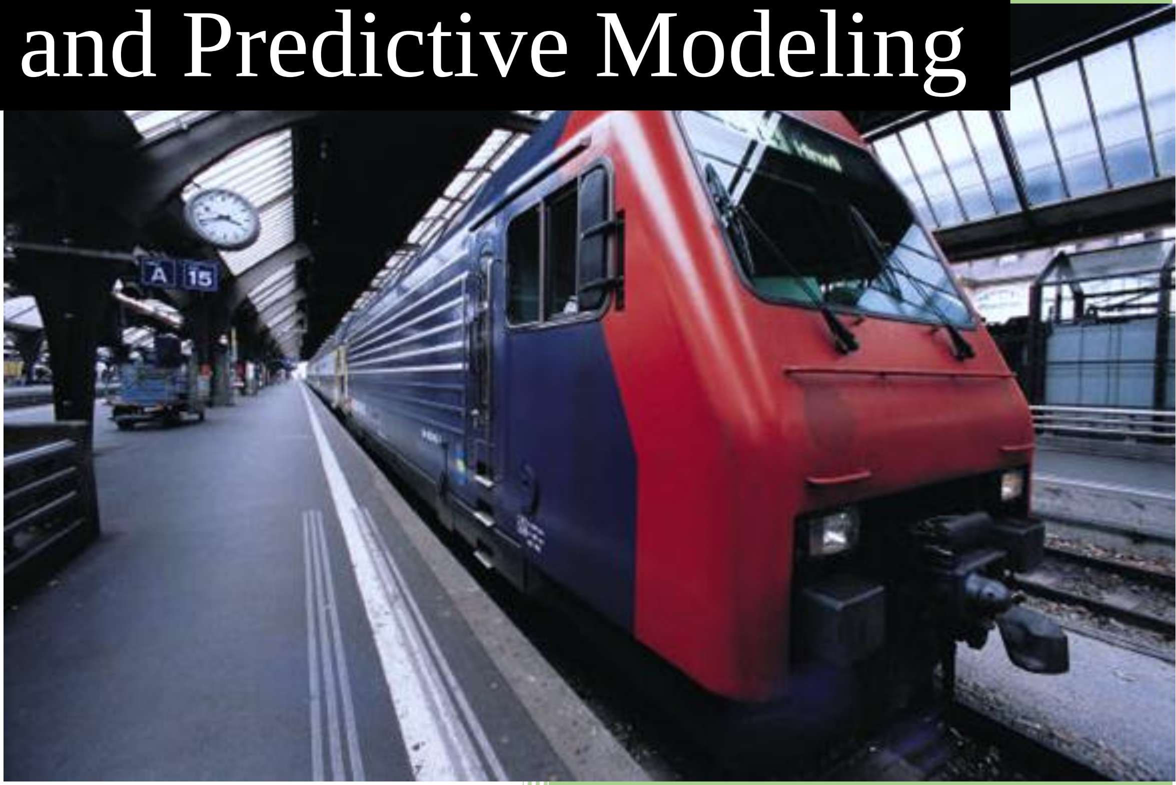
Student Performance Descriptive and Predictive Modeling



Ramadan Gannud

DSC 478

Spring 2019

Executive Summary

Education has played an important role in changing our society and shaping our future. In this project report, the data approaches student achievements in secondary education of two Portuguese schools. The data attributes include student grades, demographic, social and school related features. Furthermore, it was collected by using school reports and questionnaires.

The main goal of this project is to use Knowledge Discovery in Databases process to discover interesting patterns and extract communalities among students in order to predict their final grades. This project will include data explorations using statistical approaches to provide an overview of the data characteristics. In addition, data visualization techniques will be used to provide exploratory graphs about students' lifestyle in relevance to school.

Supervised learning using different Regression techniques will then be used to build models that predict student performances using their final scores in accordance to different variables. These models would train themselves using this dataset in order to anticipate future students' results. These models have been optimized using different optimization techniques and enhanced parameters. Moreover, Linear, Lasso and Ridge Regressions were used to come up with different models that answer the same question and to check for different results. Feature Selection process has been implemented to obtain the most informative features and predictors that explain results the most using their weights.

These methods were able to build different models that have high accuracy, the average error was used to determine and enhance accuracies of different models. The high accuracy of these models can produce better predictions on future data. This shows that students' performances are much related to their attributes and different factors could affects students' final results according to our models and exploration analyses and results.

In future, different variables could be added to the database using more information about students and schools. Different financial aspects in terms of budgeting and expenditures for different cities and states could help disclose more patterns and interesting communalities between students and these variables. Furthermore, student enrollments for different grades and number of students in classes could potentially impact students' performances. These organizational aspects could help schools and educational institutions improve their systems and quality of education.

Abstract

Higher education standards for the 21st century continues to promote discoveries in the field through learning analytics (LA). The purpose of this study is to explore different factors that could impact students' performance at their final grades. This study investigated which predictors could help us building model that predict better using different statistical methods and techniques. Linear Regression, Feature Selection, Ridge Regression, and Lasso Regression techniques will be used in this project to perform predictive analyses.

Recent data was collected using school reports and questionnaires. The results show that these methods can provide models with high accuracy for students' performances. This asserts that students' results are affected by previous achievements and impacted by different relevant features. As a result, more effective tools could be created to develop the quality of education by studying different factors that enhance student and organizational performances.

Introduction

In this Student Performance Dataset, different social and demographic variables were provided. The study has administered two public schools in Portugal. Data is sourced from the Web. This dataset consists of 33 variables and 649 observations. There are 16 numeric variables, 13 binary variables, and 4 nominal variables. Since our aimed predictive model is supposed to anticipate students' final scores, this attribute will be subset from the dataset. There were no issues of missing values.

Linear Regression was applied to the Students Performance Dataset to explore the features thought to be very important in improving the students' performance. Also, a model can be built which is able to predict students' final scores based on this method. Linear Regression was performed in Python using Jupyter Notebook in addition to different Libraries. Descriptive analysis was important since it showed correlation and covariance for students' performances and checked for multicollinearities. Ridge Regression and Lasso Regression were also applied to see if these two important techniques would make any differences and improve the accuracy of the prediction model. These two regressors push estimated coefficients towards zero and help reduce overfitting.

Methodology

Linear Regression Analysis:

The Linear Regression analysis technique was used on the education dataset using python. The standard linear regression on data main purpose was to learn more about the relationships between different independent (Predictor) variables and the dependent (Response) variable. It fits a straight line to a number of points so that the squared standard deviations of the observed values from that line are minimized. It's also known as the least square's estimation method. The line is basically a two-dimensional space that is define by the equation **Y= a + b*X.** Here, Y is the dependent variable which can be expressed in terms of the constant (a) and a slope (b) times the X as the dependent variable. In case of multivariate analysis, there exists to be more than one predicting or X variables. Thus, the line could not be visualized in 2d space and can be computed in the exact manner. Generally, the multiple regression will estimate a linear equation of the form:

$$\mathbf{Y = a + b_1{*}X_1 + b_2{*}X_2 + \ldots + b_p{*}X_p.}$$

The multiple regression analysis represents the independent contributions of each predicting variable to the prediction of response variables. Another way to represent it is if a dependent variable Y is correlated to independent variable X, the amount of change in Y could be interpreted by each individual independent variable X by having other variables constant.

Ridge Regression:

Ridge regression avoids the problem of overfitting and the failure to find unique solutions. In Ridge regression, the meta-parameter is called alpha L2. The penalty is the sum of the squares of the coefficients. It does not require unbiased estimators because it works in part. Thus, its variances are more accurate. It can reduce the variances with an increasing bias. Furthermore, ridge regression makes sure estimates are reasonably reliable to true values by adding just enough bias.

Since ridge shrinks the coefficients towards zero but not to exact zero, all parameters have to be chosen or none of them. Even though ridge doesn't preform variable selection, it still performs well for large multivariate data with number of predictors that are larger than the number of records. Ridge regression estimator is good at improving the least-squares when there is multicollinearity. Therefore, it can improve predictive performance.

Lasso Regression:

Lasso regression allow you to use complex models and avoid overfitting as well. The estimated coefficients are pushed to zero for regularization purposes. In Lasso regression, the meta-parameter is called lambda L1. Less important predictors will be set to zero to help choosing the predictors that can be left for the model. The penalty used is the sum of the absolute values of the coefficients. Lasso does both parameter shrinkage and variable selection.

Although it's better than the usual methods of automatic variable selection in terms of results, it still can produce models that make no sense. Also, it ignores nonsignificant variables that may be important and doesn't follow hierarchy principle. It lets you avoid thinking because it's automatic. Both methods can be combined for further results in Elastic Net regularization with alpha as the regularization strength and lambda as the desired sparseness of results.

Discussion and Results

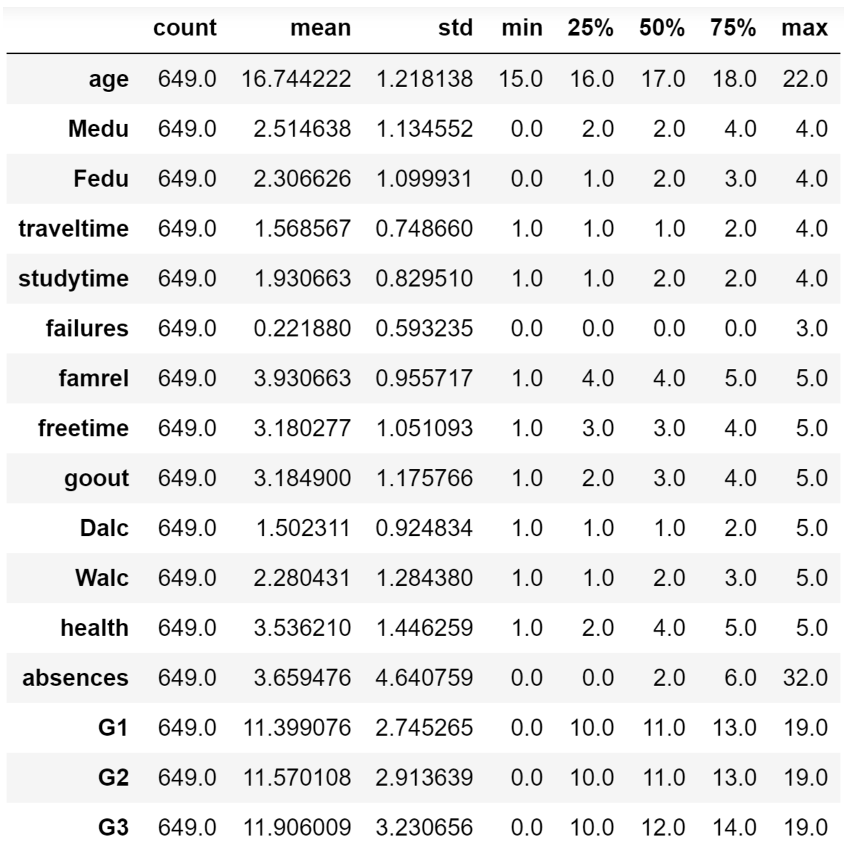
Data Structure:

Data has been imported to python environment using Jupyter Notebook and the text file was loaded via *read_csv*. The following table is a brief description of data features and their meanings.

name	**type**	**description**
school	string	student's school (binary: "GP" Gabriel Pereira or "MS" Mousinho da Silveira)
sex	string	student's sex (binary: "F" female or "M" male)
age	integer	student's age (numeric: from 15 to 22)
address	string	student's home address type (binary: "U" urban or "R" rural)
famsize	string	family size (binary: "LE3" less or equal to 3 or "GT3" greater than 3)
Pstatus	string	parent's cohabitation status (binary: "T" living together or "A" apart)
Medu	integer	mother's education (numeric: 0: none, 1: primary education (4th grade), 2: 5th to 9th grade, 3 _ secondary education or 4 _ higher education)
Fedu	integer	father's education (numeric: 0 - none, 1 - primary education (4th grade), 2 _ 5th to 9th grade, 3 _ secondary education or 4 _ higher education)
Mjob	string	mother's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")
Fjob	string	father's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")
reason	string	reason to choose this school (nominal: close to "home", school "reputation", "course" preference or "other")
guardian	string	student's guardian (nominal: "mother", "father" or "other")
traveltime	integer	home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)
studytime	integer	weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)
failures	integer	number of past class failures (numeric: n if 1<=n<3, else 4)
schoolsup	string	extra educational support (binary: yes or no)
famsup	string	family educational support (binary: yes or no)
paid	string	extra paid classes within the course subject (Math or Portuguese) (binary: yes or no)
activities	string	extra-curricular activities (binary: yes or no)
nursery	string	attended nursery school (binary: yes or no)
higher	string	wants to take higher education (binary: yes or no)
internet	string	Internet access at home (binary: yes or no)
romantic	string	with a romantic relationship (binary: yes or no)
famrel	integer	quality of family relationships (numeric: from 1 - very bad to 5 - excellent)
freetime	integer	free time after school (numeric: from 1 - very low to 5 - very high)
goout	integer	going out with friends (numeric: from 1 - very low to 5 - very high)
Dalc	integer	workday alcohol consumption (numeric: from 1 - very low to 5 - very high)
Walc	integer	weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)
health	integer	current health status (numeric: from 1 - very bad to 5 - very good)
absences	integer	number of school absences (numeric: from 0 to 93)
G1	integer	first period grade (numeric: from 0 to 20)
G2	integer	second period grade (numeric: from 0 to 20)
G3	integer	Predictor Class: final grade (numeric: from 0 to 20)

Descriptive Analysis:

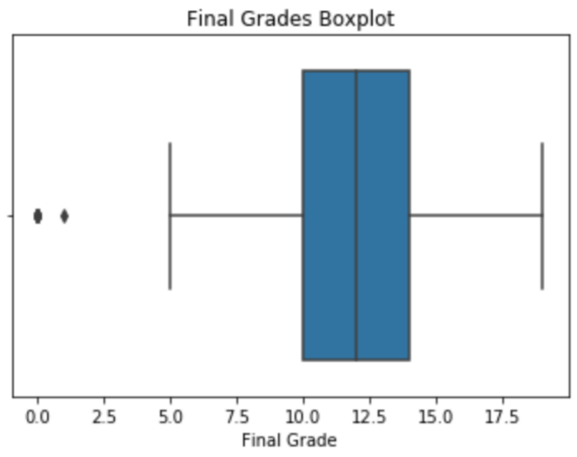
In this study we discuss the results of different analysis using different statistical measures and implementation tools including the use of visualization. 32 features will be used to analyze data and determine different patterns between them and in terms of student performances as well. The answer to our objective question would be through the descriptions of these predictors and the presentation of their variance. The use of basic statistical analysis, data exploration, and data visualization techniques provide deeper understanding of the targeted data. The following table shows the descriptive statistics of the numeric and features.



	count	mean	std	min	25%	50%	75%	max
age	649.0	16.744222	1.218138	15.0	16.0	17.0	18.0	22.0
Medu	649.0	2.514638	1.134552	0.0	2.0	2.0	4.0	4.0
Fedu	649.0	2.306626	1.099931	0.0	1.0	2.0	3.0	4.0
traveltime	649.0	1.568567	0.748660	1.0	1.0	1.0	2.0	4.0
studytime	649.0	1.930663	0.829510	1.0	1.0	2.0	2.0	4.0
failures	649.0	0.221880	0.593235	0.0	0.0	0.0	0.0	3.0
famrel	649.0	3.930663	0.955717	1.0	4.0	4.0	5.0	5.0
freetime	649.0	3.180277	1.051093	1.0	3.0	3.0	4.0	5.0
goout	649.0	3.184900	1.175766	1.0	2.0	3.0	4.0	5.0
Dalc	649.0	1.502311	0.924834	1.0	1.0	1.0	2.0	5.0
Walc	649.0	2.280431	1.284380	1.0	1.0	2.0	3.0	5.0
health	649.0	3.536210	1.446259	1.0	2.0	4.0	5.0	5.0
absences	649.0	3.659476	4.640759	0.0	0.0	2.0	6.0	32.0
G1	649.0	11.399076	2.745265	0.0	10.0	11.0	13.0	19.0
G2	649.0	11.570108	2.913639	0.0	10.0	11.0	13.0	19.0
G3	649.0	11.906009	3.230656	0.0	10.0	12.0	14.0	19.0

It is evident from the table above that the average of final grades is very similar to the average of first period grades and second period grades. Furthermore, the average age is closer to the minimum age which means our samples are mostly young and 75 percent of them are between 15 and 18 years old. Also, the average Mother Education and Father Education are very close on a scale of 5. These factors indicate homogeneity in our data and could help us approach data closely.

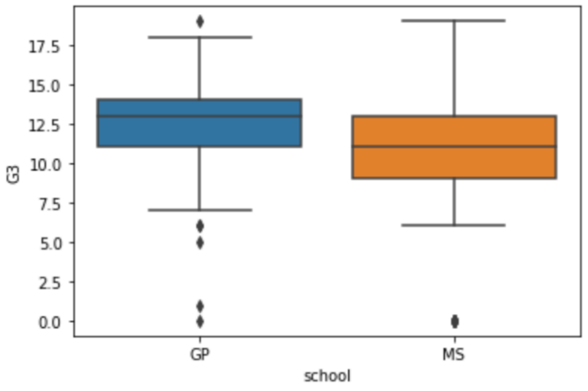
Final grades could be visualized to see their range and distributions using boxplot as shown below.



Further exploratory analysis was then implemented on categorial features to see how much they affect final scores and how data was spread to predict our target. These variables are nominal and binary.

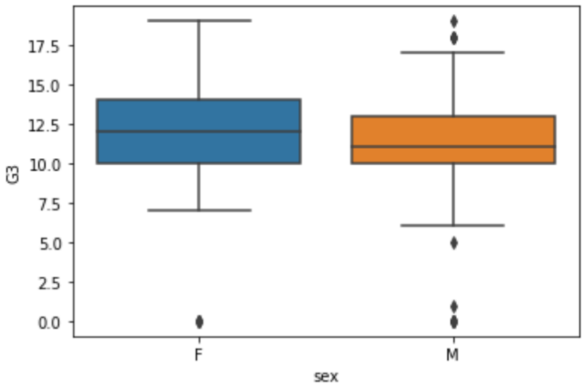
Looking at two different schools, Gabriel Pereira has more students than Mousinho da Silveira. Average student final grade in the second school which has less students is less than it is in the other.

	G3							
	count	mean	std	min	25%	50%	75%	max
school								
GP	423.0	12.576832	2.625636	0.0	11.0	13.0	14.0	19.0
MS	226.0	10.650442	3.833991	0.0	9.0	11.0	13.0	19.0



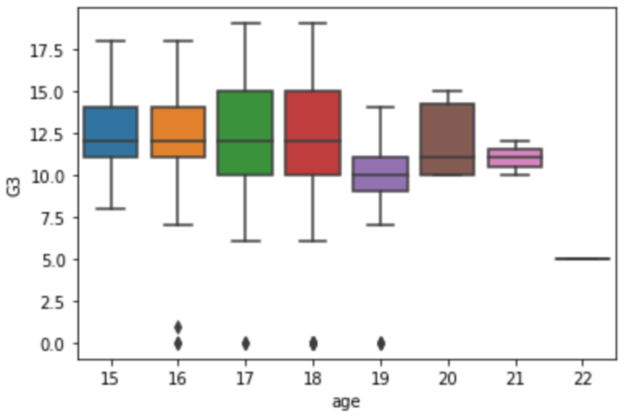
When it comes to gender, females have performed slightly better, but the mean is almost similar. Furthermore, female students were more than males. They are almost 60 % of our observations.

	G3							
	count	mean	std	min	25%	50%	75%	max
sex								
F	383.0	12.253264	3.124147	0.0	10.0	12.0	14.0	19.0
M	266.0	11.406015	3.320690	0.0	10.0	11.0	13.0	19.0



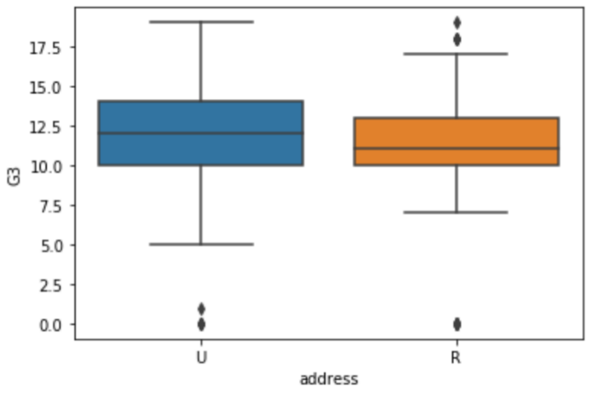
Most of the students were in age between 15 to 18 years old. As we can see in the following graph and table, most ages have similar average performances except for one age which was represented by one student who had a score of 5. Best average final grades were performed by 17s and worst was by 19s which affirms that it's a life-changing point when people grow and become adults.

	G3							
	count	mean	std	min	25%	50%	75%	max
age								
15	112.0	12.107143	2.085372	8.0	11.0	12.0	14.00	18.0
16	177.0	11.994350	2.883135	0.0	11.0	12.0	14.00	18.0
17	179.0	12.268156	3.149040	0.0	10.0	12.0	15.00	19.0
18	140.0	11.771429	4.154122	0.0	10.0	12.0	15.00	19.0
19	32.0	9.531250	3.407623	0.0	9.0	10.0	11.00	14.0
20	6.0	12.000000	2.449490	10.0	10.0	11.0	14.25	15.0
21	2.0	11.000000	1.414214	10.0	10.5	11.0	11.50	12.0
22	1.0	5.000000	NaN	5.0	5.0	5.0	5.00	5.0



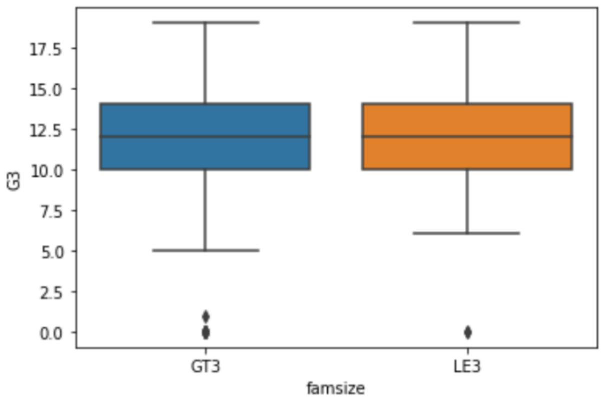
Rural students were less than half of the urban students. The average student performance for urban students is a little better than rural students.

	G3							
	count	mean	std	min	25%	50%	75%	max
address								
R	197.0	11.086294	3.605221	0.0	10.0	11.0	13.0	19.0
U	452.0	12.263274	2.987658	0.0	10.0	12.0	14.0	19.0



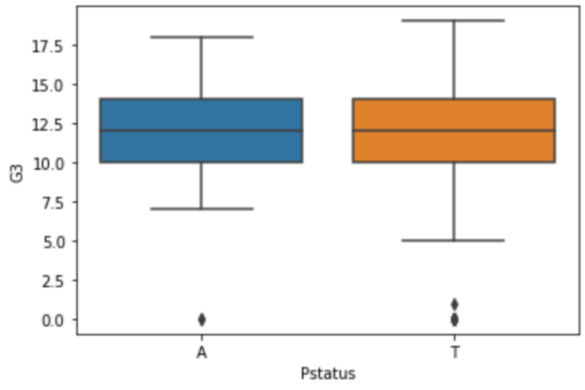
Students from families with greater than 3 people are more than with students from families with less than 3 people. They are more than twice which explains that families with higher number of people send more students to school. Student performance has not changed in relevance to family size change.

	G3							
	count	mean	std	min	25%	50%	75%	max
famsize								
GT3	457.0	11.811816	3.351426	0.0	10.0	12.0	14.0	19.0
LE3	192.0	12.130208	2.919285	0.0	10.0	12.0	14.0	19.0



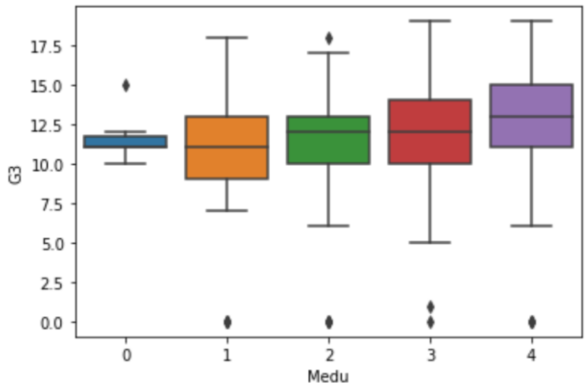
Parent’s cohabitation status has not played any role in changing the average performance for students. Students from families with parents apart still performed almost equal average to the ones from families with parents together.

	G3							
	count	mean	std	min	25%	50%	75%	max
Pstatus								
A	80.0	11.912500	3.222523	0.0	10.0	12.0	14.0	18.0
T	569.0	11.905097	3.234626	0.0	10.0	12.0	14.0	19.0



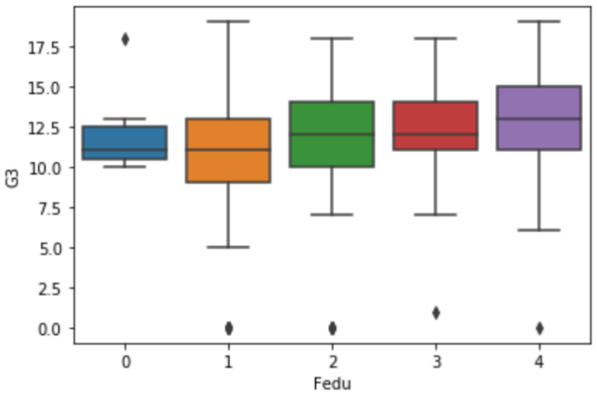
Mother education on a scale from 0 to 4 has a gradual affect. Students with highest educated mothers who were scaled 4 have the best performance among other students. It also shows that uneducated mothers are odd because they are only 6 out of 649 observations.

	G3							
	count	mean	std	min	25%	50%	75%	max
Medu								
0	6.0	11.666667	1.751190	10.0	11.0	11.0	11.75	15.0
1	143.0	10.797203	3.163523	0.0	9.0	11.0	13.00	18.0
2	186.0	11.661290	3.061232	0.0	10.0	12.0	13.00	18.0
3	139.0	11.920863	3.123229	0.0	10.0	12.0	14.00	19.0
4	175.0	13.068571	3.236978	0.0	11.0	13.0	15.00	19.0



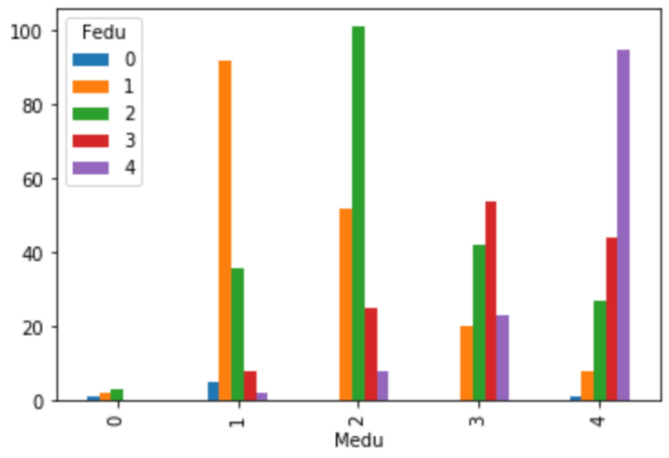
Also, father education has a gradual effect on average students’ final performance. It actually shows that students with higher educated father have performed better than other students. Furthermore, standard deviation of students with uneducated fathers is bigger than students with uneducated mothers. It’s shown in the boxplot that most of the observations with uneducated mothers are above the mean.

	G3							
	count	mean	std	min	25%	50%	75%	max
Fedu								
0	7.0	12.142857	2.794553	10.0	10.5	11.0	12.5	18.0
1	174.0	10.936782	3.424077	0.0	9.0	11.0	13.0	19.0
2	209.0	11.784689	3.448321	0.0	10.0	12.0	14.0	18.0
3	131.0	12.381679	2.491394	1.0	11.0	12.0	14.0	18.0
4	128.0	12.921875	2.915096	0.0	11.0	13.0	15.0	19.0



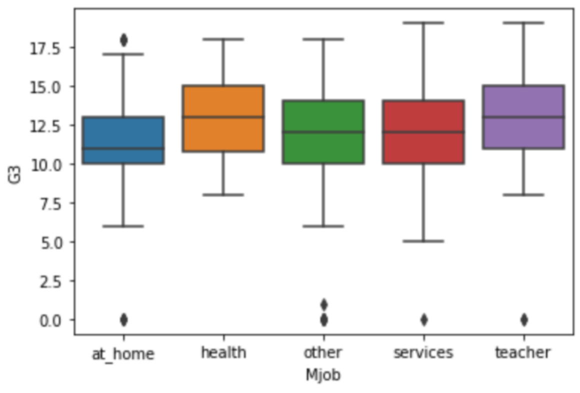
The next graph shows how much these two variables could correlate and explains the relationship between them. The bar chart shows that most of the students have mothers and fathers with a score of 2s, 4s, and 3s. It actually shows that most of the parents are educationally close and higher education differences rarely exists.

Fedu	**0**	**1**	**2**	**3**	**4**
Medu					
0	1	2	3	0	0
1	5	92	36	8	2
2	0	52	101	25	8
3	0	20	42	54	23
4	1	8	27	44	95



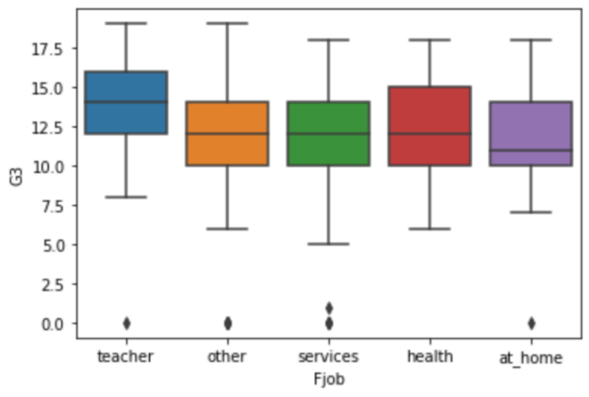
Mother Job did not have a big of an impact on average students' final grades. Students with teahcer mothers have performed the best. Students with mothers at home have performed the least.

	G3							
	count	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**
Mjob								
at_home	135.0	11.044444	3.138273	0.0	10.00	11.0	13.0	18.0
health	48.0	13.062500	2.956466	8.0	10.75	13.0	15.0	18.0
other	258.0	11.670543	3.307224	0.0	10.00	12.0	14.0	18.0
services	136.0	12.147059	2.917456	0.0	10.00	12.0	14.0	19.0
teacher	72.0	13.138889	3.307293	0.0	11.00	13.0	15.0	19.0



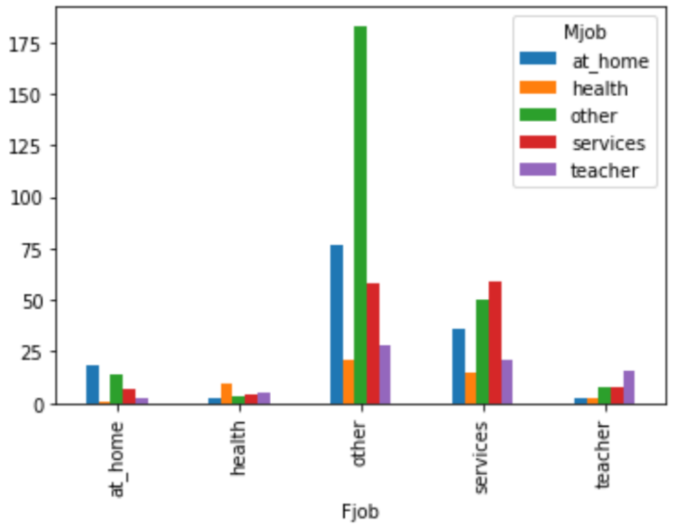
Father Job has not made big differences in terms of final students' performance. However, students with teacher fathers have performed the best in average even though students with teacher mothers are two times the students with teacher fathers.

	G3							
	count	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**
Fjob								
at_home	42.0	11.428571	3.201698	0.0	10.0	11.0	14.0	18.0
health	23.0	12.565217	3.130874	6.0	10.0	12.0	15.0	18.0
other	367.0	11.891008	3.074503	0.0	10.0	12.0	14.0	19.0
services	181.0	11.629834	3.438507	0.0	10.0	12.0	14.0	18.0
teacher	36.0	13.583333	3.400630	0.0	12.0	14.0	16.0	19.0



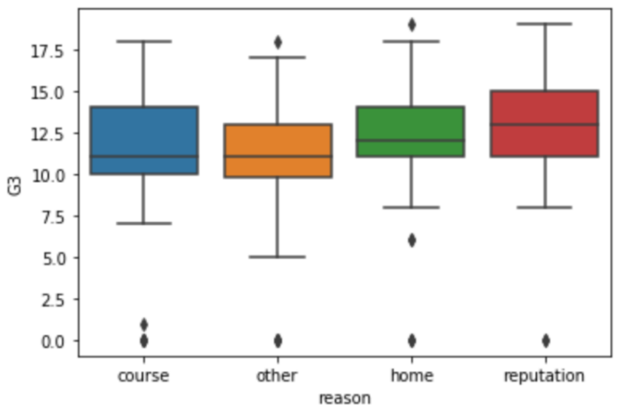
It’s shown in the bar graph below that most parents have similar jobs. Moreover, most parents have another job than what’s specified for the nominal values. Students with parents in the health sector are the least.

Mjob Fjob	at_home	health	other	services	teacher
at_home	18	1	14	7	2
health	2	9	3	4	5
other	77	21	183	58	28
services	36	15	50	59	21
teacher	2	2	8	8	16



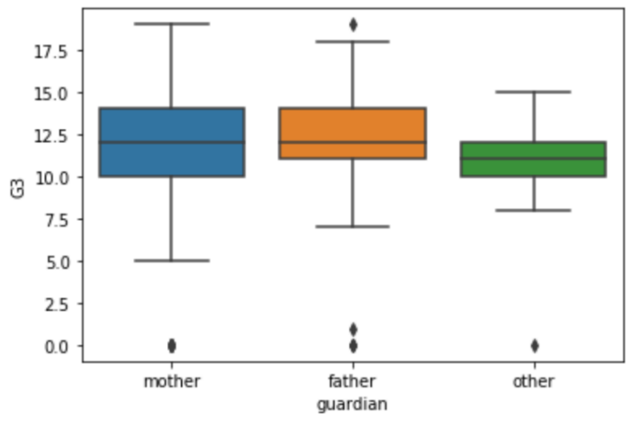
Students who joined these school because of course preferences are the most. Number of students who joined because school is closed to their houses is similar to the ones who joined because of the school reputation. Different reasons have not changed the average students’ performance a lot although students who joined because of school reputation have performed the best.

	G3							
	count	mean	std	min	25%	50%	75%	max
reason								
course	285.0	11.547368	3.108717	0.0	10.00	11.0	14.0	18.0
home	149.0	12.181208	2.952447	0.0	11.00	12.0	14.0	19.0
other	72.0	10.694444	3.931236	0.0	9.75	11.0	13.0	18.0
reputation	143.0	12.944056	3.052997	0.0	11.00	13.0	15.0	19.0

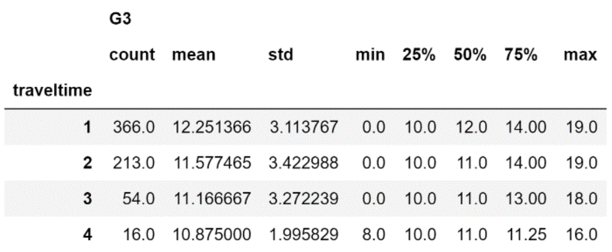


Students with guardian mothers are three times students with guardian fathers. Student guardian has not changed average students’ performance at their finals yet student with father guardians still performed slightly better than other students.

	G3							
	count	mean	std	min	25%	50%	75%	max
guardian								
father	153.0	12.202614	3.192012	0.0	11.0	12.0	14.0	19.0
mother	455.0	11.896703	3.279943	0.0	10.0	12.0	14.0	19.0
other	41.0	10.902439	2.624927	0.0	10.0	11.0	12.0	15.0



The table below shows the travel time from home to school affected students' average performance gradually. Students who live close to their schools have performed slightly better than who live further. Students who live 15 minutes or less are the most and have performed the best.



	G3							
	count	mean	std	min	25%	50%	75%	max
traveltime								
1	366.0	12.251366	3.113767	0.0	10.0	12.0	14.00	19.0
2	213.0	11.577465	3.422988	0.0	10.0	11.0	14.00	19.0
3	54.0	11.166667	3.272239	0.0	10.0	11.0	13.00	18.0
4	16.0	10.875000	1.995829	8.0	10.0	11.0	11.25	16.0

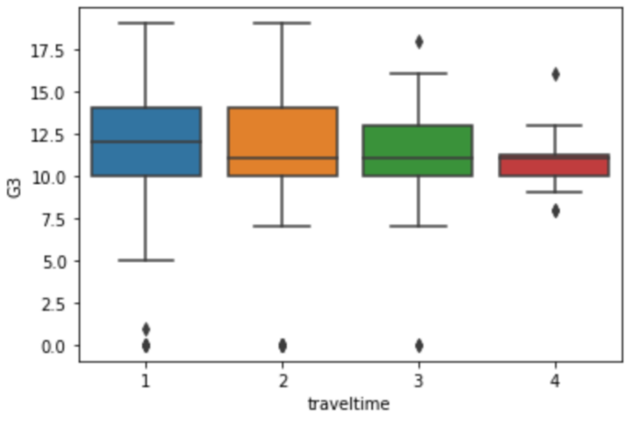
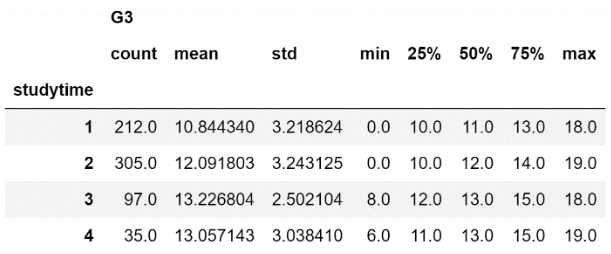
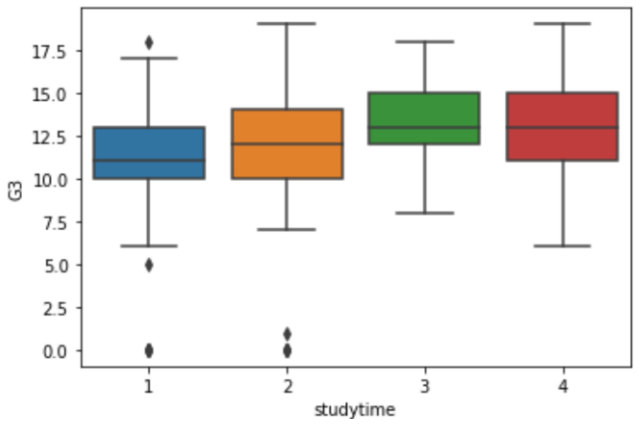


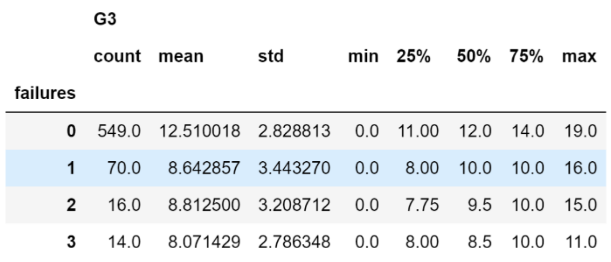
Table below explains that most of the students study less than 5 hours. Students who study more than 5 hours still performed the best and students who study less than 2 hours performed the least.



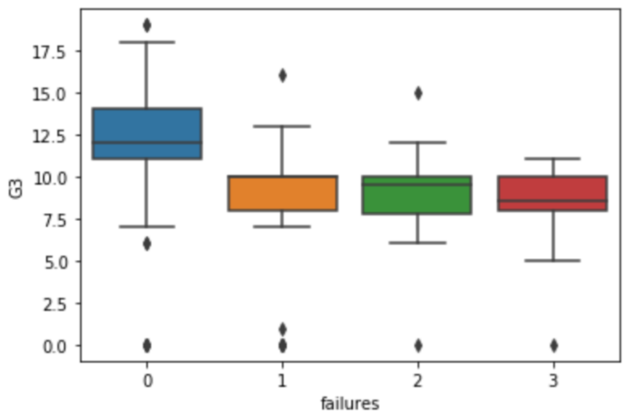
	G3							
	count	mean	std	min	25%	50%	75%	max
studytime								
1	212.0	10.844340	3.218624	0.0	10.0	11.0	13.0	18.0
2	305.0	12.091803	3.243125	0.0	10.0	12.0	14.0	19.0
3	97.0	13.226804	2.502104	8.0	12.0	13.0	15.0	18.0
4	35.0	13.057143	3.038410	6.0	11.0	13.0	15.0	19.0



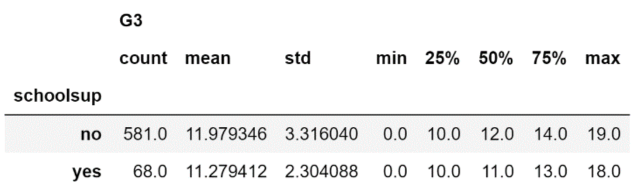
The different between students who have previous failures and students who do not is clearly shown in the boxplot below. Students who have more than one failure still performed similarly regardless the number of failures. Whilst students with no former failures has performed quite much better and they are around 4 grades apart.



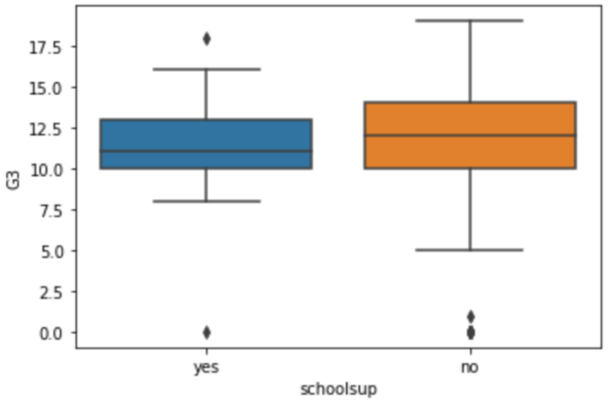
	G3							
	count	mean	std	min	25%	50%	75%	max
failures								
0	549.0	12.510018	2.828813	0.0	11.00	12.0	14.0	19.0
1	70.0	8.642857	3.443270	0.0	8.00	10.0	10.0	16.0
2	16.0	8.812500	3.208712	0.0	7.75	9.5	10.0	15.0
3	14.0	8.071429	2.786348	0.0	8.00	8.5	10.0	11.0



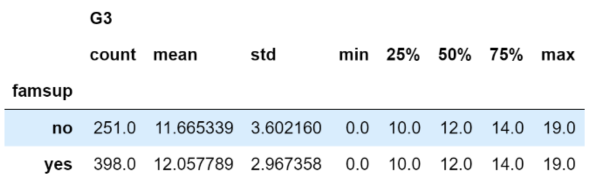
Most of the students do not have extra educational support. Extra support does not change the average student performance much. Thus, extra support seems to have been provided to students with lower performances since they still perform lower or almost similar to other students.



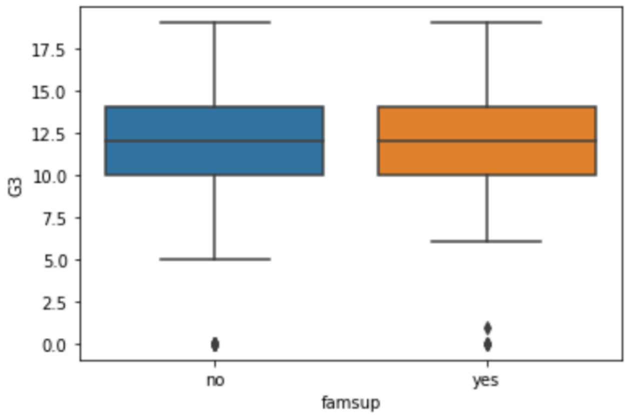
	G3							
	count	mean	std	min	25%	50%	75%	max
schoolsup								
no	581.0	11.979346	3.316040	0.0	10.0	12.0	14.0	19.0
yes	68.0	11.279412	2.304088	0.0	10.0	11.0	13.0	18.0



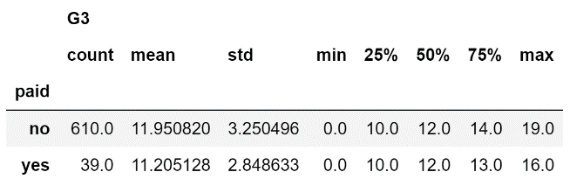
Students with family educational support are more than others with a small higher performance than students who do not have educational family support.



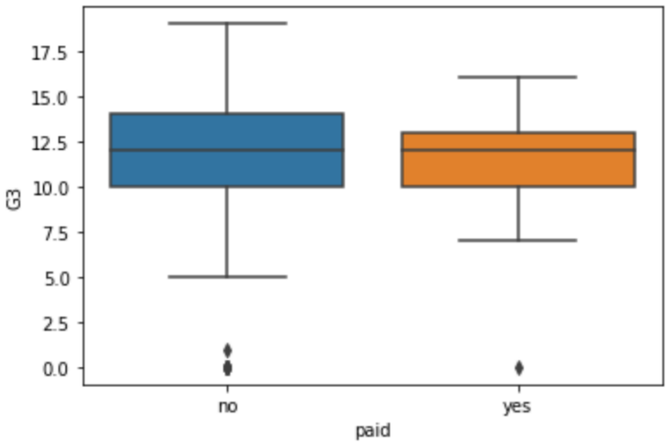
	G3							
	count	mean	std	min	25%	50%	75%	max
famsup								
no	251.0	11.665339	3.602160	0.0	10.0	12.0	14.0	19.0
yes	398.0	12.057789	2.967358	0.0	10.0	12.0	14.0	19.0



Number of students who have extra paid classes (Math or Portuguese) are very low. They performed slightly less than other students in average but the different is too small to consider.

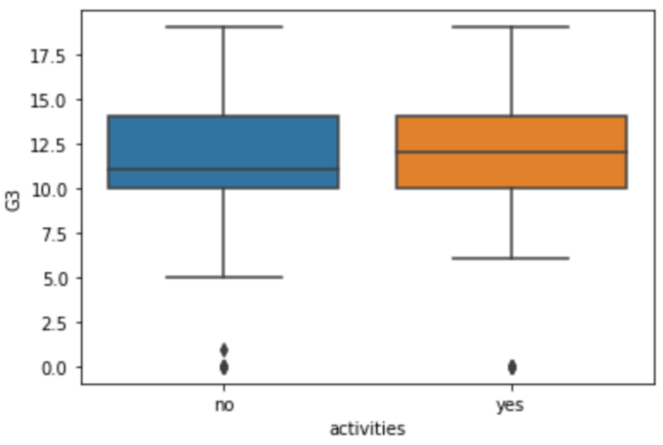


	G3							
	count	mean	std	min	25%	50%	75%	max
paid								
no	610.0	11.950820	3.250496	0.0	10.0	12.0	14.0	19.0
yes	39.0	11.205128	2.848633	0.0	10.0	12.0	13.0	16.0



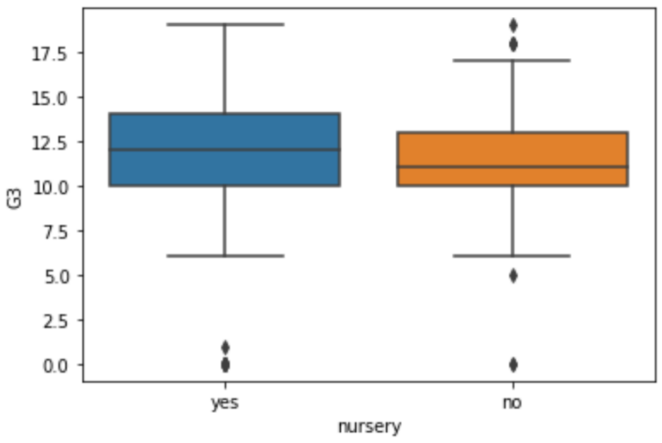
Extra-curricular activities have not changed average students' performance much even though students who perform extra activities still perform a little better. The table below shows that number of students with extra-curricular activities is close to number of students who does not have any curricular activities.

	G3							
	count	mean	std	min	25%	50%	75%	max
activities								
no	334.0	11.718563	3.235290	0.0	10.0	11.0	14.0	19.0
yes	315.0	12.104762	3.218944	0.0	10.0	12.0	14.0	19.0



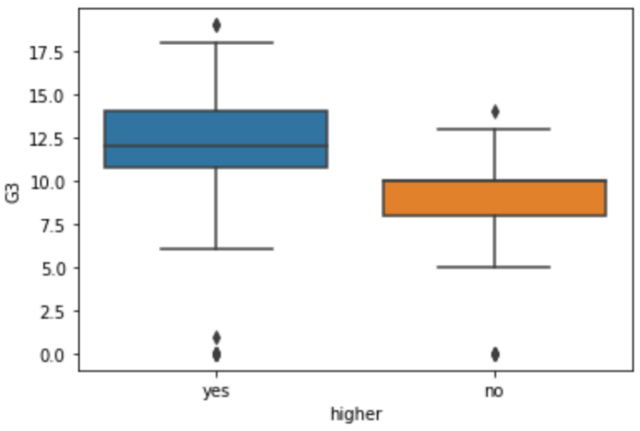
Most of the students attended the nursery school. Student performances for students who attended nursery is spread symmetrically around their mean.

	G3							
	count	mean	std	min	25%	50%	75%	max
nursery								
no	128.0	11.718750	3.006391	0.0	10.0	11.0	13.0	19.0
yes	521.0	11.952015	3.284521	0.0	10.0	12.0	14.0	19.0



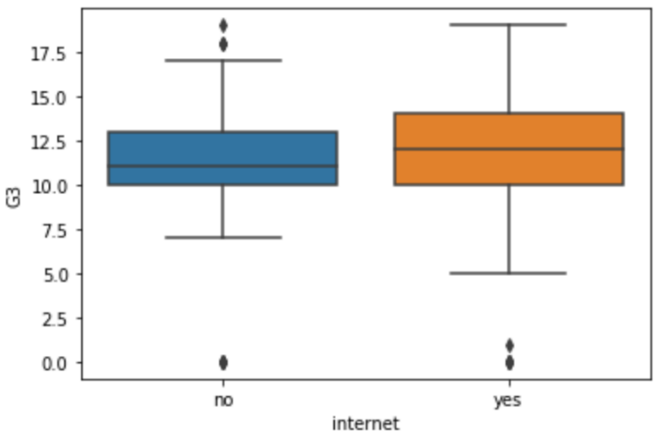
Students who want to take higher education are more than 8 times number of students who don’t. It’s clearly shown in the next graph that students who want to pursue higher education perform much better than students who do not want to. This explains student determination in improving themselves in order to be able to pursue further education.

	G3							
	count	mean	std	min	25%	50%	75%	max
higher								
no	69.0	8.797101	2.973311	0.0	8.00	10.0	10.0	14.0
yes	580.0	12.275862	3.058402	0.0	10.75	12.0	14.0	19.0



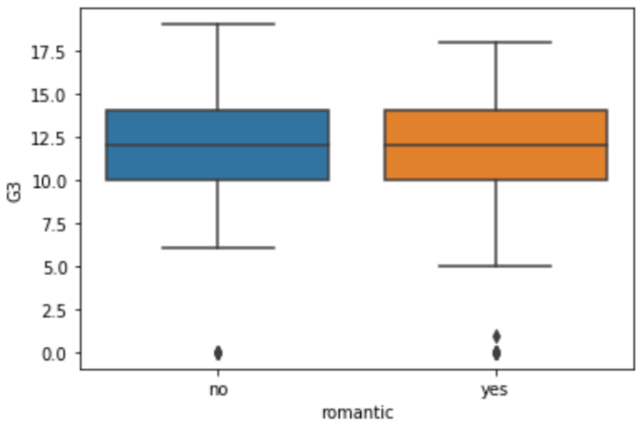
Students with internet access are three times more than students who have no internet access. Also, students who have internet access have a better average performance than others.

	G3							
	count	mean	std	min	25%	50%	75%	max
internet								
no	151.0	11.026490	3.446635	0.0	10.0	11.0	13.0	19.0
yes	498.0	12.172691	3.117146	0.0	10.0	12.0	14.0	19.0



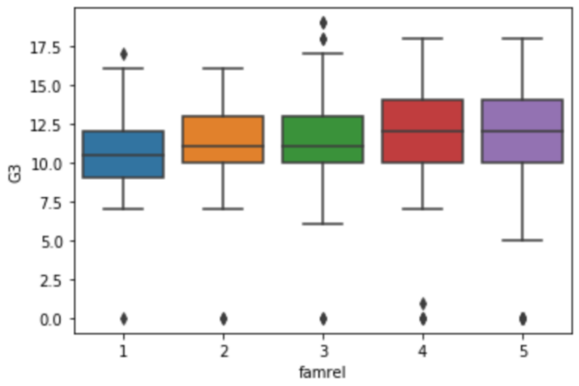
Students with no romantic relationship still performed unnoticeably better in their finals. The table below shows that students with no romantic relationships are almost as twice as student with a romantic relationship.

	G3							
	count	mean	std	min	25%	50%	75%	max
romantic								
no	410.0	12.129268	3.003726	0.0	10.0	12.0	14.0	19.0
yes	239.0	11.523013	3.560771	0.0	10.0	12.0	14.0	18.0



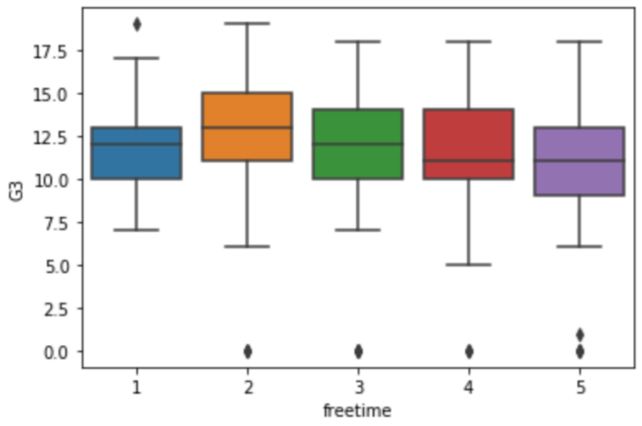
Students with family relations score 4 out of 5 have performed the best in average. They represent almost 50% of the students. It's evident from the table below that family relationship affect student final performance gradually yet it decreases the performance at its highest.

	G3							
	count	mean	std	min	25%	50%	75%	max
famrel								
1	22.0	10.636364	3.645551	0.0	9.0	10.5	12.0	17.0
2	29.0	10.862069	3.710257	0.0	10.0	11.0	13.0	16.0
3	101.0	11.594059	3.033738	0.0	10.0	11.0	13.0	19.0
4	317.0	12.343849	2.937923	0.0	10.0	12.0	14.0	18.0
5	180.0	11.633333	3.584417	0.0	10.0	12.0	14.0	18.0



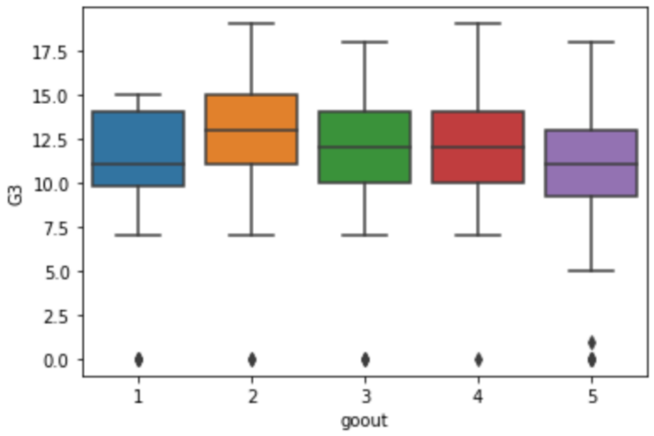
Students perform the worst when they have the freest time after school. In addition, they perform the best when their free time score is 2 out of 5. Almost 40% of students have a free time score 3 out of 5. This explains that students who study more have less free time than others and therefore, their average performance is higher than others.

	G3							
	count	mean	std	min	25%	50%	75%	max
freetime								
1	45.0	11.733333	2.499091	7.0	10.0	12.0	13.0	19.0
2	107.0	12.710280	3.458676	0.0	11.0	13.0	15.0	19.0
3	251.0	12.059761	3.041121	0.0	10.0	12.0	14.0	18.0
4	178.0	11.713483	3.064640	0.0	10.0	11.0	14.0	18.0
5	68.0	10.691176	3.982264	0.0	9.0	11.0	13.0	18.0



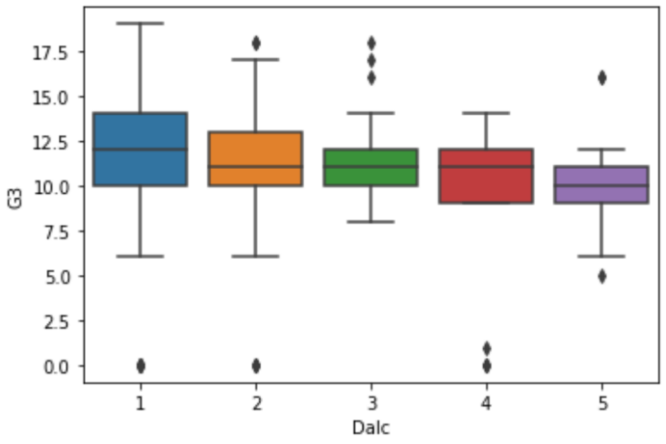
In the next table, students who do not go out with friends have the least performances. Furthermore, students who go out a lot have similar low performances as well. At the same time, students who go out in between are more and perform better.

	G3							
	count	mean	std	min	25%	50%	75%	max
goout								
1	48.0	10.729167	3.846743	0.0	9.75	11.0	14.0	15.0
2	145.0	12.668966	3.171170	0.0	11.00	13.0	15.0	19.0
3	205.0	12.151220	2.902258	0.0	10.00	12.0	14.0	18.0
4	141.0	11.971631	2.818164	0.0	10.00	12.0	14.0	19.0
5	110.0	10.872727	3.719794	0.0	9.25	11.0	13.0	18.0

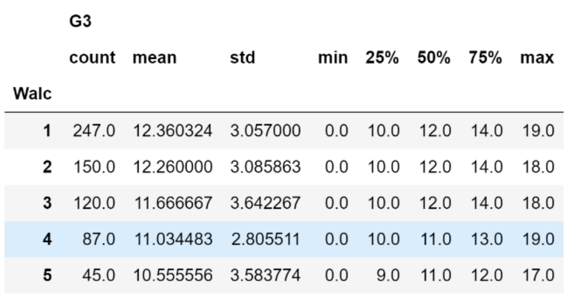


Students with low workday alcohol consumption are dominant. The table below shows that student with the lowest consumption have the best final grades among others. The lowest performances are students with score of 4 out 5 for workday alcohol consumption.

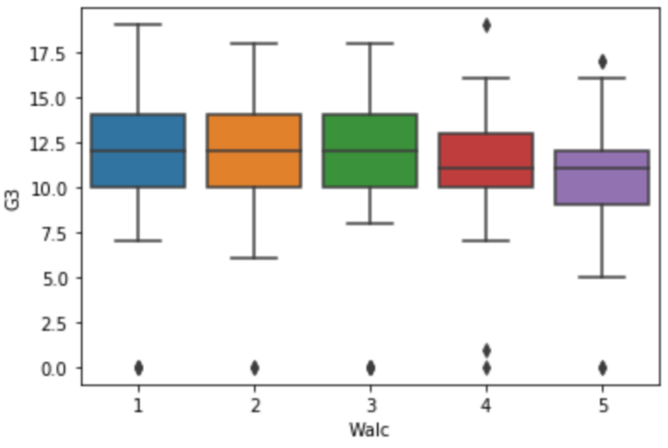
	G3							
	count	mean	std	min	25%	50%	75%	max
Dalc								
1	451.0	12.299335	3.102898	0.0	10.0	12.0	14.0	19.0
2	121.0	11.363636	3.329164	0.0	10.0	11.0	13.0	18.0
3	43.0	11.139535	2.252844	8.0	10.0	11.0	12.0	18.0
4	17.0	8.941176	5.129213	0.0	9.0	11.0	12.0	14.0
5	17.0	10.235294	2.948080	5.0	9.0	10.0	11.0	16.0



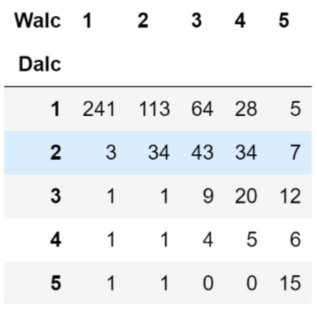
Students with higher weekend alcohol consumption performed the lowest. The performance then increases gradually according to the weekend alcohol consumption until it reaches highest average performance for students who consume alcohol the least during weekends.



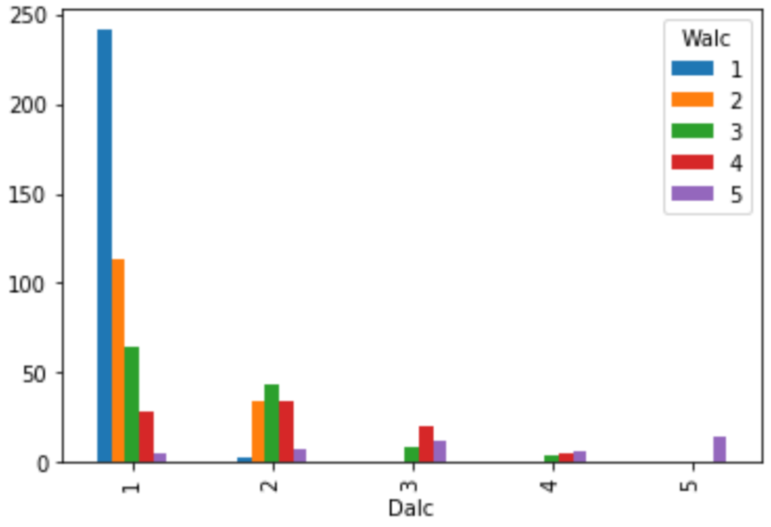
	G3							
	count	mean	std	min	25%	50%	75%	max
Walc								
1	247.0	12.360324	3.057000	0.0	10.0	12.0	14.0	19.0
2	150.0	12.260000	3.085863	0.0	10.0	12.0	14.0	18.0
3	120.0	11.666667	3.642267	0.0	10.0	12.0	14.0	18.0
4	87.0	11.034483	2.805511	0.0	10.0	11.0	13.0	19.0
5	45.0	10.555556	3.583774	0.0	9.0	11.0	12.0	17.0



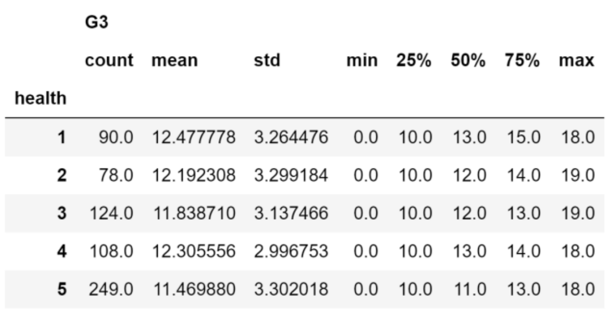
The next table and graph show how students with different alcohol consumptions correlate during different days of the week (workdays and weekends). The bar graph shows that most of the students have a low score of alcohol consumption during weekdays and weekends. Most of other students consume alcohol during weekends differently. Similar alcohol consumption behaviors during both periods rarely exist. Thus, correlations are higher at lower weekday's consumptions with higher consumptions during weekends.



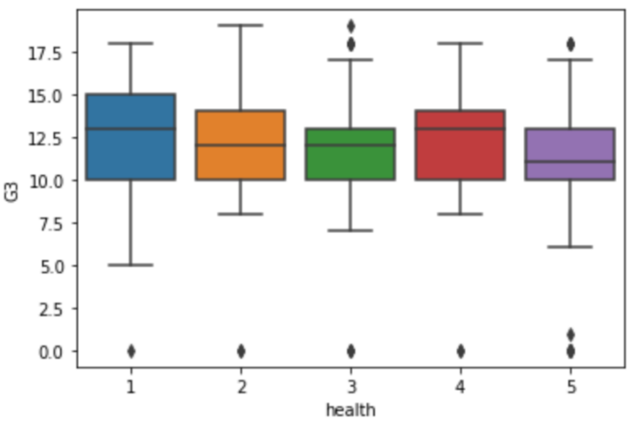
Walc	1	2	3	4	5
Dalc					
1	241	113	64	28	5
2	3	34	43	34	7
3	1	1	9	20	12
4	1	1	4	5	6
5	1	1	0	0	15



Health has an interesting relationship with final student performances. Students with very bad health have performed the best in average while students with very good health perform the lowest. The differences in performance are very small and could be neglectable. Yet, it shows that more than 60 % of students have a medium to a very good health.

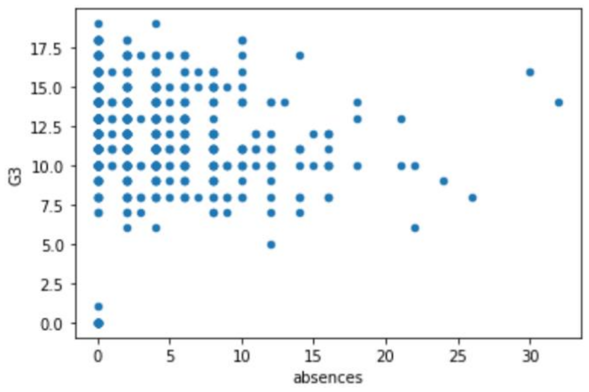


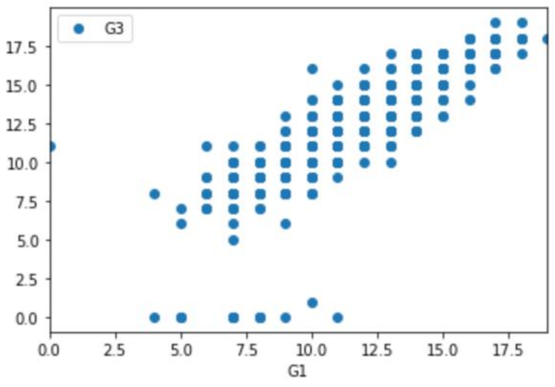
	G3							
	count	mean	std	min	25%	50%	75%	max
health								
1	90.0	12.477778	3.264476	0.0	10.0	13.0	15.0	18.0
2	78.0	12.192308	3.299184	0.0	10.0	12.0	14.0	19.0
3	124.0	11.838710	3.137466	0.0	10.0	12.0	13.0	19.0
4	108.0	12.305556	2.996753	0.0	10.0	13.0	14.0	18.0
5	249.0	11.469880	3.302018	0.0	10.0	11.0	13.0	18.0

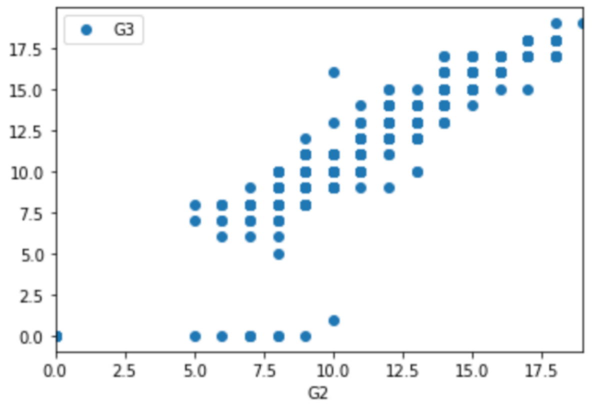


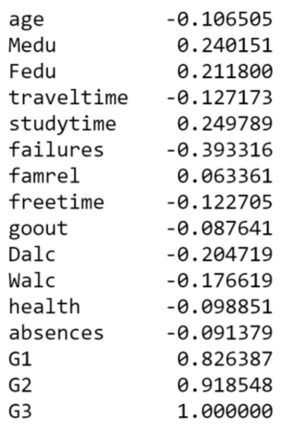
In the graphs, scatterplots were used to check if there is any linear relationship between numeric features and final students' performances. These continues variables are going to be plotter against our target variable to check for significant correlations between them. According to graphs below, absences have a weak relationship with student's performance. On contrary, first and second period grades show a strong linear relationship with students' final performance and results.

In addition, correlation matrix using *corr* function helps us look at the correlation coefficients for final grades with other predictors. It's clearly shown that G1 and G2 are highly correlated with our dependent variable G3 and their coefficients are 0.82 and 0.91 respectively.



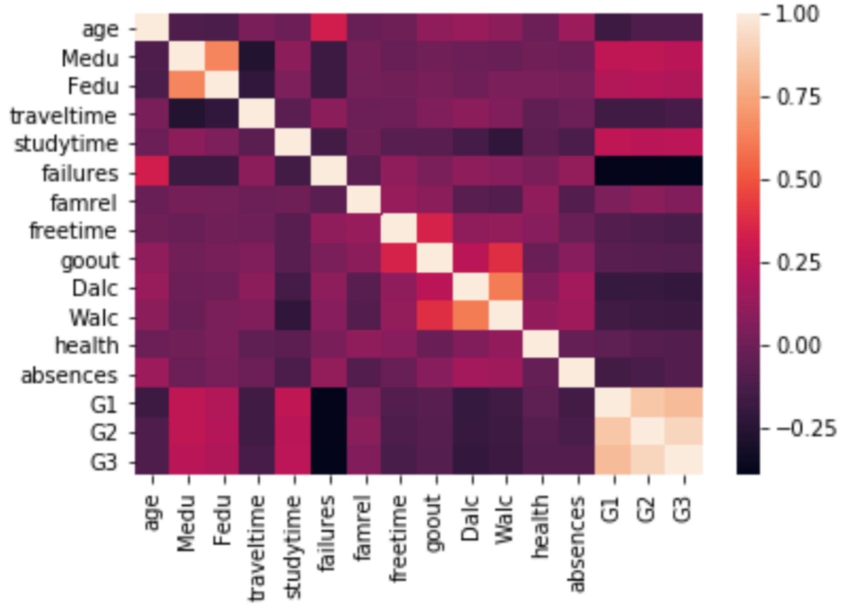






```
age           -0.106505
Medu           0.240151
Fedu           0.211800
traveltime    -0.127173
studytime      0.249789
failures      -0.393316
famrel         0.063361
freetime      -0.122705
goout         -0.087641
Dalc          -0.204719
Walc          -0.176619
health        -0.098851
absences      -0.091379
G1             0.826387
G2             0.918548
G3             1.000000
```

First, correlation matrix was used to identify correlations between different variables. In the correlation matrix, the sizes of some correlation place constraints on the sizes of others. Second, it's important to look at frameworks of individual correlations where match acute relationships are balanced due to correlation between two variables. The next graph is a visualized representation of the correlation matrix.



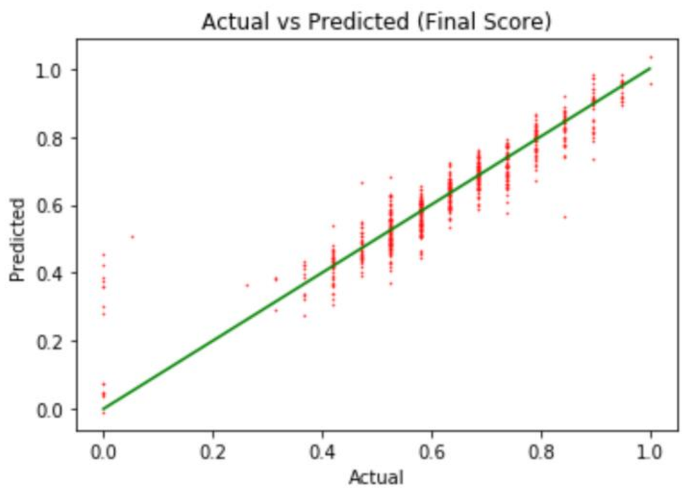
Multicollinearity Exaggerates the standard error of the betas and reverse beta slops which causes instabilities in beta estimates. Therefore, multiple regression and multivariate multiple regression for several dependent variables are good techniques that analyze correlation and create predictive models that anticipate students' final results.

Predictive Analytics:

First prediction technique used is Linear Regression. To perform regression on data, all categorical variables must be converted to dummy variables using pandas. Our data frame shape has then been changed from 33 variables to 59 variables. Data is not normalized and it's important to use normalization techniques so data can have one scale. Minimum-Maximum Scaler preprocessing function from scikit-learn was used to perform the standardization process.

Target variable were then removed to be subset in a different data frame. It was dropped from the training frame and separated for repression. *sklearn.linear_model* was used to import different functions using different techniques from. Our data then was fitted in a linear regression model to predict average final students' performances. Error was then computed by checking the differences between predicted and actual values. Root Mean Square Error (RMSE) was then computed. RMSE is the standard deviation of the residuals which measures how spread out these residuals are.

RMSE on the training data was very low (RMSE = 0.0658) which is a good sign except for overfitting. Regression coefficients were then displayed to show different loadings among different training variables. Actual values of the target attribute were then plotted against Predicted values to show the correlation between them.



Cross validation is a very important technique that help improve prediction models during linear regression process. It’s using multiple mini training and testing splits and use them to tune the model. Data is partitioned into k subsets called folds and train the model using one of the folds each time and use the remaining folds as the test set. This technique could reduce the unwanted noise and randomness in our model and avoid overfitting.

10-fold cross validation were performed to compare the cross validation RMSE to the training RMSE using *KFoldmodule* from *sklearn.model_selection.* Average RMSE for all training folds was computed and compared to the training RMSE. The change was too small and RMSE on training was 0.0659 in comparison to 0.0664 for the cross validation RMSE.

Feature Selection:

Using scikit-learn regression model from *sklearn.linear_model*, a function was created to takes as input the training data, target variable, number of cross validations, the model method, and percentile step. The script returns the optimal percentage of the most informative features to use. *feature_selection.SelectPercentile* was used to find the most informative variables. The mean scores was determined using *model_selection.cross_val_score* as it was cited in the appendix.

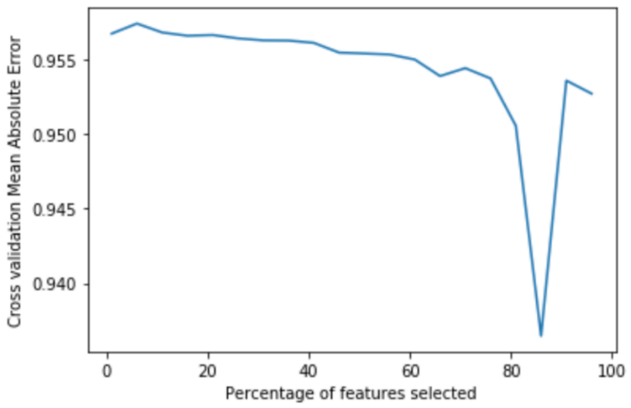
School data set and target variable frame were processed in this function using 5 cross validations and 5 percentile steps. The results in the graph below show that 4 variables were chosen as the most informative values because they have high scores (failures, G1, G2, higher_no). Thus, the optimal number of features is 4 and the optimal percentile of features is 6. G1 and G2 have the highest weights which explains the strong linear relationship between these variables that was detected above using correlation coefficients and correlation matrix.

```
The most informative variables and their weights are:

failures         7.423385071195365
G1        39.59772918378967
G2        30.386972706316584
higher_no        10.635921741608422


Optimal percentile of features:[6]

Optimal number of features:4
```



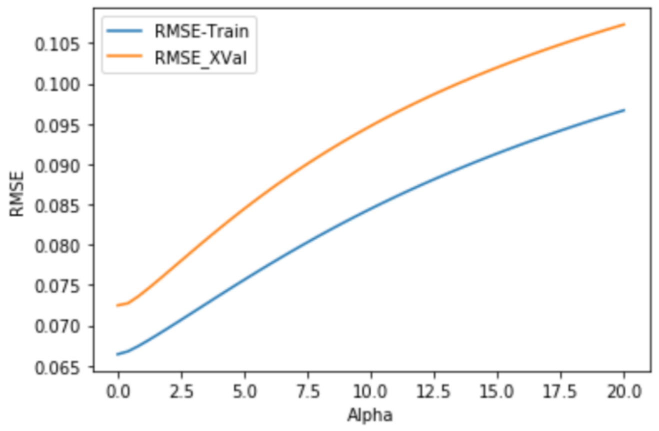
Ridge and Lasso Regressions:

Ridge regression was then performed to avoid the problem of overfitting and the failure to find unique solutions. Ridge help improve the model and add a penalty to avoid extracting noise. Lasso Regression using the modules from *sklearn.linear_model* was implemented in addition to performing systematic model selection to identify the optimal alpha parameter.

A 20%-80% randomized data split was used to set aside the test portion and use it later for testing purposes to evaluate the model. Only the 80% randomized chosen data will be used for training the model. A function will then be created to take the training data, target variable, number of cross validations, and model method to be trained (Lasso or Regression). The script runs to find the optimal alpha that has the least average of RMSE on n numbers of cross validations and RMSE on the training set. Furthermore, error values (RMSE) on the training and cross-validation splits were plotted across the specified values of the alpha parameters.

Using the training data and target variable with 5 cross validations, chosen optimal alpha was 0.01 which has an average RMSE of 0.06944. The graph below show the relationship between both RMSEs and alpha parameters specified in the range of (.01,20,50) and as it’s clarified in scripts in the appendix.

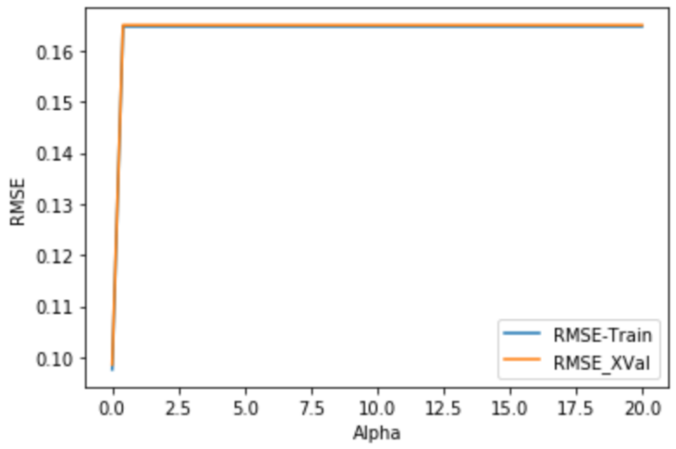
```
Optimal alpha = 0.010 with RMSE avarage = 0.06944
```



Since this function returns the optimal alpha parameter, this alpha was then used to perform ridge regression on the testing set using the same number of cross validations 5. After fitting the testing data using the optimal parameter, the RMSE was computed. RMSE for the testing set was 0.0536 which is considered good. The RMSE on testing is less than it is on training which is 0.069. This model was able to predict on the testing set almost with the same RMSE and anticipation capabilities.

The same function will then be used to implement lasso regression using the same number of cross validations. The function will return the optimal alpha for the model and its corresponding average RMSE. As shown in the graph below, the optimal alpha was 0.01 and the average RMSE was 0.09812. Average RMSE on training set using ridge regression is still smaller than it is on the training set using the same model.

```
Optimal alpha = 0.010 with RMSE avarage = 0.09812
```



Moreover, optimal alpha parameter for lasso regression model was then used on the testing set to see the changes in the RMSE value. The RMSE on the testing set using the optimal alpha parameter was 0.1007 which is slightly higher yet so similar to the RMSE on the training set using the same model.

Conclusion

Education is a very important element in our society. Knowledge Discovery Process can play a crucial role which allows many Data Mining techniques to extract knowledge from row data and obtain wisdom. This process provides interesting patterns in the education sector and enhance the quality of education standards. In this paper, the final students' performance has been addressed for predictions using past school results, demographic, social, and other school related features. The different exploratory analysis methods, statistical measures, and techniques were used to provide detailed descriptive analysis about students and their performances.

The obtained results of the prediction models using different regressors reveal that it's possible to achieve high predictive accuracy. This explains that students' performances are highly affected by previous achievements. This also shows that some features could account for more variances than others. These predictive models were able to provide the optimal number of features which could explain a higher percentage of variance in the target data set. Using different parameters helped us build more accurate models using optimal values that detect more signals in the training set.

These models could help build a student prediction engine as a support for a school system. This engine could help improve students' performances and generate school reports. These reports can be effective feedbacks for students and school professionals. This dataset could still be enriched with more student data which may reveal more interesting patterns and student behaviors that help schools and educational organizations take more educated decisions and elevate results.

Appendix

Python code:

```
## Import Modules
import numpy as np
import pandas as pd
from os import chdir
import os
import statistics
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression, Lasso, Ridge, ElasticNet,
SGDRegressor
%matplotlib inline
In [1]:
## Change working directory
os.chdir('C:\\Users\\rimo\\Desktop')
os.getcwd()
In [2]:
## Load text files via read_csv

school = pd.read_csv('school_grades_dataset.csv', sep = ',')
In [3]:
#649 observations X 33 features
school.info()
In [4]:
school.head()
In [5]:
school.describe().T
In [6]:
# Seperate the target attribute
final_grade = school['G3']
final_grade.head()
In [7]:
#Exploratory analysis for target data
import seaborn as sns
sns.boxplot(final_grade)
plot.title('Final Grades Boxplot')
plot.xlabel('Final Grade')
In [8]:
final_grade.describe()
```

In [9]:
```
#Exploratory Analysis for Categorial data in terms of Final School Grade
#"student's school (binary: ""GP"" Gabriel Pereira or ""MS"" Mousinho da Silv
eira)"
school[['school','G3']].groupby("school").describe()
```
In [10]:
```
g3 = school[['school','G3']]
sns.boxplot(x='school', y='G3', data=g3)
```
In [11]:
```
#"student's sex (binary: ""F"" female or ""M"" male)"
school[['sex','G3']].groupby("sex").describe()
```
In [12]:
```
sex = school[['sex','G3']]
sns.boxplot(x='sex', y='G3', data=sex)
```
In [13]:
```
#student's age (numeric: from 15 to 22)
school[['age','G3']].groupby("age").describe()
```
In [14]:
```
age = school[['age','G3']]
sns.boxplot(x='age', y='G3', data=age)
```
In [15]:
```
#"student's home address type (binary: ""U"" urban or ""R"" rural)"
school[['address','G3']].groupby("address").describe()
```
In [16]:
```
address = school[['address','G3']]
sns.boxplot(x='address', y='G3', data=address)
```
In [17]:
```
#"family size (binary: ""LE3"" less or equal to 3 or ""GT3"" greater than 3)"
school[['famsize','G3']].groupby("famsize").describe()
```
In [18]:
```
famsize = school[['famsize','G3']]
sns.boxplot(x='famsize', y='G3', data=famsize)
```
In [19]:
```
#"parent's cohabitation status (binary: ""T"" living together or ""A"" apart)
"
school[['Pstatus','G3']].groupby("Pstatus").describe()
```
In [20]:
```
Pstatus = school[['Pstatus','G3']]
sns.boxplot(x='Pstatus', y='G3', data=Pstatus)
```
In [21]:

```
#"mother's education (numeric: 0: none,   1: primary education (4th grade),
2: 5th to 9th grade,
#3 _ secondary education or 4 _ higher education)"
school[['Medu','G3']].groupby("Medu").describe()
```

In [22]:

```
Medu = school[['Medu','G3']]
sns.boxplot(x='Medu', y='G3', data=Medu)
```

In [23]:

```
#"father's education (numeric: 0 - none,   1 - primary education (4th grade),
2 _ 5th to
#9th grade,  3 _ secondary education or 4 _ higher education)"
school[['Fedu','G3']].groupby("Fedu").describe()
```

In [24]:

```
Fedu = school[['Fedu','G3']]
sns.boxplot(x='Fedu', y='G3', data=Fedu)
```

In [25]:

```
cross = pd.crosstab(school["Medu"], school["Fedu"])
cross
```

In [26]:

```
plt.show(cross.plot(kind="bar"))
```

In [27]:

```
#"mother's job (nominal: ""teacher"",  ""health"" care related,  civil ""serv
ices""
#(e.g. administrative or police),  ""at_home"" or ""other"")"
school[['Mjob','G3']].groupby("Mjob").describe()
```

In [28]:

```
Mjob = school[['Mjob','G3']]
sns.boxplot(x='Mjob', y='G3', data=Mjob)
```

In [29]:

```
#"father's job (nominal: ""teacher"",  ""health"" care related,  civil ""serv
ices""
#(e.g. administrative or police),  ""at_home"" or ""other"")"
school[['Fjob','G3']].groupby("Fjob").describe()
```

In [30]:

```
Fjob = school[['Fjob','G3']]
sns.boxplot(x='Fjob', y='G3', data=Fjob)
```

In [31]:

```
cross_job = pd.crosstab(school["Fjob"], school["Mjob"])
cross_job
```

In [32]:

```
plt.show(cross_job.plot(kind="bar"))
```

In [33]:
```
#"reason to choose this school (nominal: close to ""home"", school ""reputati
on"",
#""course"" preference or ""other"")"
school[['reason','G3']].groupby("reason").describe()
```

In [34]:
```
reason = school[['reason','G3']]
sns.boxplot(x='reason', y='G3', data=reason)
```

In [35]:
```
#"student's guardian (nominal: ""mother"",  ""father"" or ""other"")"
school[['guardian','G3']].groupby("guardian").describe()
```

In [36]:
```
guardian = school[['guardian','G3']]
sns.boxplot(x='guardian', y='G3', data=guardian)
```

In [37]:
```
#"home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 3
0 min. to 1 hour,
#or 4 - >1 hour)"
school[['traveltime','G3']].groupby("traveltime").describe()
```

In [38]:
```
traveltime = school[['traveltime','G3']]
sns.boxplot(x='traveltime', y='G3', data=traveltime)
```

In [39]:
```
#"weekly study time ( 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4
- >10 hours)"
school[['studytime','G3']].groupby("studytime").describe()
```

In [40]:
```
studytime = school[['studytime','G3']]
sns.boxplot(x='studytime', y='G3', data=studytime)
```

In [41]:
```
#number of past class failures (numeric: n if 1<=n<3,  else 4)"
school[['failures','G3']].groupby("failures").describe()
```

In [42]:
```
failures = school[['failures','G3']]
sns.boxplot(x='failures', y='G3', data=failures)
```

In [43]:
```
#extra educational support (binary: yes or no)
school[['schoolsup','G3']].groupby("schoolsup").describe()
```

In [44]:

```
schoolsup = school[['schoolsup','G3']]
sns.boxplot(x='schoolsup', y='G3', data=schoolsup)
```

In [45]:
```
#family educational support (binary: yes or no)
school[['famsup','G3']].groupby("famsup").describe()
```

In [46]:
```
famsup = school[['famsup','G3']]
sns.boxplot(x='famsup', y='G3', data=famsup)
```

In [47]:
```
#extra paid classes within the course subject (Math or Portuguese) (binary: y
es or no)
school[['paid','G3']].groupby("paid").describe()
```

In [48]:
```
paid = school[['paid','G3']]
sns.boxplot(x='paid', y='G3', data=paid)
```

In [49]:
```
#extra-curricular activities (binary: yes or no)
school[['activities','G3']].groupby("activities").describe()
```

In [50]:
```
activities = school[['activities','G3']]
sns.boxplot(x='activities', y='G3', data=activities)
```

In [51]:
```
#attended nursery school (binary: yes or no)
school[['nursery','G3']].groupby("nursery").describe()
```

In [52]:
```
nursery = school[['nursery','G3']]
sns.boxplot(x='nursery', y='G3', data=nursery)
```

In [53]:
```
#wants to take higher education (binary: yes or no)
school[['higher','G3']].groupby("higher").describe()
```

In [54]:
```
higher = school[['higher','G3']]
sns.boxplot(x='higher', y='G3', data=higher)
```

In [55]:
```
#Internet access at home (binary: yes or no)
school[['internet','G3']].groupby("internet").describe()
```

In [56]:
```
internet = school[['internet','G3']]
sns.boxplot(x='internet', y='G3', data=internet)
```

In [57]:

```
#with a romantic relationship (binary: yes or no)
school[['romantic','G3']].groupby("romantic").describe()
```

In [58]:

```
romantic = school[['romantic','G3']]
sns.boxplot(x='romantic', y='G3', data=romantic)
```

In [59]:

```
#quality of family relationships (numeric: from 1 - very bad to 5 - excellent
)
school[['famrel','G3']].groupby("famrel").describe()
```

In [60]:

```
famrel = school[['famrel','G3']]
sns.boxplot(x='famrel', y='G3', data=famrel)
```

In [61]:

```
#free time after school (numeric: from 1 - very low to 5 - very high)
school[['freetime','G3']].groupby("freetime").describe()
```

In [62]:

```
freetime = school[['freetime','G3']]
sns.boxplot(x='freetime', y='G3', data=freetime)
```

In [63]:

```
#going out with friends (numeric: from 1 - very low to 5 - very high)
school[['goout','G3']].groupby("goout").describe()
```

In [64]:

```
goout = school[['goout','G3']]
sns.boxplot(x='goout', y='G3', data=goout)
```

In [65]:

```
#workday alcohol consumption (numeric: from 1 - very low to 5 - very high)
school[['Dalc','G3']].groupby("Dalc").describe()
```

In [66]:

```
Dalc = school[['Dalc','G3']]
sns.boxplot(x='Dalc', y='G3', data=Dalc)
```

In [67]:

```
#weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)
school[['Walc','G3']].groupby("Walc").describe()
```

In [68]:

```
Walc = school[['Walc','G3']]
sns.boxplot(x='Walc', y='G3', data=Walc)
```

In [69]:

```
cross_alc = pd.crosstab(school["Dalc"], school["Walc"])
cross_alc
```

In [70]:

```
plt.show(cross_alc.plot(kind="bar"))
```

In [71]:
```
#current health status (numeric: from 1 - very bad to 5 - very good)
school[['health','G3']].groupby("health").describe()
```

In [72]:
```
health = school[['health','G3']]
sns.boxplot(x='health', y='G3', data=health)
```

In [73]:
```
school.plot(x="absences", y="G3", kind="scatter")
```

In [74]:
```
#Exploratory of numerical attributes
school.plot(x='G1', y='G3', style='o')
```

In [75]:
```
school.plot(x='G2', y='G3', style='o')
```

In [76]:
```
#correlation matrix coefficients
corr = school.corr()
corr
```

In [77]:
```
import seaborn as sns
sns.heatmap(corr, xticklabels=corr.columns.values, yticklabels=corr.columns.v
alues)
```

In [78]:
```
corr['G3']
```

In [79]:
```
#Perform standard linear regression on data
```

In [80]:
```
#Get dummies for the original dataset
school_dummy = pd.get_dummies(school)
school_dummy.head(10)
```

In [81]:
```
school_dummy.describe().T
```

In [82]:
```
from sklearn.preprocessing import MinMaxScaler
mm = MinMaxScaler()
min_max_scaler = mm.fit(school_dummy)
school_norm= min_max_scaler.transform(school_dummy)
school_norm = pd.DataFrame(school_norm, columns=school_dummy.columns)
school_norm.head()
```

In [83]:

```
school_norm.describe()
```

In [84]:

```
# Seperate the target attribute
y = school_norm['G3']
y.head()
```

In [85]:

```
# Drop the target attribute from the training matrix
x = school_norm.drop(["G3"], axis=1)
x.head()
```

In [86]:

```
x.describe().T
```

In [87]:

```
from sklearn.linear_model import LinearRegression, Lasso, Ridge, ElasticNet,
SGDRegressor
# Create linear regression object
linreg = LinearRegression()

# Train the model using the training sets
linreg.fit(x,y)
```

In [88]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [89]:

```
# Let's see predictions for the first 10 instances
print(linreg.predict(x[:10]))
```

In [90]:

```
# Compute RMSE on training data
p = linreg.predict(x)
# Now we can constuct a vector of errors
err = abs(p-y)

# Let's see the error on the first 10 predictions
print (err[:10])
```

In [91]:

```
# Dot product of error vector with itself gives us the sum of squared errors
total_error = np.dot(err,err)

# Compute RMSE
rmse_train = np.sqrt(total_error/len(p))
print (rmse_train)
```

In [92]:

```
# We can view the regression coefficients
```

```
print ('Regression Coefficients: \n', linreg.coef_)
```

In [93]:
```
#correlation between the predicted and actual values of the target attribute.
import matplotlib.pyplot as plot
plot.scatter(y, p, color = 'red', s=0.1)
plot.title('Actual vs Predicted (Final Score)')
plot.xlabel('Actual')
plot.ylabel('Predicted')
plot.show()
```

In [94]:
```
import pylab as pl
# Plot outputs
%matplotlib inline
pl.plot( y, p,'ro',markersize=0.5)
pl.plot([0,1],[0,1], 'g-')
plot.title('Actual vs Predicted (Final Score)')
pl.xlabel('Actual')
pl.ylabel('Predicted')
pl.show()
```

In [95]:
```
# 10-fold crossvalidation were performed to compare the cross-validation RMSE
to the training
# RMSE using KFoldmodule from sklearn.model_selection
```

In [96]:
```
from sklearn.model_selection import KFold
x = np.array(x)
y = np.array(y)
n = 10
kf = KFold(n_splits=n)
xval_err = 0
for train,test in kf.split(x):
    linreg.fit(x[train],y[train])
    p = linreg.predict(x[test])
    e = p-y[test]
    xval_err += np.sqrt(np.dot(e,e)/len(x[test]))

rmse_10cv = xval_err/n
```

In [97]:
```
method_name = 'Simple Linear Regression'
print('Method: %s' %method_name)
print('RMSE on training: %.4f' %rmse_train)
print('RMSE on 10-fold CV: %.4f' %rmse_10cv)
```

In [98]:
```
#Using scikit-learn regression model from sklearn.linear_model, a function wa
s created to takes
#as input the training data, target variable, number of cross validations, th
e model method, and
#percentile step. The script returns the optimal percentage of the most infor
mative features to
#use. feature_selection.SelectPercentile was used to find the most informativ
e variables.
#The mean scores was determined using model_selection.cross_val_score.
```

In [99]:
```
from sklearn import feature_selection
x = school_norm.drop(["G3"], axis=1)
y = school_norm['G3']
```

In [100]:
```
from sklearn import model_selection

def feature_select(x_train, target, crossv, model, percentile_step):
    if model == 'regression':
        method = LinearRegression()
        feature_method = feature_selection.f_regression

    percentiles = np.array(range(1, 100, percentile_step))
    results = []
    columns = []
    f = []
    for i in range(1, 100, percentile_step):
        fs = feature_selection.SelectPercentile(feature_method, percentile=i)
        x_train_fs = fs.fit_transform(x_train, target)
        scores = model_selection.cross_val_score(method, x_train_fs, target,
cv=crossv, scoring='neg_mean_absolute_error')
        pos_score = 1 + scores
        #print (i,pos_score.mean())
        results = np.append(results, pos_score.mean())
        array = (x_train.columns[fs.get_support()].values).tolist()
        columns += [array]
        f_list = []
        for ii in range(len(array)):
            f_list.append(fs.scores_[ii])
        f += [f_list]

    optimal_percentile = np.where(results == results.max())[0]
    print('The most informative variables and their weights are:', "\n")
    for i in range(len(columns[int(optimal_percentile)])):
        print(columns[int(optimal_percentile)][i],'\t', f[int(optimal_percent
ile)][i] )

    print("\n")
    print ("Optimal percentile of features:{0}".format(percentiles[optimal_pe
rcentile]), "\n")
    optimal_num_features = int(percentiles[optimal_percentile]*len(x.columns)
/100) +1
    print ("Optimal number of features:{0}".format(optimal_num_features), "\n
")

    # Plot percentile of features VS. cross-validation scores
    import pylab as pl
    pl.figure()
    pl.xlabel("Percentage of features selected")
    pl.ylabel("Cross validation Mean Absolute Error")
    pl.plot(percentiles,results)
```

In [101]:

```
feature_select(x, y, 5,'regression',5)
```

In [102]:

```
#Ridge regression was then performed to avoid the problem of overfitting and
the failure to find
#unique solutions. Ridge help improve the model and add a penalty to avoid ex
tracting noise.
#Lasso Regression using the modules from sklearn.linear_model was implemented
in addition to
#performing systematic model selection to identify the optimal alpha paramete
r.
#A 20%-80% randomized data split was used to set aside the test portion and u
se it later for
#testing purposes to evaluate the model. Only the 80% randomized chosen data
will be used for
#training the model. A function will then be created to take the training dat
a, target variable,
#number of cross validations, and model method to be trained (Lasso or Regres
sion).
#The script runs to find the optimal alpha that has the least average of RMSE
on n numbers of
#cross validations and RMSE on the training set. Furthermore, error values (R
MSE) on the training
```

```
#and cross-validation splits were plotted across the specified values of the
alpha parameters.
```

In [103]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2, rand
om_state=33)
```

In [104]:

```
X_trainp = np.array(X_train)
y_trainp = np.array(y_train)
X_test = np.array(X_test)
y_test = np.array(y_test)
```

In [105]:

```
def select_model(trainp, targetp, name, ncv):

    if name == 'ridge':
        print('Ridge Regression')
        print('alpha\t RMSE_train\t RMSE_{}cv\n'.format(ncv))
    elif name == 'lasso':
        print('Lasso Regression\n')
        print('alpha\t RMSE_train\t RMSE_5cv\n')
    alpha = np.linspace(.01,20,50)
    t_rmse = np.array([])
    cv_rmse = np.array([])
    avg_rmse = np.array([])
    optimal = np.array([])

    for a in alpha:

        if name == 'lasso':
            met = Lasso(alpha=a)
        elif name == 'ridge':
            met = Ridge(alpha=a)

        # computing the RMSE on training data
        met.fit(trainp,targetp)
        p = met.predict(trainp)
        err = p-targetp
        total_error = np.dot(err,err)
        rmse_train = np.sqrt(total_error/len(p))

        # computing RMSE using ncv-fold cross validation
        kf = KFold(n_splits=ncv)
        xval_err = 0
        for train, test in kf.split(trainp):
            met.fit(trainp[train], targetp[train])
            p = met.predict(trainp[test])
            err = p - targetp[test]
            xval_err += np.sqrt(np.dot(err,err)/len(trainp[test]))

        rmse_ncv = xval_err/ncv
        rmse_avg = (rmse_ncv+rmse_train)/2
        optimal = np.append(optimal, [a])
        t_rmse = np.append(t_rmse, [rmse_train])
        cv_rmse = np.append(cv_rmse, [rmse_ncv])
        avg_rmse = np.append(avg_rmse, [rmse_avg])
        print('{:.3f}\t {:.4f}\t\t {:.4f}'.format(a,rmse_train,rmse_ncv))

    optimal_avg_rmse = avg_rmse[np.where(avg_rmse == avg_rmse.min())[0]][0]
    optimal_alpha = optimal[np.where(avg_rmse == avg_rmse.min())[0]][0]
    print('Optimal alpha = {:.3f} with RMSE avarage = {:.5f}'.format(optimal_
alpha,optimal_avg_rmse))

    pl.plot(alpha, t_rmse, label='RMSE-Train')
    pl.plot(alpha, cv_rmse, label='RMSE_XVal')
    pl.legend( ('RMSE-Train', 'RMSE_XVal') )
    pl.ylabel('RMSE')
    pl.xlabel('Alpha')
    pl.show()
    return(optimal_alpha)
```

In [106]:

```
#Setting up optimal alpha to apply on the test set using ridge regression
alpha_r = select_model(X_trainp, y_trainp, 'ridge', 5)
```

In [107]:

```
# Create linear regression object with a ridge coefficient equals to optimal
alpha_r
ridge_test = Ridge(fit_intercept=True, alpha=alpha_r)

# Train the model using the training set
ridge_test.fit(X_trainp,y_trainp)
```

In [108]:

```
# Compute RMSE on testing data
p = ridge_test.predict(X_test)
err = p-y_test
total_error = np.dot(err,err)
rmse_test = np.sqrt(total_error/len(p))
```

```
method_name = 'Ridge Regression'
print('Method: %s' %method_name)
print('RMSE on testing: %.4f' %rmse_test)
```

In [109]:

```
#Setting up optimal alpha to apply on the test set using lasso regression
alpha_l = select_model(X_trainp, y_trainp, 'lasso', 5)
```

In [110]:

```
# Create linear regression object with a lasso coefficient equals to optimal
alpha_l
lasso_test = Lasso(fit_intercept=True, alpha=alpha_l)

# Train the model using the training set
lasso_test.fit(X_trainp,y_trainp)
```

In [111]:

```
# Compute RMSE on testing data
p = lasso_test.predict(X_test)
err = p-y_test
total_error = np.dot(err,err)
rmse_test = np.sqrt(total_error/len(p))


method_name = 'Lasso Regression'
print('Method: %s' %method_name)
print('RMSE on testing: %.4f' %rmse_test)
```