0.009	-1.035	-0.152
Dalc	-0.2722	0.331	-0.823	0.411	-0.923	0.378
Walc	0.2634	0.248	1.062	0.289	-0.224	0.751
health	-0.1768	0.161	-1.098	0.273	-0.493	0.140
absences	0.0563	0.029	1.943	0.053	-0.001	0.113
school_MS	0.7256	0.792	0.917	0.360	-0.831	2.282
sex_M	1.2624	0.500	2.525	0.012	0.279	2.246
address_U	0.5513	0.584	0.944	0.346	-0.597	1.700
famsize_LE3	0.7028	0.488	1.439	0.151	-0.257	1.663
Pstatus_T	-0.3201	0.724	-0.442	0.659	-1.744	1.104
Mjob_health	0.9981	1.118	0.893	0.373	-1.201	3.197
Mjob_other	-0.3590	0.713	-0.503	0.615	-1.762	1.044
Mjob_services	0.6583	0.798	0.825	0.410	-0.911	2.227
Mjob_teacher	-1.2415	1.038	-1.196	0.233	-3.283	0.800
Fjob_health	0.3477	1.438	0.242	0.809	-2.480	3.176
Fjob_other	-0.6197	1.023	-0.606	0.545	-2.632	1.392

Fjob_services	-0.4658	1.057	-0.441	0.660	-2.544	1.613
Fjob_teacher	1.3262	1.297	1.023	0.307	-1.224	3.876
reason_home	0.0785	0.554	0.142	0.887	-1.011	1.168
reason_other	0.7771	0.818	0.950	0.343	-0.831	2.385
reason_reputation	0.6130	0.577	1.063	0.288	-0.521	1.747
guardian_mother	0.0698	0.546	0.128	0.898	-1.003	1.143
guardian_other	0.7501	0.999	0.751	0.453	-1.216	2.716
schoolsup_yes	-1.3506	0.667	-2.025	0.044	-2.662	-0.039
famsup_yes	-0.8618	0.479	-1.800	0.073	-1.803	0.080
paid_yes	0.3397	0.478	0.711	0.477	-0.600	1.279
activities_yes	-0.3295	0.445	-0.741	0.459	-1.205	0.546
nursery_yes	-0.1773	0.549	-0.323	0.747	-1.258	0.903
higher_yes	1.3705	1.078	1.272	0.204	-0.749	3.490
internet_yes	0.4981	0.620	0.804	0.422	-0.720	1.717
romantic_yes	-1.0945	0.469	-2.332	0.020	-2.017	-0.172

Omnibus:	30.431	**Durbin-Watson:**	2.054
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	35.239
Skew:	-0.696	**Prob(JB):**	2.23e-08
Kurtosis:	3.450	**Cond. No.**	443.

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [22]:
```
# Starting Regression Model with Interaction effects
# Created MX4 which contains all the variables

MX4 = mathdata_dummy[['age',
 'Medu',
 'Fedu',
 'traveltime',
'studytime',
 'failures',
 'famrel',
'freetime',
 'goout',
 'Dalc',
 'Walc',
 'health',
 'absences',
 'G1',
 'G2',
 'G3',
 'school_MS',
 'sex_M',
 'address_U',
 'famsize_LE3',
 'Pstatus_T',
 'Mjob_health',
 'Mjob_other',
 'Mjob_services',
 'Mjob_teacher',
 'Fjob_health',
 'Fjob_other',
 'Fjob_services',
 'Fjob_teacher',
 'reason_home',
 'reason_other',
 'reason_reputation',
 'guardian_mother',
 'guardian_other',
'schoolsup_yes',
 'famsup_yes',
 'paid_yes',
 'activities_yes',
 'nursery_yes',
 'higher_yes',
 'internet_yes',
 'romantic_yes']]

MX4.head()
```

Out[22]:

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetime	goout	Dalc	...	guardian_
0	18	4	4	2	2	0	4	3	4	1	...	
1	17	1	1	1	2	0	5	3	3	1	...	
2	15	1	1	1	2	3	4	3	2	2	...	
3	15	4	2	1	3	0	3	2	2	1	...	
4	16	3	3	1	2	0	4	3	2	1	...	

5 rows × 42 columns

In [23]:
```
# REGRESSION MODEL WITH INTERACTION EFFECTS

import statsmodels.formula.api as smf
model_interaction = smf.ols(formula='G3 ~ failures + goout + sex_M + schoolsup
_yes + romantic_yes + failures:goout + failures:sex_M + failures:schoolsup_yes
+ failures:romantic_yes + goout:sex_M + goout:schoolsup_yes + goout:romantic_y
es + sex_M:schoolsup_yes + sex_M:romantic_yes +schoolsup_yes:romantic_yes', da
ta=MX4).fit()
summary = model_interaction.summary()
summary
```

Out[23]:

OLS Regression Results

Dep. Variable:	G3	**R-squared:**	0.197
Model:	OLS	**Adj. R-squared:**	0.166
Method:	Least Squares	**F-statistic:**	6.210
Date:	Wed, 27 Nov 2019	**Prob (F-statistic):**	9.59e-12
Time:	16:54:45	**Log-Likelihood:**	-1117.8
No. Observations:	395	**AIC:**	2268.
Df Residuals:	379	**BIC:**	2331.
Df Model:	15		
Covariance Type:	nonrobust		

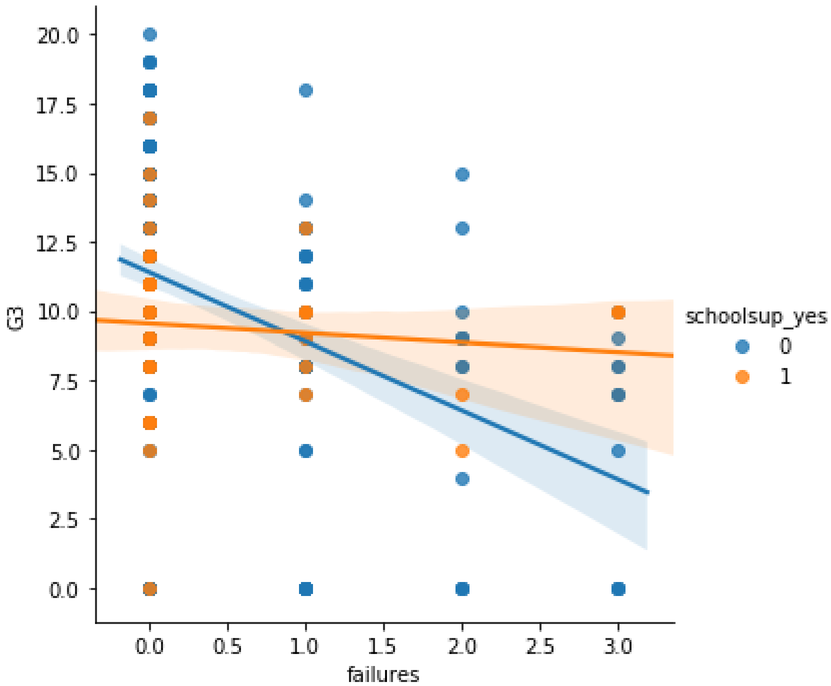
	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	12.8078	1.137	11.268	0.000	10.573	15.043
failures	-2.9940	0.923	-3.243	0.001	-4.810	-1.178
goout	-0.5200	0.342	-1.521	0.129	-1.192	0.152
sex_M	1.1825	1.338	0.884	0.377	-1.448	3.813
schoolsup_yes	-0.4928	1.896	-0.260	0.795	-4.220	3.234
romantic_yes	-2.3911	1.402	-1.706	0.089	-5.147	0.365
failures:goout	0.3877	0.248	1.561	0.119	-0.101	0.876
failures:sex_M	-1.2273	0.620	-1.980	0.048	-2.446	-0.009
failures:schoolsup_yes	2.0577	0.928	2.217	0.027	0.233	3.883
failures:romantic_yes	-0.2817	0.606	-0.465	0.642	-1.472	0.909
goout:sex_M	-0.0426	0.395	-0.108	0.914	-0.820	0.735
goout:schoolsup_yes	-0.4057	0.552	-0.735	0.463	-1.490	0.679
goout:romantic_yes	0.3432	0.417	0.823	0.411	-0.476	1.163
sex_M:schoolsup_yes	-1.1113	1.378	-0.806	0.421	-3.822	1.599
sex_M:romantic_yes	0.9246	0.926	0.998	0.319	-0.897	2.746
schoolsup_yes:romantic_yes	1.0825	1.494	0.724	0.469	-1.856	4.021

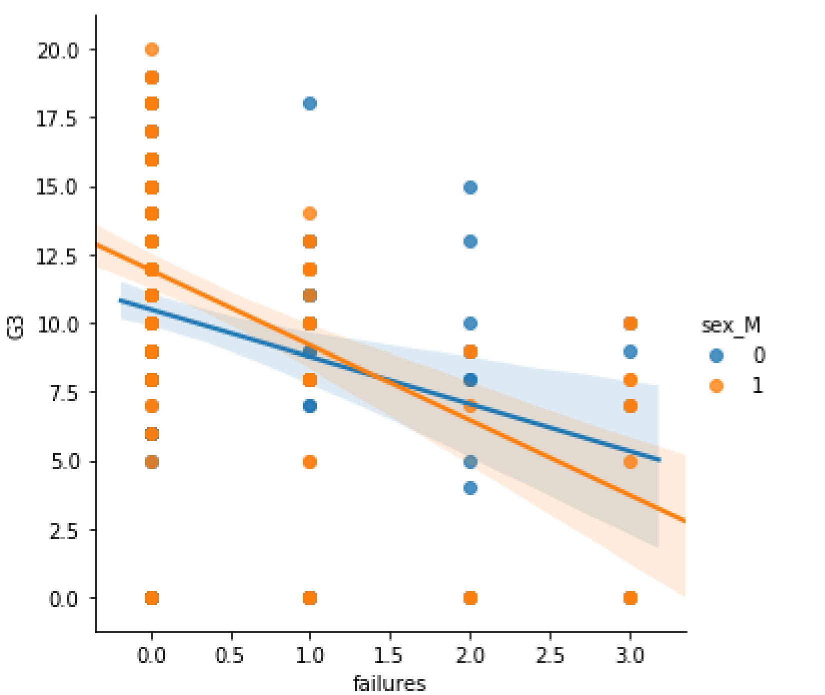
Omnibus:	31.291	**Durbin-Watson:**	2.040
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	36.669
Skew:	-0.692	**Prob(JB):**	1.09e-08
Kurtosis:	3.558	**Cond. No.**	47.0

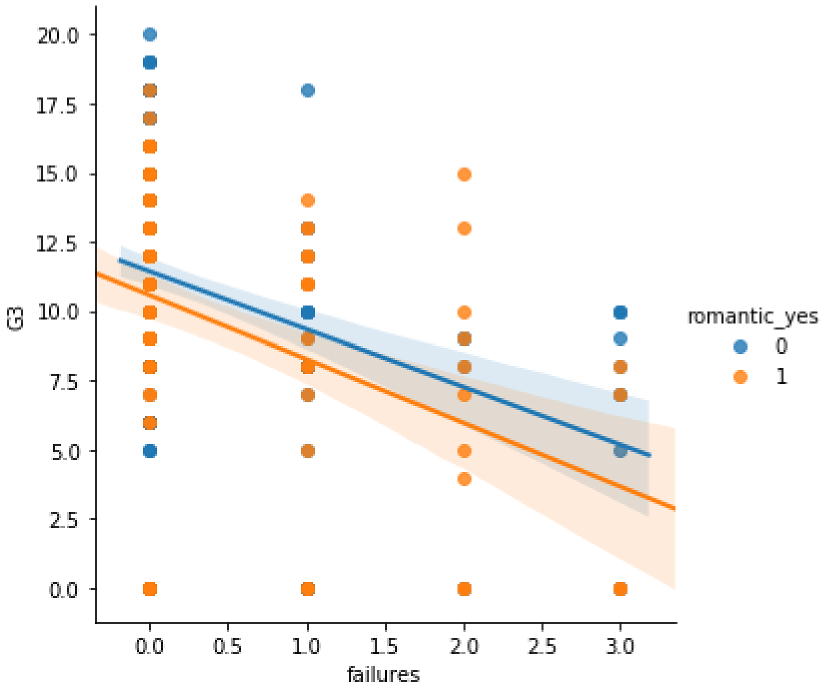
Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

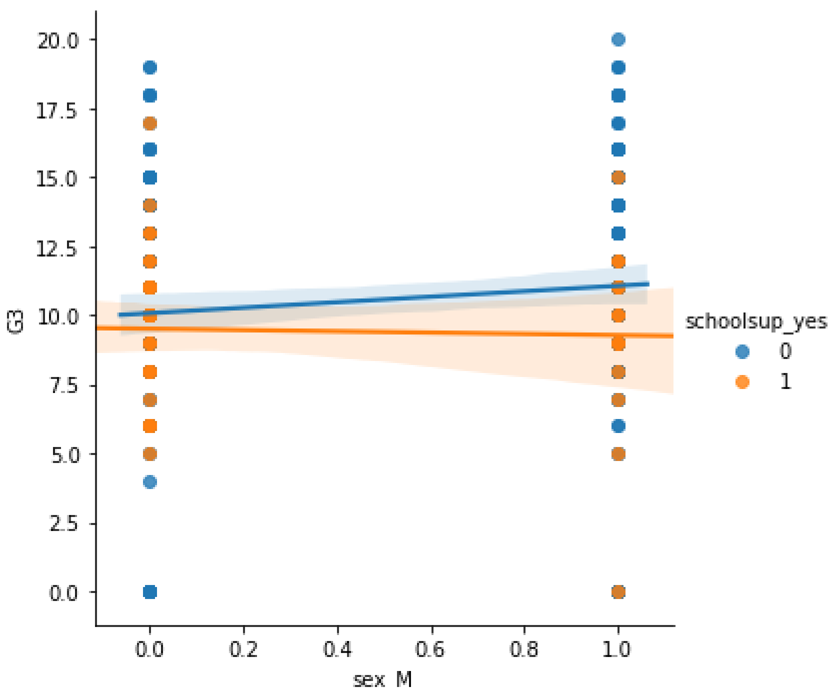
In [28]:
```
# Intersecting lines represent Interaction effect

# Presence of Interaction effect of failures*schoolsup_yes on G3
import seaborn
seaborn.lmplot(y='G3', x='failures', hue='schoolsup_yes', data=MX4)

# Presence of Interaction effect of failures*sex_M on G3
seaborn.lmplot(y='G3', x='failures', hue='sex_M', data=MX4)

# No Presence of Interaction effect between failures*romantic_yes on G3
seaborn.lmplot(y='G3', x='failures', hue='romantic_yes', data=MX4)

# No Presence of Interaction effect between sex_M*schoolsup_yes on G3
seaborn.lmplot(y='G3', x='sex_M', hue='schoolsup_yes', data=MX4)
```

```
Out[28]: <seaborn.axisgrid.FacetGrid at 0x269fdb8b7f0>
Out[28]: <seaborn.axisgrid.FacetGrid at 0x269fdba7588>
Out[28]: <seaborn.axisgrid.FacetGrid at 0x269fdb456d8>
Out[28]: <seaborn.axisgrid.FacetGrid at 0x269fdc404e0>
```






romantic_yes
0
1
G3
failures
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0


schoolsup_yes
0
1
G3
sex_M
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0

In [ ]:

In [1]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

In [2]:
```
%matplotlib inline
import numpy as np
import pandas as pd
import sklearn

import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
## Read data from csv file 'student-por.csv'
port_data = pd.read_csv('student-por.csv', sep=';')
```

In [4]:
```
port_data.head()
```

Out[4]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

In [5]:
```
# checking the shape of the dataset
port_data.shape
```

Out[5]: (649, 33)

In [6]:
```
# Making dummy variables in portugese data and saving

portdata_dummy = pd.get_dummies(port_data,columns=['school','sex','address','famsize','Pstatus','Mjob','Fjob','reason','guardian','schoolsup','famsup','paid','activities','nursery','higher','internet','romantic'], drop_first=True)


portdata_dummy.head()
```

Out[6]:

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetime	goout	Dalc	...	guardian_
0	18	4	4	2	2	0	4	3	4	1	...	
1	17	1	1	1	2	0	5	3	3	1	...	
2	15	1	1	1	2	0	4	3	2	2	...	
3	15	4	2	1	3	0	3	2	2	1	...	
4	16	3	3	1	2	0	4	3	2	1	...	

5 rows × 42 columns

In [7]:
```
# Starting Regression

# PX - selecting only the predictor variables and not the response variable G3
including G1 and G2

PX = portdata_dummy[['age',
 'Medu',
 'Fedu',
 'traveltime',
 'studytime',
 'failures',
 'famrel',
 'freetime',
 'goout',
 'Dalc',
 'Walc',
 'health',
 'absences',
 'school_MS',
 'sex_M',
 'address_U',
 'famsize_LE3',
 'Pstatus_T',
 'Mjob_health',
 'Mjob_other',
 'Mjob_services',
 'Mjob_teacher',
 'Fjob_health',
 'Fjob_other',
 'Fjob_services',
 'Fjob_teacher',
 'reason_home',
 'reason_other',
 'reason_reputation',
 'guardian_mother',
 'guardian_other',
 'schoolsup_yes',
 'famsup_yes',
 'paid_yes',
 'activities_yes',
 'nursery_yes',
 'higher_yes',
 'internet_yes',
 'romantic_yes']]


PX.head()

print(PX.shape)
```

Out[7]:

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetime	goout	Dalc	...	guardian_
0	18	4	4	2	2	0	4	3	4	1	...	
1	17	1	1	1	2	0	5	3	3	1	...	
2	15	1	1	1	2	0	4	3	2	2	...	
3	15	4	2	1	3	0	3	2	2	1	...	
4	16	3	3	1	2	0	4	3	2	1	...	

5 rows × 39 columns

```
(649, 39)
```

In [8]:
```
# listing the columns of portuguese_dummy dataset

list(portdata_dummy.columns)
```

Out[8]:
```
['age',
 'Medu',
 'Fedu',
 'traveltime',
 'studytime',
 'failures',
 'famrel',
 'freetime',
 'goout',
 'Dalc',
 'Walc',
 'health',
 'absences',
 'G1',
 'G2',
 'G3',
 'school_MS',
 'sex_M',
 'address_U',
 'famsize_LE3',
 'Pstatus_T',
 'Mjob_health',
 'Mjob_other',
 'Mjob_services',
 'Mjob_teacher',
 'Fjob_health',
 'Fjob_other',
 'Fjob_services',
 'Fjob_teacher',
 'reason_home',
 'reason_other',
 'reason_reputation',
 'guardian_mother',
 'guardian_other',
 'schoolsup_yes',
 'famsup_yes',
 'paid_yes',
 'activities_yes',
 'nursery_yes',
 'higher_yes',
 'internet_yes',
 'romantic_yes']
```

In [9]:
```
# Y dependent variable of portdata_dummy

PY = portdata_dummy['G3']

PY.head()
```

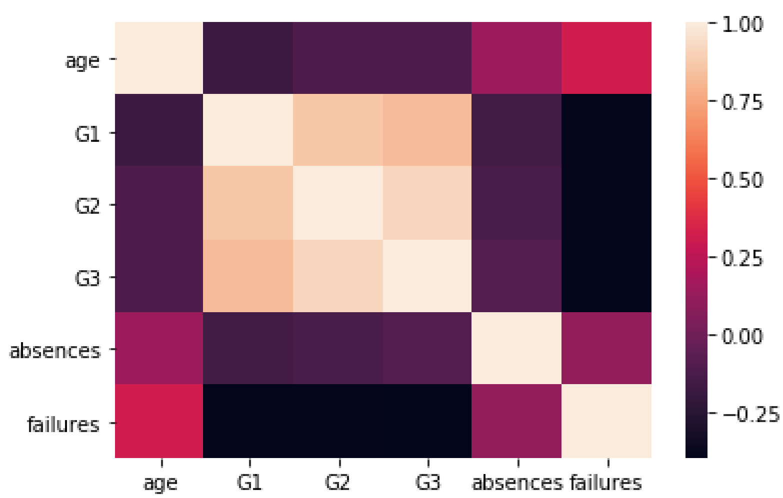
Out[9]:
```
0    11
1    11
2    12
3    14
4    13
Name: G3, dtype: int64
```

In [10]:
```
# checking correlation between numeric variables
PXGrade = portdata_dummy[['age','G1','G2','G3','absences','failures']]
correlation1 = PXGrade.corr()
correlation1

# heatmap of numeric variables
import seaborn
seaborn.heatmap(correlation1)
```

Out[10]:

	age	G1	G2	G3	absences	failures
age	1.000000	-0.174322	-0.107119	-0.106505	0.149998	0.319968
G1	-0.174322	1.000000	0.864982	0.826387	-0.147149	-0.384210
G2	-0.107119	0.864982	1.000000	0.918548	-0.124745	-0.385782
G3	-0.106505	0.826387	0.918548	1.000000	-0.091379	-0.393316
absences	0.149998	-0.147149	-0.124745	-0.091379	1.000000	0.122779
failures	0.319968	-0.384210	-0.385782	-0.393316	0.122779	1.000000

Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1bbaa26b630>
```



In [11]:
```
PXGrade.head()
```

Out[11]:

	age	G1	G2	G3	absences	failures
0	18	0	11	11	4	0
1	17	9	11	11	2	0
2	15	12	13	12	6	0
3	15	14	14	14	0	0
4	16	11	13	13	0	0

In [12]:
```
#Excluding G1 and G2 as they are higly correlated with G3

# PORTUGUESE REGRESSION MODEL

import statsmodels.api as sm

PX = sm.add_constant(PX)

mod1 = sm.OLS(PY,PX)

fii1 = mod1.fit()
fii1
```

Out[12]:
```
<statsmodels.regression.linear_model.RegressionResultsWrapper at 0x1bbab594ac
8>
```

In [13]:
```
som1 = fii1.summary()
```

In [14]:
```
som1
```

Out[14]:

OLS Regression Results

Dep. Variable:	G3	**R-squared:**	0.360
Model:	OLS	**Adj. R-squared:**	0.319
Method:	Least Squares	**F-statistic:**	8.797
Date:	Wed, 27 Nov 2019	**Prob (F-statistic):**	3.27e-38
Time:	18:54:39	**Log-Likelihood:**	-1536.5
No. Observations:	649	**AIC:**	3153.
Df Residuals:	609	**BIC:**	3332.
Df Model:	39		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	8.6815	1.985	4.373	0.000	4.783	12.580
age	0.1562	0.102	1.528	0.127	-0.045	0.357
Medu	0.0353	0.151	0.233	0.816	-0.262	0.332
Fedu	0.1669	0.138	1.211	0.226	-0.104	0.437
traveltime	0.0625	0.159	0.393	0.695	-0.250	0.375
studytime	0.4067	0.140	2.906	0.004	0.132	0.682
failures	-1.4122	0.205	-6.906	0.000	-1.814	-1.011
famrel	0.1616	0.116	1.391	0.165	-0.066	0.390
freetime	-0.1378	0.112	-1.226	0.221	-0.358	0.083
goout	-0.0661	0.107	-0.615	0.539	-0.277	0.145
Dalc	-0.2048	0.153	-1.338	0.181	-0.505	0.096
Walc	-0.0815	0.118	-0.688	0.492	-0.314	0.151
health	-0.1874	0.077	-2.428	0.015	-0.339	-0.036
absences	-0.0381	0.025	-1.531	0.126	-0.087	0.011
school_MS	-1.2003	0.267	-4.490	0.000	-1.725	-0.675
sex_M	-0.6331	0.250	-2.532	0.012	-1.124	-0.142
address_U	0.3227	0.262	1.233	0.218	-0.191	0.837
famsize_LE3	0.3025	0.245	1.235	0.217	-0.179	0.784
Pstatus_T	0.1769	0.347	0.510	0.610	-0.504	0.858
Mjob_health	0.9015	0.538	1.677	0.094	-0.154	1.957
Mjob_other	0.0504	0.303	0.166	0.868	-0.544	0.645
Mjob_services	0.4205	0.373	1.127	0.260	-0.312	1.153
Mjob_teacher	0.5118	0.502	1.020	0.308	-0.474	1.498
Fjob_health	-0.6122	0.752	-0.814	0.416	-2.090	0.865
Fjob_other	-0.1844	0.456	-0.404	0.686	-1.080	0.712

Fjob_services	-0.6434	0.479	-1.343	0.180	-1.585	0.298
Fjob_teacher	0.5797	0.672	0.862	0.389	-0.741	1.900
reason_home	0.0505	0.285	0.177	0.859	-0.509	0.610
reason_other	-0.4349	0.368	-1.183	0.237	-1.157	0.287
reason_reputation	0.2177	0.298	0.730	0.465	-0.368	0.803
guardian_mother	-0.3385	0.265	-1.276	0.202	-0.859	0.182
guardian_other	0.1050	0.532	0.197	0.844	-0.939	1.149
schoolsup_yes	-1.3112	0.364	-3.602	0.000	-2.026	-0.596
famsup_yes	-0.0204	0.228	-0.089	0.929	-0.469	0.428
paid_yes	-0.3716	0.461	-0.805	0.421	-1.278	0.535
activities_yes	0.2192	0.223	0.981	0.327	-0.220	0.658
nursery_yes	-0.2161	0.271	-0.796	0.426	-0.749	0.317
higher_yes	1.7330	0.383	4.528	0.000	0.981	2.485
internet_yes	0.2529	0.276	0.915	0.360	-0.290	0.796
romantic_yes	-0.4316	0.229	-1.883	0.060	-0.882	0.019

Omnibus:	127.139	**Durbin-Watson:**	1.926
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	422.670
Skew:	-0.908	**Prob(JB):**	1.65e-92
Kurtosis:	6.512	**Cond. No.**	372.

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [15]:
```
# Creating PX2 with all the variables including G3

PX2 = portdata_dummy[['age',
 'Medu',
 'Fedu',
 'traveltime',
'studytime',
 'failures',
 'famrel',
'freetime',
 'goout',
 'Dalc',
 'Walc',
 'health',
 'absences',
 'G1',
 'G2',
 'G3',
 'school_MS',
 'sex_M',
 'address_U',
 'famsize_LE3',
 'Pstatus_T',
 'Mjob_health',
 'Mjob_other',
 'Mjob_services',
 'Mjob_teacher',
 'Fjob_health',
 'Fjob_other',
 'Fjob_services',
 'Fjob_teacher',
 'reason_home',
 'reason_other',
 'reason_reputation',
 'guardian_mother',
 'guardian_other',
'schoolsup_yes',
 'famsup_yes',
 'paid_yes',
 'activities_yes',
 'nursery_yes',
 'higher_yes',
 'internet_yes',
 'romantic_yes']]

PX2.head()
```

Out[15]:

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetime	goout	Dalc	...	guardian_
0	18	4	4	2	2	0	4	3	4	1	...	
1	17	1	1	1	2	0	5	3	3	1	...	
2	15	1	1	1	2	0	4	3	2	2	...	
3	15	4	2	1	3	0	3	2	2	1	...	
4	16	3	3	1	2	0	4	3	2	1	...	

5 rows × 42 columns

In [16]:
```
# Portuguese Regression Model with Interaction effects
# Including Interaction Terms

import statsmodels.formula.api as smf
model_interaction = smf.ols(formula='G3 ~ studytime + failures + health + scho
ol_MS + sex_M + schoolsup_yes + higher_yes + studytime:failures + studytime:he
alth + studytime:school_MS + studytime:sex_M + studytime:schoolsup_yes + study
time:higher_yes + failures:health + failures:school_MS + failures:sex_M + fail
ures:schoolsup_yes + failures:higher_yes + health:school_MS + health:sex_M + h
ealth:schoolsup_yes + health:higher_yes + school_MS:sex_M +  school_MS:schools
up_yes + school_MS:higher_yes + sex_M:schoolsup_yes + sex_M:higher_yes', data=
PX2).fit()
summary = model_interaction.summary()
summary
```

Out[16]:

OLS Regression Results

Dep. Variable:	G3	**R-squared:**	0.331
Model:	OLS	**Adj. R-squared:**	0.302
Method:	Least Squares	**F-statistic:**	11.37
Date:	Wed, 27 Nov 2019	**Prob (F-statistic):**	7.88e-39
Time:	18:55:22	**Log-Likelihood:**	-1551.1
No. Observations:	649	**AIC:**	3158.
Df Residuals:	621	**BIC:**	3284.
Df Model:	27		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	9.6265	1.724	5.583	0.000	6.241	13.012
studytime	0.3822	0.608	0.628	0.530	-0.812	1.576
failures	-1.9320	0.912	-2.119	0.034	-3.722	-0.142
health	0.0085	0.340	0.025	0.980	-0.660	0.677
school_MS	-0.8621	1.080	-0.798	0.425	-2.983	1.259
sex_M	-0.1956	1.084	-0.180	0.857	-2.325	1.934
schoolsup_yes	-1.5326	1.497	-1.024	0.306	-4.472	1.406
higher_yes	3.6606	1.575	2.324	0.020	0.567	6.754
studytime:failures	-0.1000	0.328	-0.304	0.761	-0.745	0.545
studytime:health	-0.0002	0.097	-0.002	0.999	-0.191	0.190
studytime:school_MS	0.3593	0.310	1.159	0.247	-0.250	0.968
studytime:sex_M	-0.2732	0.280	-0.975	0.330	-0.823	0.277
studytime:schoolsup_yes	-0.6878	0.461	-1.493	0.136	-1.593	0.217
studytime:higher_yes	0.2073	0.508	0.408	0.683	-0.790	1.205
failures:health	0.2233	0.150	1.488	0.137	-0.071	0.518
failures:school_MS	-0.3565	0.412	-0.865	0.388	-1.166	0.453
failures:sex_M	0.4143	0.433	0.956	0.339	-0.437	1.265
failures:schoolsup_yes	1.5609	0.628	2.485	0.013	0.327	2.794
failures:higher_yes	-0.6825	0.437	-1.561	0.119	-1.541	0.176
health:school_MS	-0.0339	0.163	-0.208	0.835	-0.354	0.286
health:sex_M	0.0219	0.159	0.138	0.891	-0.291	0.334
health:schoolsup_yes	0.3179	0.266	1.195	0.233	-0.205	0.840
health:higher_yes	-0.2745	0.287	-0.956	0.339	-0.838	0.289
school_MS:sex_M	-0.1973	0.498	-0.396	0.692	-1.175	0.781
school_MS:schoolsup_yes	0.9170	0.951	0.964	0.335	-0.951	2.785

school_MS:higher_yes	-1.2122	0.778	-1.558	0.120	-2.740	0.316
sex_M:schoolsup_yes	-0.2618	0.836	-0.313	0.754	-1.903	1.379
sex_M:higher_yes	-0.0048	0.791	-0.006	0.995	-1.558	1.549

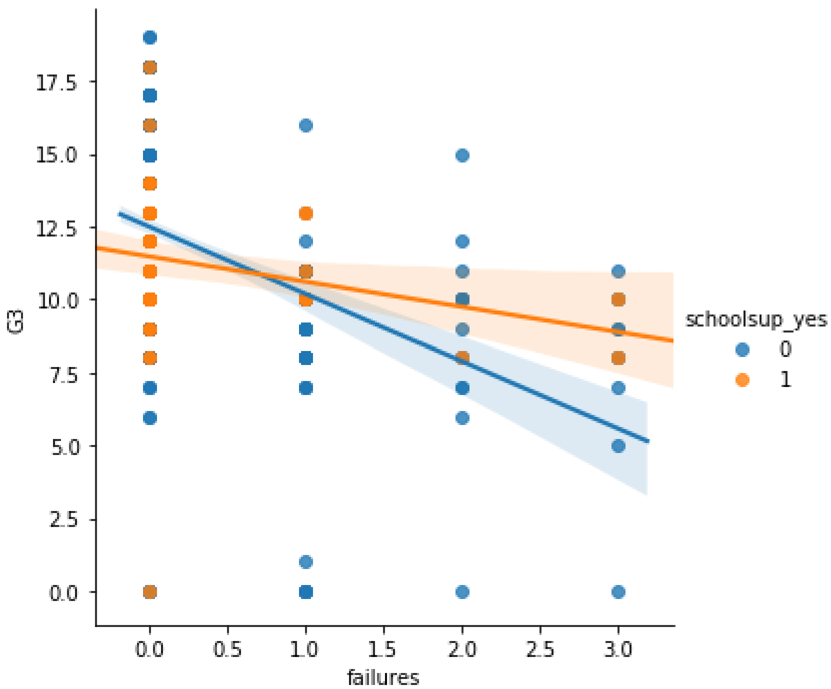
Omnibus:	116.763	**Durbin-Watson:**	1.895
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	351.956
Skew:	-0.865	**Prob(JB):**	3.75e-77
Kurtosis:	6.166	**Cond. No.**	235.

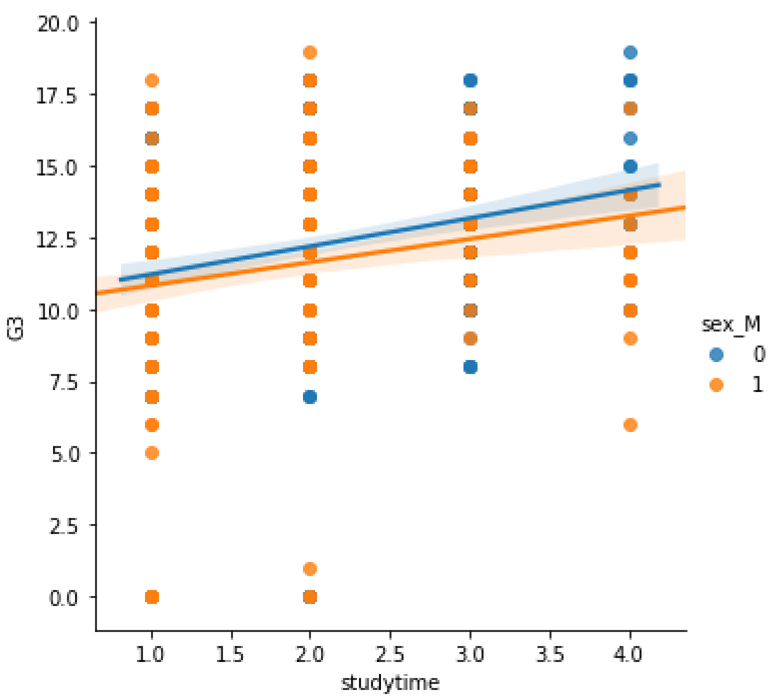
Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [21]:
```
# Interaction is present between failures*schoolsup_yes on G3
import seaborn
seaborn.lmplot(y='G3', x='failures', hue='schoolsup_yes', data=PX2)


# No interaction is present between studytime*sex_M on G3
seaborn.lmplot(y='G3', x='studytime', hue='sex_M', data=PX2)
```

Out[21]: <seaborn.axisgrid.FacetGrid at 0x184d938dcf8>

Out[21]: <seaborn.axisgrid.FacetGrid at 0x184d94177f0>





In [ ]:

In [ ]:

In [1]:
```
import numpy as np
import pandas as pd
from sklearn import preprocessing
from tensorflow import keras
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras import optimizers
import keras.utils as ker
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from keras.layers import Dense, InputLayer, Flatten, Dropout
import tensorflow as tf
from sklearn.metrics import classification_report, confusion_matrix, accuracy_
score, roc_auc_score, roc_curve, precision_score, recall_score, accuracy_score
, f1_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.utils import resample
```

```
Using TensorFlow backend.
```

In [2]:
```
## Read data from csv file 'student-mat.csv'
math_data = pd.read_csv('encoded_math_data.csv')
```

In [3]:
```
## Encoding Schools
list_of_schools = []
for i in math_data['school']:
    if i == 'GP':
        school = 1
    else:
        school = 0
    list_of_schools.append(school)

math_data['school'] = list_of_schools
```

In [4]:
```
## One-hot encoding binary variables.
school_one_hot = ker.to_categorical(math_data['school']).tolist()
sex_one_hot = ker.to_categorical(math_data['sex']).tolist()
address_one_hot = ker.to_categorical(math_data['address']).tolist()
pstatus_one_hot = ker.to_categorical(math_data['Pstatus']).tolist()
fedu_one_hot = ker.to_categorical(math_data['Fedu']).tolist()
medu_one_hot = ker.to_categorical(math_data['Medu']).tolist()
schoolsup_one_hot = ker.to_categorical(math_data['schoolsup']).tolist()
famsup_one_hot = ker.to_categorical(math_data['famsup']).tolist()
paid_one_hot = ker.to_categorical(math_data['paid']).tolist()
activities_one_hot = ker.to_categorical(math_data['activities']).tolist()
nursery_one_hot = ker.to_categorical(math_data['nursery']).tolist()
higher_one_hot = ker.to_categorical(math_data['higher']).tolist()
internet_one_hot = ker.to_categorical(math_data['internet']).tolist()
romantic_one_hot = ker.to_categorical(math_data['romantic']).tolist()

## Adding one-hot vectors to df
math_data['school_one_hot'] = school_one_hot
math_data['sex_one_hot'] = sex_one_hot
math_data['address_one_hot'] = address_one_hot
math_data['pstatus_one_hot'] = pstatus_one_hot
math_data['fedu_one_hot'] = fedu_one_hot
math_data['medu_one_hot'] = medu_one_hot
math_data['schoolsup_one_hot'] = schoolsup_one_hot
math_data['famsup_one_hot'] = famsup_one_hot
math_data['paid_one_hot'] = paid_one_hot
math_data['activities_one_hot'] = activities_one_hot
math_data['nursery_one_hot'] = nursery_one_hot
math_data['higher_one_hot'] = higher_one_hot
math_data['internet_one_hot'] = internet_one_hot
math_data['romantic_one_hot'] = romantic_one_hot
```

In [5]:
```
# Creating a new binary variable - 1 if student failed first grading period
previous_grade_list = []
for i in math_data['M1']:
    if i < 9.5: # Fail
        label = 1
    else: # Pass
        label = 0
    previous_grade_list.append(label)
math_data['previous_pass_fail'] = previous_grade_list
```

In [6]:
```
## Creating labels - Pass(0) or Fail(1)
list_of_labels = []
for i in math_data['M3']:
    if i < 9.5: # Fail
        label = 1
    else: # Pass
        label = 0
    list_of_labels.append(label)
math_data['label'] = list_of_labels
```

In [7]:
```
math_data['label'].value_counts()
```

Out[7]:
```
0    265
1    130
Name: label, dtype: int64
```

In [8]:
```
## Upsample the minority class to deal with the skewed dataset.
math_data_maj = math_data[math_data['label']==0]
math_data_min = math_data[math_data['label']==1]
math_data_min_upsampled = resample(math_data_min, replace=True, n_samples=265)
math_data_balanced = pd.concat([math_data_maj, math_data_min_upsampled])
math_data_balanced['label'].value_counts()
```

Out[8]:
```
1    265
0    265
Name: label, dtype: int64
```

In [9]:
```
math_data_balanced = math_data_balanced.reset_index(drop=True)
```

In [10]:
```
balanced_math_data = math_data_balanced.drop(math_data_balanced.columns[[0]],
axis=1)
```

In [11]:
```
## Creating input vector (X)
X = []
for i in range(0, len(balanced_math_data)):
    x = []
#     x.append(balanced_math_data['age'][i])
    x.append(balanced_math_data['Medu'][i])
    x.append(balanced_math_data['Fedu'][i])
#     x.append(balanced_math_data['both_parents_college'][i])
#     x.append(balanced_math_data['studytime'][i])
#     x.append(balanced_math_data['famrel'][i])
#     x.append(balanced_math_data['freetime'][i])
    x.append(balanced_math_data['goout'][i])
    x.append(balanced_math_data['Dalc'][i])
    x.append(balanced_math_data['Walc'][i])
#     x.append(balanced_math_data['health'][i])
#     x.append(balanced_math_data['m_absences'][i])
    x.append(balanced_math_data['failures'][i])

#     x.extend(balanced_math_data['sex_one_hot'][i])
#     x.extend(balanced_math_data['address_one_hot'][i])
#     x.extend(balanced_math_data['pstatus_one_hot'][i])
#     x.extend(balanced_math_data['schoolsup_one_hot'][i])
#     x.extend(balanced_math_data['famsup_one_hot'][i])
    x.extend(balanced_math_data['paid_one_hot'][i])
#     x.extend(balanced_math_data['activities_one_hot'][i])
#     x.extend(balanced_math_data['nursery_one_hot'][i])
#     x.extend(balanced_math_data['school_one_hot'][i])
    x.extend(balanced_math_data['higher_one_hot'][i])
    x.extend(balanced_math_data['internet_one_hot'][i])
    x.extend(balanced_math_data['romantic_one_hot'][i])
    x.append(balanced_math_data['previous_pass_fail'][i])
    x.append(balanced_math_data['M1'][i])
    X.append(x)
```

In [12]:
```
Y = np.array(balanced_math_data['label'])
X = np.array(X)
```

In [47]:
```
## split dataset into train-test.
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.25, stra
tify=Y)
```

In [14]:
```
X.shape[1:]
```

Out[14]:
```
(16,)
```

In [15]:
```
X_train[0]
```

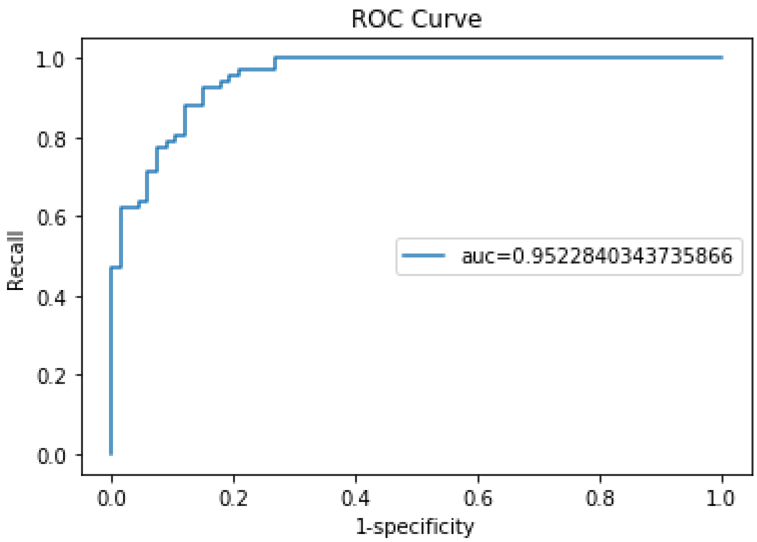
Out[15]:
```
array([ 3.,  2.,  2.,  1.,  1.,  0.,  0.,  1.,  0.,  1.,  0.,  1.,  1.,
        0.,  0., 14.])
```

In [16]:
```
## DNN model utilizing TF's Keras API
model = keras.models.Sequential()
model.add(keras.layers.InputLayer(input_shape=X.shape[1:]))
model.add(keras.layers.Dense(128, activation='sigmoid'))
model.add(keras.layers.Dense(128, activation='sigmoid'))
model.add(keras.layers.Dense(1, activation='sigmoid'))
model.compile(optimizer='adam',
              loss='binary_crossentropy')
model.summary()
model.fit(X_train, y_train, epochs=36, batch_size=1, validation_split=0.2)
Y_pred = model.predict_classes(X_test)

## Metrics -
print('\nPrecision score: {:.4f}'.format(precision_score(y_test, Y_pred)))
print('Recall score: {:.4f}'.format(recall_score(y_test, Y_pred)))
print('Accuracy score: {:.4f}'.format(accuracy_score(y_test, Y_pred)))
print('F1 score: {:.4f}'.format(f1_score(y_test, Y_pred)))

print('\nClassification accuracy report:')
print(classification_report(y_test, Y_pred))
print('\nConfusion matrix:')
print(confusion_matrix(y_test, Y_pred))

## Creating an ROC/AUC curve to visualize performance.
classification_probs = model.predict_proba(X_test)
classification_AUC = roc_auc_score(y_test, classification_probs)
print("\nAUC Index: {:.3f}".format(classification_AUC))
fpr, tpr, threshold = roc_curve(y_test,  classification_probs)
plt.plot(fpr,tpr,label="auc="+str(classification_AUC))
plt.legend(loc=5)
plt.ylabel('Recall')
plt.xlabel('1-specificity')
plt.title('ROC Curve')
plt.show()
```

```
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow_cor
e/python/ops/resource_variable_ops.py:1628: calling BaseResourceVariable.__in
it__ (from tensorflow.python.ops.resource_variable_ops) with constraint is de
precated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow_cor
e/python/ops/nn_impl.py:183: where (from tensorflow.python.ops.array_ops) is
deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 128)               2176
_________________________________________________________________
dense_1 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 129
=================================================================
Total params: 18,817
Trainable params: 18,817
Non-trainable params: 0
_________________________________________________________________
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow_cor
e/python/keras/optimizer_v2/optimizer_v2.py:460: BaseResourceVariable.constra
int (from tensorflow.python.ops.resource_variable_ops) is deprecated and will
be removed in a future version.
Instructions for updating:
Apply a constraint manually following the optimizer update step.
Train on 317 samples, validate on 80 samples
Epoch 1/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.5627 - val
_loss: 0.4635
Epoch 2/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3852 - val
_loss: 0.4482
Epoch 3/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3370 - val
_loss: 0.5854
Epoch 4/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3440 - val
_loss: 0.4768
Epoch 5/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3426 - val
_loss: 0.4928
Epoch 6/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3334 - val
_loss: 0.4799
Epoch 7/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3312 - val
_loss: 0.4940
Epoch 8/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3316 - val
_loss: 0.4912
Epoch 9/36
```

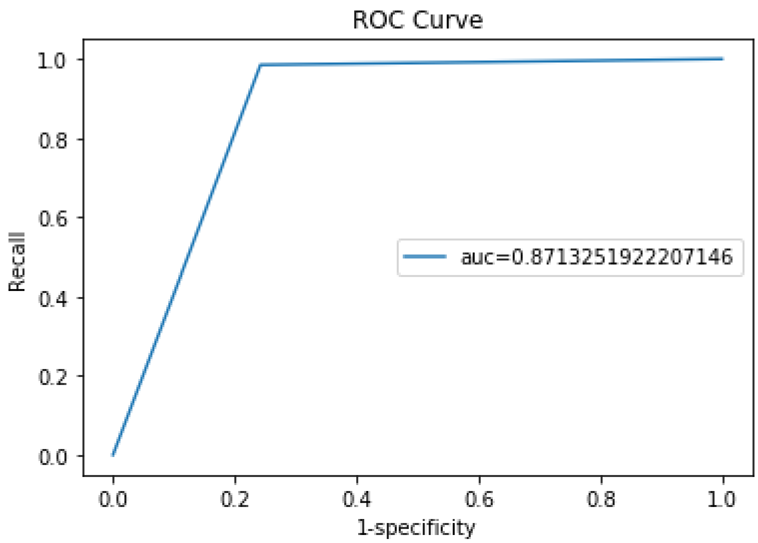
```
317/317 [==============================] - 1s 2ms/sample - loss: 0.3324 - val
_loss: 0.5050
Epoch 10/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3260 - val
_loss: 0.4934
Epoch 11/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3231 - val
_loss: 0.5045
Epoch 12/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3076 - val
_loss: 0.5042
Epoch 13/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3033 - val
_loss: 0.5136
Epoch 14/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3151 - val
_loss: 0.4786
Epoch 15/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3105 - val
_loss: 0.5101
Epoch 16/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2960 - val
_loss: 0.4789
Epoch 17/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3121 - val
_loss: 0.4739
Epoch 18/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3047 - val
_loss: 0.5283
Epoch 19/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3022 - val
_loss: 0.4994
Epoch 20/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3001 - val
_loss: 0.5151
Epoch 21/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2766 - val
_loss: 0.4916
Epoch 22/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2966 - val
_loss: 0.4903
Epoch 23/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2886 - val
_loss: 0.4862
Epoch 24/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2849 - val
_loss: 0.5587
Epoch 25/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.3142 - val
_loss: 0.4742
Epoch 26/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2890 - val
_loss: 0.4861
Epoch 27/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2829 - val
_loss: 0.4785
Epoch 28/36
```

```
317/317 [==============================] - 1s 2ms/sample - loss: 0.2870 - val
_loss: 0.4781
Epoch 29/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2876 - val
_loss: 0.5133
Epoch 30/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2795 - val
_loss: 0.4935
Epoch 31/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2866 - val
_loss: 0.4870
Epoch 32/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2685 - val
_loss: 0.4974
Epoch 33/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2769 - val
_loss: 0.4997
Epoch 34/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2849 - val
_loss: 0.4898
Epoch 35/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2666 - val
_loss: 0.5010
Epoch 36/36
317/317 [==============================] - 1s 2ms/sample - loss: 0.2727 - val
_loss: 0.4951

Precision score: 0.8267
Recall score: 0.9394
Accuracy score: 0.8722
F1 score: 0.8794

Classification accuracy report:
              precision    recall  f1-score   support

           0       0.93      0.81      0.86        67
           1       0.83      0.94      0.88        66

   micro avg       0.87      0.87      0.87       133
   macro avg       0.88      0.87      0.87       133
weighted avg       0.88      0.87      0.87       133


Confusion matrix:
[[54 13]
 [ 4 62]]

AUC Index: 0.952
```


ROC Curve
1.0
0.8
0.6
0.4
0.2
0.0
Recall
auc=0.9522840343735866
0.0
0.2
0.4
0.6
0.8
1.0
1-specificity

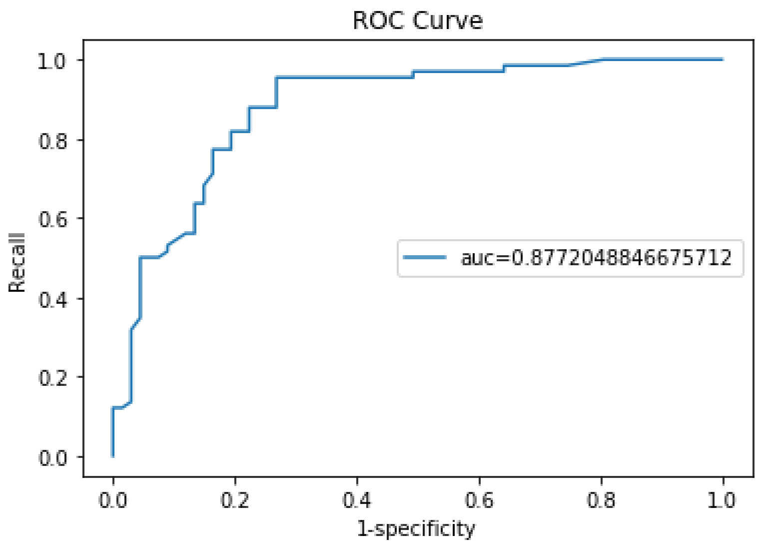
In [27]:
```
## Tuning hyperparameters of tree - cross-validated grid-search over a paramet
er grid.
optimized_tree = DecisionTreeClassifier()
params = {"max_depth": range(1,10),
           "min_samples_split": range(2,10,1),
           "max_leaf_nodes": range(2,5)}

opt_tree = GridSearchCV(optimized_tree, params, cv=5) ##  folds in stratified
 k-fold.
opt_tree.fit(X_train,y_train)
print("Best Parameters:", opt_tree.best_params_)

## Grid Search Tree Metrics
grid_tree_y_pred = opt_tree.predict(X_test)
grid_tree_probs = opt_tree.predict_proba(X_test)
grid_tree_AUC = roc_auc_score(y_test, grid_tree_probs[:, 1])  ## Probability h
ere just like lecture notes.

print('\nPrecision score: {:.4f}'.format(precision_score(y_test, grid_tree_y_p
red)))
print('Recall score: {:.4f}'.format(recall_score(y_test, grid_tree_y_pred)))
print('Accuracy score: {:.4f}'.format(accuracy_score(y_test, grid_tree_y_pred
)))
print('F1 score: {:.4f}'.format(f1_score(y_test, grid_tree_y_pred)))

print("\nAUC Index:", grid_tree_AUC)
fpr, tpr, threshold = roc_curve(y_test,  grid_tree_probs[:, 1])
plt.plot(fpr,tpr,label="auc="+str(grid_tree_AUC))
plt.legend(loc=5)
plt.ylabel('Recall')
plt.xlabel('1-specificity')
plt.title('ROC Curve')
plt.show()
```

```
Best Parameters: {'max_depth': 1, 'max_leaf_nodes': 2, 'min_samples_split':
2}

Precision score: 0.8049
Recall score: 0.9851
Accuracy score: 0.8722
F1 score: 0.8859

AUC Index: 0.8713251922207146
```



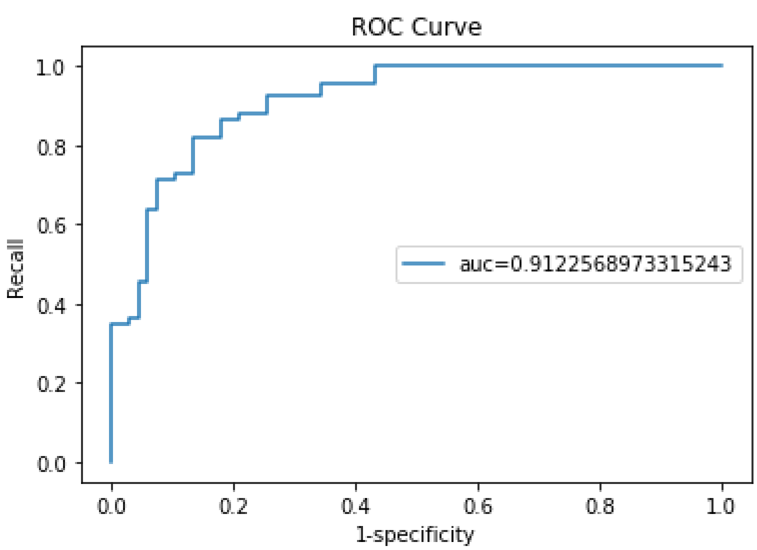
In [38]:
```
## Random Forest - cross-validated grid-search over a parameter grid.
rf = RandomForestClassifier(n_estimators=100, n_jobs=-1, bootstrap=True)
params = {"max_depth": range(1,10),
          "min_samples_split": range(2,10,1),
          "max_leaf_nodes": range(2,5)}

opt_rf = GridSearchCV(rf, params)
opt_rf.fit(X_train,y_train)
print("Best Parameters:", opt_rf.best_params_)

rf_y_pred = opt_rf.predict(X_test)
rf_probs = opt_rf.predict_proba(X_test)

## Metrics
print('Precision score: {:.4f}'.format(precision_score(y_test,rf_y_pred)))
print('Recall score: {:.4f}'.format(recall_score(y_test,rf_y_pred)))
print('Accuracy score: {:.4f}'.format(accuracy_score(y_test,rf_y_pred)))
print('F1 score: {:.4f}'.format(f1_score(y_test,rf_y_pred)))

rf_AUC = roc_auc_score(y_test, rf_probs[:, 1])
print("\nAUC Index:", rf_AUC)
fpr, tpr, threshold = roc_curve(y_test,  rf_probs[:, 1])
plt.plot(fpr,tpr,label="auc="+str(rf_AUC))
plt.legend(loc=5)
plt.ylabel('Recall')
plt.xlabel('1-specificity')
plt.title('ROC Curve')
plt.show()
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/model_selection/_split.py:205
3: FutureWarning: You should specify a value for 'cv' instead of relying on t
he default value. The default value will change from 3 to 5 in version 0.22.
  warnings.warn(CV_WARNING, FutureWarning)

Best Parameters: {'max_depth': 8, 'max_leaf_nodes': 2, 'min_samples_split':
2}
Precision score: 0.7733
Recall score: 0.8788
Accuracy score: 0.8120
F1 score: 0.8227

AUC Index: 0.8772048846675712
```



In [48]:
```
## Logistic Regression
log_regression = LogisticRegression().fit(X_train, y_train)
logistic_y_pred = log_regression.predict(X_test)
log_probs = log_regression.predict_proba(X_test)

## Metrics
print('Precision score: {:.4f}'.format(precision_score(y_test,logistic_y_pred
)))
print('Recall score: {:.4f}'.format(recall_score(y_test,logistic_y_pred)))
print('Accuracy score: {:.4f}'.format(accuracy_score(y_test,logistic_y_pred)))
print('F1 score: {:.4f}'.format(f1_score(y_test,logistic_y_pred)))

log_AUC = roc_auc_score(y_test, log_probs[:, 1])
print("\nAUC Index:", log_AUC)
fpr, tpr, threshold = roc_curve(y_test,  log_probs[:, 1])
plt.plot(fpr,tpr,label="auc="+str(log_AUC))
plt.legend(loc=5)
plt.ylabel('Recall')
plt.xlabel('1-specificity')
plt.title('ROC Curve')
plt.show()
```

```
Precision score: 0.8308
Recall score: 0.8182
Accuracy score: 0.8271
F1 score: 0.8244

AUC Index: 0.9122568973315243

/opt/conda/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433:
FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a s
olver to silence this warning.
  FutureWarning)
```



In [1]:
```
import numpy as np
import pandas as pd
from sklearn import preprocessing
from tensorflow import keras
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras import optimizers
import keras.utils as ker
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from keras.models import Sequential
from keras.layers import Dense, InputLayer, Flatten, Dropout
import tensorflow as tf
from sklearn.metrics import classification_report, confusion_matrix, accuracy_
score, roc_auc_score, roc_curve, precision_score, recall_score, accuracy_score
, f1_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import LinearSVC, SVC
from sklearn.utils import resample
```

```
Using TensorFlow backend.
```

In [2]:
```
## Read data from csv file 'student-por.csv'
por_data = pd.read_csv('encoded_por_data.csv')
```

In [3]:
```
## Encoding Schools
list_of_schools = []
for i in por_data['school']:
    if i == 'GP':
        school = 1
    else:
        school = 0
    list_of_schools.append(school)

por_data['school'] = list_of_schools
```

In [4]:
```
## One-hot encoding binary variables.
school_one_hot = ker.to_categorical(por_data['school']).tolist()
sex_one_hot = ker.to_categorical(por_data['sex']).tolist()
address_one_hot = ker.to_categorical(por_data['address']).tolist()
pstatus_one_hot = ker.to_categorical(por_data['Pstatus']).tolist()
fedu_one_hot = ker.to_categorical(por_data['Fedu']).tolist()
medu_one_hot = ker.to_categorical(por_data['Medu']).tolist()
schoolsup_one_hot = ker.to_categorical(por_data['schoolsup']).tolist()
famsup_one_hot = ker.to_categorical(por_data['famsup']).tolist()
paid_one_hot = ker.to_categorical(por_data['paid']).tolist()
activities_one_hot = ker.to_categorical(por_data['activities']).tolist()
nursery_one_hot = ker.to_categorical(por_data['nursery']).tolist()
higher_one_hot = ker.to_categorical(por_data['higher']).tolist()
internet_one_hot = ker.to_categorical(por_data['internet']).tolist()
romantic_one_hot = ker.to_categorical(por_data['romantic']).tolist()

## Adding one-hot vectors to df
por_data['school_one_hot'] = school_one_hot
por_data['sex_one_hot'] = sex_one_hot
por_data['address_one_hot'] = address_one_hot
por_data['pstatus_one_hot'] = pstatus_one_hot
por_data['fedu_one_hot'] = fedu_one_hot
por_data['medu_one_hot'] = medu_one_hot
por_data['schoolsup_one_hot'] = schoolsup_one_hot
por_data['famsup_one_hot'] = famsup_one_hot
por_data['paid_one_hot'] = paid_one_hot
por_data['activities_one_hot'] = activities_one_hot
por_data['nursery_one_hot'] = nursery_one_hot
por_data['higher_one_hot'] = higher_one_hot
por_data['internet_one_hot'] = internet_one_hot
por_data['romantic_one_hot'] = romantic_one_hot
```

In [5]:
```
# Creating a new binary variable - 1 if student failed first grading period
previous_grade_list = []
for i in por_data['P1']:
    if i < 9.5: # Fail
        label = 1
    else: # Pass
        label = 0
    previous_grade_list.append(label)
por_data['previous_pass_fail'] = previous_grade_list
```

In [6]:
```
## Creating labels - Pass(0) or Fail(1)
list_of_labels = []
for i in por_data['P3']:
    if i < 9.5: # Fail
        label = 1
    else: # Pass
        label = 0
    list_of_labels.append(label)
por_data['label'] = list_of_labels
```

In [7]:
```
por_data['label'].value_counts()
```

Out[7]:
```
0    549
1    100
Name: label, dtype: int64
```

In [8]:
```
## Upsample the minority class to deal with the skewed dataset.
por_data_maj = por_data[por_data['label']==0]
por_data_min = por_data[por_data['label']==1]
por_data_min_upsampled = resample(por_data_min, replace=True, n_samples=549)
por_data_balanced = pd.concat([por_data_maj, por_data_min_upsampled])
por_data_balanced['label'].value_counts()
```

Out[8]:
```
1    549
0    549
Name: label, dtype: int64
```

In [9]:
```
por_data_balanced = por_data_balanced.reset_index(drop=True)
balanced_por_data = por_data_balanced.drop(por_data_balanced.columns[[0]], axi
s=1)
```

In [10]:
```
## Creating input vector (X)
X = []
for i in range(0, len(balanced_por_data)):
    x = []
#     x.append(balanced_por_data['age'][i])
    x.append(balanced_por_data['Medu'][i])
    x.append(balanced_por_data['Fedu'][i])
#     x.append(balanced_por_data['both_parents_college'][i])
#     x.append(balanced_por_data['studytime'][i])
#     x.append(balanced_por_data['famrel'][i])
#     x.append(balanced_por_data['freetime'][i])
    x.append(balanced_por_data['goout'][i])
    x.append(balanced_por_data['Dalc'][i])
    x.append(balanced_por_data['Walc'][i])
#     x.append(balanced_por_data['health'][i])
#     x.append(balanced_por_data['m_absences'][i])
    x.append(balanced_por_data['failures'][i])

#     x.extend(balanced_por_data['sex_one_hot'][i])
#     x.extend(balanced_por_data['address_one_hot'][i])
#     x.extend(balanced_por_data['pstatus_one_hot'][i])
#     x.extend(balanced_por_data['schoolsup_one_hot'][i])
#     x.extend(balanced_por_data['famsup_one_hot'][i])
    x.extend(balanced_por_data['paid_one_hot'][i])
#     x.extend(balanced_por_data['activities_one_hot'][i])
#     x.extend(balanced_por_data['nursery_one_hot'][i])
#     x.extend(balanced_por_data['school_one_hot'][i])
    x.extend(balanced_por_data['higher_one_hot'][i])
    x.extend(balanced_por_data['internet_one_hot'][i])
    x.extend(balanced_por_data['romantic_one_hot'][i])
    x.append(balanced_por_data['previous_pass_fail'][i])
    x.append(balanced_por_data['P1'][i])
    X.append(x)
```

In [11]:
```
Y = np.array(balanced_por_data['label'])
X = np.array(X)
```

In [41]:
```
## split dataset into train-test.
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.25, stra
tify=Y)
```

In [22]:
```
## DNN model utilizing TF's Keras API
model = keras.models.Sequential()
model.add(keras.layers.InputLayer(input_shape=X.shape[1:]))
model.add(keras.layers.Dense(128, activation='sigmoid'))
model.add(keras.layers.Dense(128, activation='sigmoid'))
model.add(keras.layers.Dense(1, activation='sigmoid'))
model.compile(optimizer='adam',
              loss='binary_crossentropy')
model.summary()
model.fit(X_train, y_train, epochs=36, batch_size=1, validation_split=0.2)
Y_pred = model.predict_classes(X_test)

## Metrics -
print('\nPrecision score: {:.4f}'.format(precision_score(y_test, Y_pred)))
print('Recall score: {:.4f}'.format(recall_score(y_test, Y_pred)))
print('Accuracy score: {:.4f}'.format(accuracy_score(y_test, Y_pred)))
print('F1 score: {:.4f}'.format(f1_score(y_test, Y_pred)))

print('\nClassification accuracy report:')
print(classification_report(y_test, Y_pred))
print('\nConfusion matrix:')
print(confusion_matrix(y_test, Y_pred))

## Creating an ROC/AUC curve to visualize performance.
classification_probs = model.predict_proba(X_test)
classification_AUC = roc_auc_score(y_test, classification_probs)
print("\nAUC Index: {:.3f}".format(classification_AUC))
fpr, tpr, threshold = roc_curve(y_test,  classification_probs)
plt.plot(fpr,tpr,label="auc="+str(classification_AUC))
plt.legend(loc=5)
plt.ylabel('Recall')
plt.xlabel('1-specificity')
plt.title('ROC Curve')
plt.show()
```

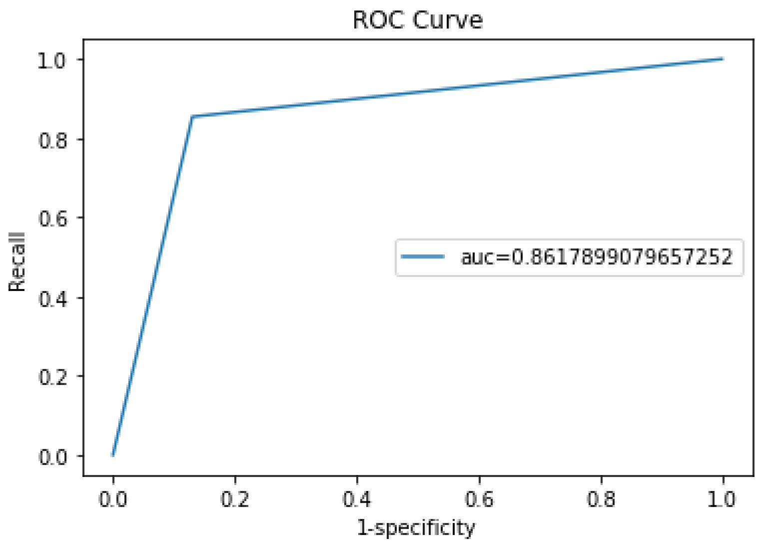
```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_6 (Dense)              (None, 128)               2176      
_________________________________________________________________
dense_7 (Dense)              (None, 128)               16512     
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 129       
=================================================================
Total params: 18,817
Trainable params: 18,817
Non-trainable params: 0
_________________________________________________________________
Train on 658 samples, validate on 165 samples
Epoch 1/36
658/658 [==============================] - 1s 2ms/sample - loss: 0.4472 - val
_loss: 0.3657
Epoch 2/36
658/658 [==============================] - 1s 2ms/sample - loss: 0.3166 - val
_loss: 0.3488
Epoch 3/36
 53/658 [=>............................] - ETA: 1s - loss: 0.2759

ERROR:root:Internal Python error in the inspect module.
Below is the traceback from this internal error.
```

```
Traceback (most recent call last):
  File "/opt/conda/lib/python3.7/site-packages/IPython/core/interactiveshell.
py", line 3326, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
  File "<ipython-input-22-ce763311a399>", line 10, in <module>
    model.fit(X_train, y_train, epochs=36, batch_size=1, validation_split=0.
2)
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/python/keras/e
ngine/training.py", line 703, in fit
    use_multiprocessing=use_multiprocessing)
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/python/keras/e
ngine/training_arrays.py", line 669, in fit
    steps_name='steps_per_epoch')
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/python/keras/e
ngine/training_arrays.py", line 388, in model_iteration
    batch_outs = f(ins_batch)
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/python/keras/b
ackend.py", line 3356, in __call__
    return nest.map_structure(self._eval_if_composite, output_structure)
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/python/util/ne
st.py", line 524, in map_structure
    structure[0], [func(*x) for x in entries],
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/python/util/ne
st.py", line 524, in <listcomp>
    structure[0], [func(*x) for x in entries],
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/python/keras/b
ackend.py", line 3301, in _eval_if_composite
    if isinstance(tensor, composite_tensor.CompositeTensor):
  File "/opt/conda/lib/python3.7/abc.py", line 139, in __instancecheck__
    return _abc_instancecheck(cls, instance)
KeyboardInterrupt

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/opt/conda/lib/python3.7/site-packages/IPython/core/interactiveshell.
py", line 2040, in showtraceback
    stb = value._render_traceback_()
AttributeError: 'KeyboardInterrupt' object has no attribute '_render_tracebac
k_'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/opt/conda/lib/python3.7/site-packages/IPython/core/ultratb.py", line
1101, in get_records
    return _fixed_getinnerframes(etb, number_of_lines_of_context, tb_offset)
  File "/opt/conda/lib/python3.7/site-packages/IPython/core/ultratb.py", line
319, in wrapped
    return f(*args, **kwargs)
  File "/opt/conda/lib/python3.7/site-packages/IPython/core/ultratb.py", line
353, in _fixed_getinnerframes
    records = fix_frame_records_filenames(inspect.getinnerframes(etb, contex
t))
  File "/opt/conda/lib/python3.7/inspect.py", line 1502, in getinnerframes
    frameinfo = (tb.tb_frame,) + getframeinfo(tb, context)
  File "/opt/conda/lib/python3.7/inspect.py", line 1460, in getframeinfo
```

```
    filename = getsourcefile(frame) or getfile(frame)
  File "/opt/conda/lib/python3.7/inspect.py", line 696, in getsourcefile
    if getattr(getmodule(object, filename), '__loader__', None) is not None:
  File "/opt/conda/lib/python3.7/inspect.py", line 733, in getmodule
    if ismodule(module) and hasattr(module, '__file__'):
  File "/opt/conda/lib/python3.7/site-packages/tensorflow/__init__.py", line
50, in __getattr__
    module = self._load()
  File "/opt/conda/lib/python3.7/site-packages/tensorflow/__init__.py", line
44, in _load
    module = _importlib.import_module(self.__name__)
  File "/opt/conda/lib/python3.7/importlib/__init__.py", line 127, in import_
module
    return _bootstrap._gcd_import(name[level:], package, level)
  File "<frozen importlib._bootstrap>", line 1006, in _gcd_import
  File "<frozen importlib._bootstrap>", line 983, in _find_and_load
  File "<frozen importlib._bootstrap>", line 967, in _find_and_load_unlocked
  File "<frozen importlib._bootstrap>", line 677, in _load_unlocked
  File "<frozen importlib._bootstrap_external>", line 728, in exec_module
  File "<frozen importlib._bootstrap>", line 219, in _call_with_frames_remove
d
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/contrib/__init
__.py", line 54, in <module>
    from tensorflow.contrib import gan
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/contrib/gan/__
init__.py", line 28, in <module>
    from tensorflow.contrib.gan.python import estimator
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/contrib/gan/__
init__.py", line 28, in <module>
    from tensorflow.contrib.gan.python import estimator
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/contrib/gan/py
thon/estimator/__init__.py", line 27, in <module>
    from tensorflow.contrib.gan.python.estimator.python import gan_estimator
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/contrib/gan/py
thon/estimator/python/gan_estimator.py", line 21, in <module>
    from tensorflow.contrib.gan.python.estimator.python import gan_estimator_
impl
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/contrib/gan/py
thon/estimator/python/gan_estimator_impl.py", line 26, in <module>
    from tensorflow.contrib.gan.python import train as tfgan_train
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/contrib/gan/py
thon/train.py", line 38, in <module>
    from tensorflow.contrib.slim.python.slim import learning as slim_learning
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/contrib/slim/_
_init__.py", line 37, in <module>
    from tensorflow.contrib.slim.python.slim import summaries
  File "/opt/conda/lib/python3.7/site-packages/tensorflow_core/contrib/slim/p
ython/slim/summaries.py", line 31, in <module>
    """
KeyboardInterrupt

---------------------------------------------------------------------------
```

In [23]:
```
## Tuning hyperparameters of tree - cross-validated grid-search over a paramet
er grid.
optimized_tree = DecisionTreeClassifier()
params = {"max_depth": range(1,10),
           "min_samples_split": range(2,10,1),
           "max_leaf_nodes": range(2,5)}

opt_tree = GridSearchCV(optimized_tree, params, cv=5) ##  folds in stratified
 k-fold.
opt_tree.fit(X_train,y_train)
print("Best Parameters:", opt_tree.best_params_)

## Grid Search Tree Metrics
grid_tree_y_pred = opt_tree.predict(X_test)
grid_tree_probs = opt_tree.predict_proba(X_test)
grid_tree_AUC = roc_auc_score(y_test, grid_tree_probs[:, 1])  ## Probability h
ere just like lecture notes.

print('\nPrecision score: {:.4f}'.format(precision_score(y_test, grid_tree_y_p
red)))
print('Recall score: {:.4f}'.format(recall_score(y_test, grid_tree_y_pred)))
print('Accuracy score: {:.4f}'.format(accuracy_score(y_test, grid_tree_y_pred
)))
print('F1 score: {:.4f}'.format(f1_score(y_test, grid_tree_y_pred)))

print("\nAUC Index:", grid_tree_AUC)
fpr, tpr, threshold = roc_curve(y_test,  grid_tree_probs[:, 1])
plt.plot(fpr,tpr,label="auc="+str(grid_tree_AUC))
plt.legend(loc=5)
plt.ylabel('Recall')
plt.xlabel('1-specificity')
plt.title('ROC Curve')
plt.show()
```

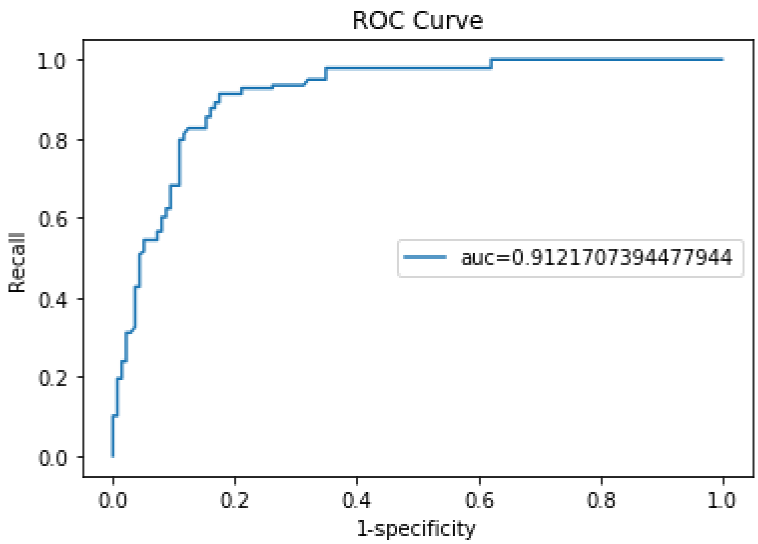
```
Best Parameters: {'max_depth': 1, 'max_leaf_nodes': 2, 'min_samples_split':
2}

Precision score: 0.8667
Recall score: 0.8540
Accuracy score: 0.8618
F1 score: 0.8603

AUC Index: 0.8617899079657252
```



In [32]:
```
## Random Forest - cross-validated grid-search over a parameter grid.
rf = RandomForestClassifier(n_estimators=100, n_jobs=-1, bootstrap=True)
params = {"max_depth": range(1,10),
           "min_samples_split": range(2,10,1),
           "max_leaf_nodes": range(2,5)}

opt_rf = GridSearchCV(rf, params)
opt_rf.fit(X_train,y_train)
print("Best Parameters:", opt_rf.best_params_)

rf_y_pred = opt_rf.predict(X_test)
rf_probs = opt_rf.predict_proba(X_test)

## Metrics
print('Precision score: {:.4f}'.format(precision_score(y_test,rf_y_pred)))
print('Recall score: {:.4f}'.format(recall_score(y_test,rf_y_pred)))
print('Accuracy score: {:.4f}'.format(accuracy_score(y_test,rf_y_pred)))
print('F1 score: {:.4f}'.format(f1_score(y_test,rf_y_pred)))

rf_AUC = roc_auc_score(y_test, rf_probs[:, 1])
print("\nAUC Index:", rf_AUC)
fpr, tpr, threshold = roc_curve(y_test,  rf_probs[:, 1])
plt.plot(fpr,tpr,label="auc="+str(rf_AUC))
plt.legend(loc=5)
plt.ylabel('Recall')
plt.xlabel('1-specificity')
plt.title('ROC Curve')
plt.show()
```

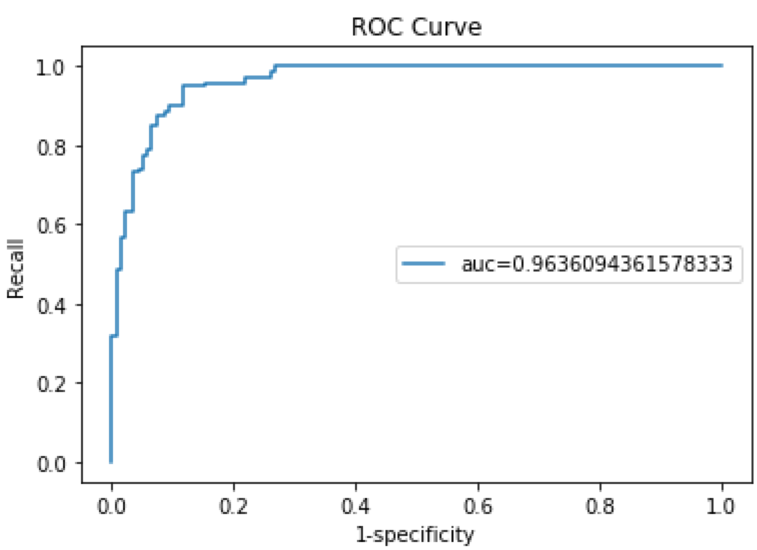
```
/opt/conda/lib/python3.7/site-packages/sklearn/model_selection/_split.py:205
3: FutureWarning: You should specify a value for 'cv' instead of relying on t
he default value. The default value will change from 3 to 5 in version 0.22.
  warnings.warn(CV_WARNING, FutureWarning)
```

```
Best Parameters: {'max_depth': 3, 'max_leaf_nodes': 3, 'min_samples_split':
4}
Precision score: 0.8429
Recall score: 0.8551
Accuracy score: 0.8473
F1 score: 0.8489

AUC Index: 0.9121707394477944
```



In [42]:
```python
## Logistic Regression
log_regression = LogisticRegression().fit(X_train, y_train)
logistic_y_pred = log_regression.predict(X_test)
log_probs = log_regression.predict_proba(X_test)

## Metrics
print('Precision score: {:.4f}'.format(precision_score(y_test,logistic_y_pred
)))
print('Recall score: {:.4f}'.format(recall_score(y_test,logistic_y_pred)))
print('Accuracy score: {:.4f}'.format(accuracy_score(y_test,logistic_y_pred)))
print('F1 score: {:.4f}'.format(f1_score(y_test,logistic_y_pred)))

log_AUC = roc_auc_score(y_test, log_probs[:, 1])
print("\nAUC Index:", log_AUC)
fpr, tpr, threshold = roc_curve(y_test,  log_probs[:, 1])
plt.plot(fpr,tpr,label="auc="+str(log_AUC))
plt.legend(loc=5)
plt.ylabel('Recall')
plt.xlabel('1-specificity')
plt.title('ROC Curve')
plt.show()
```

```
Precision score: 0.8873
Recall score: 0.9130
Accuracy score: 0.8982
F1 score: 0.9000

AUC Index: 0.9636094361578333

/opt/conda/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433:
FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a s
olver to silence this warning.
  FutureWarning)
```



In [ ]: