Students Performance Analysis And What To Infer From It

Gaspare Mattarella

6/14/2021

Contents

Abstract

In the first part of this paper we are going to perform a regression analysis on a dataset concerning students performances in secondary school in Portugal. Our goal is to find the variables that most explain the variances, understand how and possibly why this would be the case. To achieve this goal we will use with different models starting from the basic linear regression and going on selecting the best features with a stepwise selection model, a LASSO and finally a Robust regresssion. Then we will try to obtain additional informative power thanks to two Tree Based models. In the second part of the assignment we will instead see how a simple K means algorithm can well divide the dataset in two clusters representing good and bad performative students.

1 PART I: SUPERVISED LEARNING

1.1 Introduction

In this analysis I am going to dive into Portuguese public education trying to predict and infer secondary school students performance. In Portugal, the secondary education consists of 3 years of schooling, preceding 9 years of basic education and followed by higher education. Most of the students join the public and free education system. There are several courses (e.g. Sciences and Technologies, Visual Arts) that share core subjects such as the Portuguese Language and Mathematics, subjects on which the dataset is constructed. A 20-point grading scale is used, where 0 is the lowest grade and 20 is the perfect score. During the school year, students are evaluated in three periods and the last evaluation (G3 of Table 1) corresponds to the final grade.

The database, that can be retrieved at the following link, was built from two sources: school reports, based on paper sheets and including few attributes (i.e. the three period grades and number of school absences); and questionnaires, used to complement the previous information. The Goal of this analysis is to understand which variables, among the ones available to us, can help us explain the variability of student performances. Here a brief description of the variables in the dataset:

Table 1

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

30. absences - number of school absences (numeric: from 0 to 93)
31. G1 - first period grade (numeric: from 0 to 20)
32. G2 - second period grade (numeric: from 0 to 20)
33. G3 - final grade (numeric: from 0 to 20, output target)

In the table below we can observe the distribution of all the numeric variable, check that there are no missing values, check for anomalies in the range of the data or in their mean and standard deviation. For what we can see, everything is in the right place.

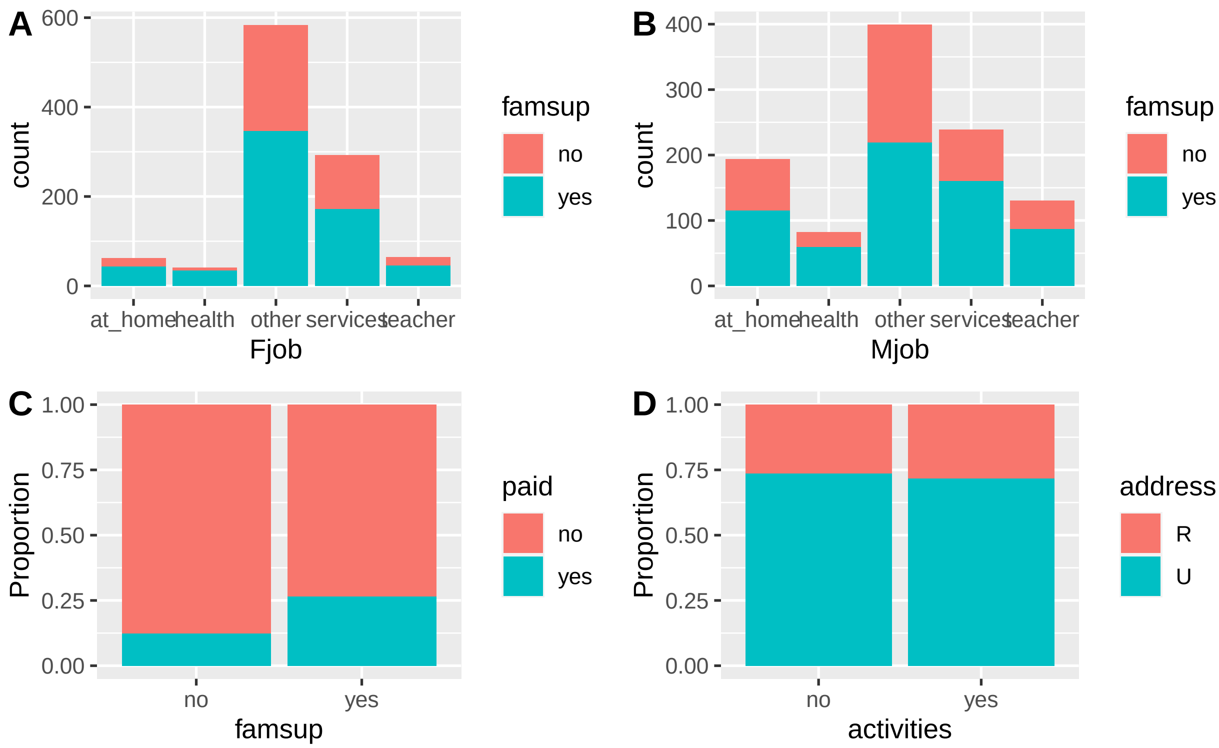
```
## Variable   |  Mean |   SD | IQR |          Range | Skewness | Kurtosis |    n | n_Missing
## ------------------------------------------------------------------------------------------
## age        | 16.73 | 1.24 |   2 | [15.00, 22.00] |     0.43 |     0.04 | 1044 |         0
## Medu       |  2.60 | 1.12 |   2 |   [0.00, 4.00] |    -0.14 |    -1.23 | 1044 |         0
## Fedu       |  2.39 | 1.10 |   2 |   [0.00, 4.00] |     0.12 |    -1.17 | 1044 |         0
## traveltime |  1.52 | 0.73 |   1 |   [1.00, 4.00] |     1.37 |     1.48 | 1044 |         0
## studytime  |  1.97 | 0.83 |   1 |   [1.00, 4.00] |     0.67 | 6.62e-03 | 1044 |         0
## failures   |  0.26 | 0.66 |   0 |   [0.00, 3.00] |     2.78 |     7.50 | 1044 |         0
## famrel     |  3.94 | 0.93 |   1 |   [1.00, 5.00] |    -1.06 |     1.29 | 1044 |         0
## freetime   |  3.20 | 1.03 |   1 |   [1.00, 5.00] |    -0.18 |    -0.36 | 1044 |         0
## goout      |  3.16 | 1.15 |   2 |   [1.00, 5.00] |     0.04 |    -0.84 | 1044 |         0
## Dalc       |  1.49 | 0.91 |   1 |   [1.00, 5.00] |     2.16 |     4.48 | 1044 |         0
## Walc       |  2.28 | 1.29 |   2 |   [1.00, 5.00] |     0.63 |    -0.78 | 1044 |         0
## health     |  3.54 | 1.42 |   2 |   [1.00, 5.00] |    -0.50 |    -1.08 | 1044 |         0
## absences   |  4.43 | 6.21 |   6 |  [0.00, 75.00] |     3.74 |    26.60 | 1044 |         0
## G1         | 11.21 | 2.98 |   4 |  [0.00, 19.00] |     0.08 |    -0.33 | 1044 |         0
## G2         | 11.25 | 3.29 |   4 |  [0.00, 19.00] |    -0.50 |     1.34 | 1044 |         0
## G3         | 11.34 | 3.86 |   4 |  [0.00, 20.00] |    -0.99 |     1.74 | 1044 |         0
```

Something I want to be highlighted is that the three grades variables (G1,G2,G3) have very similar ranges, mean and standard deviation. Not that it is unexpected but it is definitely a problem. Trying to predict the final grade (G3) using also G1 and G2 as predictors among the others will likely lead to excellent performances altought it is indeed like cheating. That's why I am going to deal with that in a few lines. First, let's make us acquainted with the data.

1.2 Exploratory Data Analysis

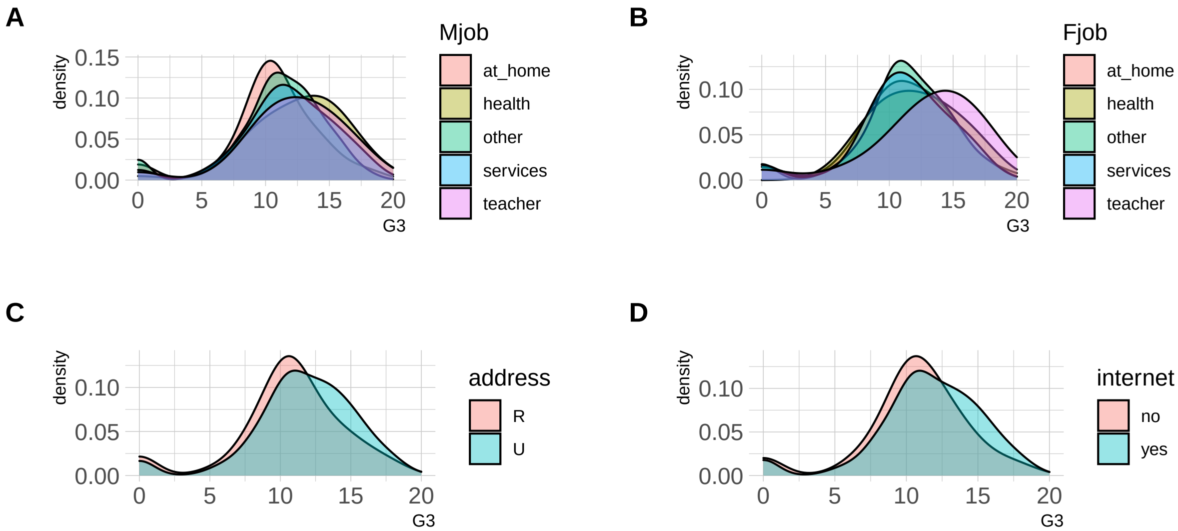
From the first two images, that relate parents' occupation with the presence of family support, we can see that when it comes to fathers at home, teachers but especially in healthcare, family support is much more predominant than in the others. Same thing for mothers with a more balance for mothers at home and a little bit more for mothers in services. Third image shows us that for kids with family support is more likely to receive extra paid classes. Fourth image shows instead that there is almost no difference for kids who lives in rural and urban area to partecipate extra curricular activities. Surprisingly, I must admit.

Figure 1

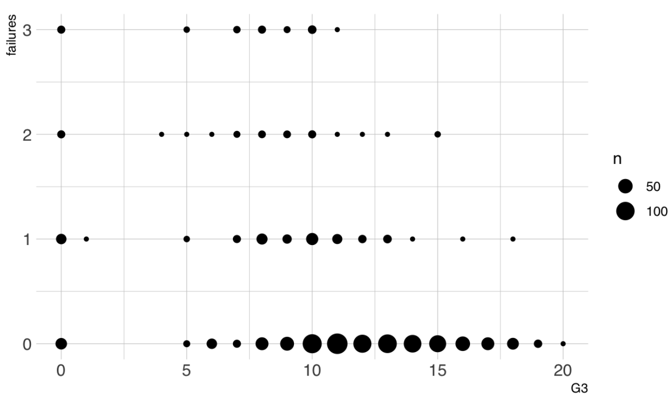


In the next set of images we can spot the relations between the parents' job and kid's final grade. For Mother's occupation, it is clear that 'at home' mothers seems to have the smaller mean and smaller variance. We can move orderly trough higher mean and higher variance with other, services, teachers and finally, with an elegant skew to the right, healthcare. For what it concerns father's job, we can instead note an overlapping of all the occupation, altought with different variability, except for teachers, which again present a clear skew to the right. Third image shows us that kids who live in urban areas have slightly higher means and thinner tails to the left of the distribution. Almost an identical picture we can observe when it comes to having or not internet at home.

Figure 2



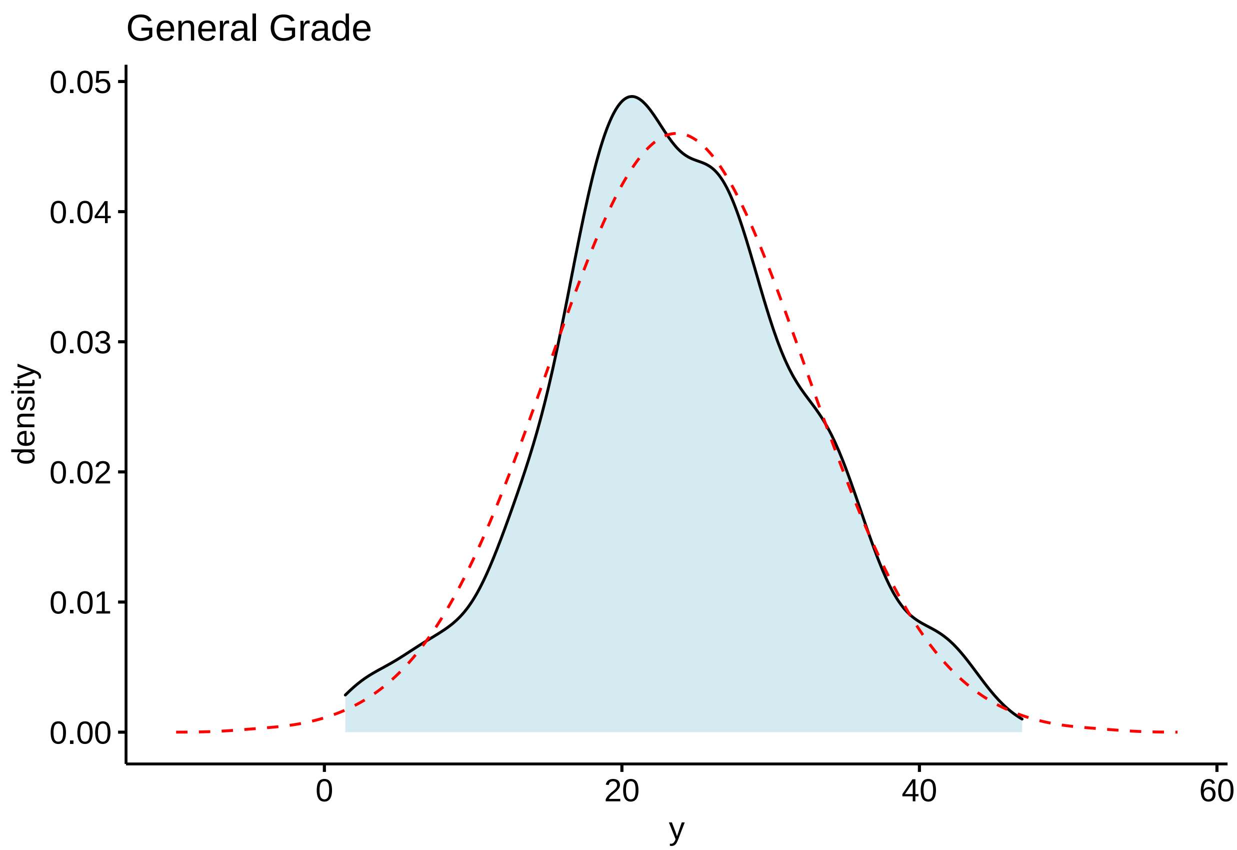
Another image worth of mention, the below one, highlights the relation between failures and high grades. This will be important in later analysis. Before going further in exploration, we need to address the problem of response variable and the intermediate grades. As mentioned before, the three grades are *extremely* correlated and including G1 and/or G2 as predictors could be somehow useless to our scope, i.e. infer which among our regressors are statistically significant in explaining the variance of student performance. My is the following: I won't throw away G1 and G2 because they still may contain valuable information, instead I will take the average between all of them creating in so a new variable which represents the general performance of the student, not linked to a particular period of time. Then I will apply a BoxCox transformation to the variable that we will, from now on, simply call y just to obtain a normal distribution of the response variable.




```
## Box-Cox Transformation
##
## 1044 data points used to estimate Lambda
##
## Input data summary:
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.30    9.30   11.30   11.27   13.30   19.30
##
## Largest/Smallest: 14.8
```

```
## Sample Skewness: -0.289
##
## Estimated Lambda: 1.3
```

The "normalization" of the response variable is a prerequisite for the analysis of variance we're going to perform on the data. In the following table we can see the density plot and an overlaying normal distribution.




```
## Levene's Test for Homogeneity of Variance (center = median)
##         Df F value Pr(>F)
## group    4    1.37 0.2423
##       1039
```

The other prerequisite is the homogeneity of variance across groups that we're going to test with a Levene Test. We prefer a robust Levene test to a classic Bartlett because the latter is sensitive to lack of normality.

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = y ~ Mjob, data = df)
##
## Linear Hypotheses:
##                          Estimate Std. Error t value Pr(>|t|)
## health - at_home == 0      5.5021     1.0952   5.024 5.95e-06 ***
## other - at_home == 0       1.3713     0.7277   1.884  0.59802
## services - at_home == 0    2.9945     0.8035   3.727  0.00204 **
```

```
## teacher - at_home == 0    4.5140     0.9424   4.790 1.91e-05 ***
## other - health == 0      -4.1308     1.0081  -4.098  0.00045 ***
## services - health == 0   -2.5075     1.0641  -2.357  0.18632
## teacher - health == 0    -0.9881     1.1725  -0.843  1.00000
## services - other == 0     1.6233     0.6801   2.387  0.17168
## teacher - other == 0      3.1427     0.8396   3.743  0.00192 **
## teacher - services == 0   1.5194     0.9061   1.677  0.93859
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
```

To test for more than 2 groups we need to remember that regular p-values will be meaningless and that we need to perform a Multiple Comparisons of Means, then we must proceed with a Bonferroni correction of the p-values. From table and plot above we can see how the differences between the specif jobs and "at home" are all statistically significant while they're not different between them nor with "other" with the exception of "teacher".

```
## Levene's Test for Homogeneity of Variance (center = median)
##         Df F value Pr(>F)
## group    4  1.4102 0.2285
##       1039
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = y ~ Fjob, data = df)
##
## Linear Hypotheses:
##                         Estimate Std. Error t value Pr(>|t|)
## health - at_home == 0    2.76614    1.68922   1.638 1.000000
## other - at_home == 0     0.70188    1.12091   0.626 1.000000
## services - at_home == 0  0.66861    1.17347   0.570 1.000000
## teacher - at_home == 0   5.53467    1.48972   3.715 0.002138 **
## other - health == 0     -2.06425    1.35581  -1.523 1.000000
## services - health == 0  -2.09752    1.39957  -1.499 1.000000
## teacher - health == 0    2.76853    1.67363   1.654 0.983881
## services - other == 0   -0.03327    0.60146  -0.055 1.000000
## teacher - other == 0     4.83278    1.09727   4.404 0.000117 ***
## teacher - services == 0  4.86606    1.15091   4.228 0.000257 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
```

For what it concerns father's job, the only significant difference comes from the "teacher" group with all the rest except that for "health". Same thing we do for the reason variable, observing that "reputation" is actually significantly different from the other values.

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
```

```
## Fit: aov(formula = y ~ reason, data = df)
##
## Linear Hypotheses:
##                            Estimate Std. Error t value Pr(>|t|)
## home - course == 0           1.1843     0.6601   1.794  0.43843
## other - course == 0         -0.1238     0.9022  -0.137  1.00000
## reputation - course == 0     3.1396     0.6683   4.698 1.79e-05 ***
## other - home == 0           -1.3081     0.9607  -1.362  1.00000
## reputation - home == 0       1.9553     0.7454   2.623  0.05304 .
## reputation - other == 0      3.2633     0.9664   3.377  0.00456 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
```

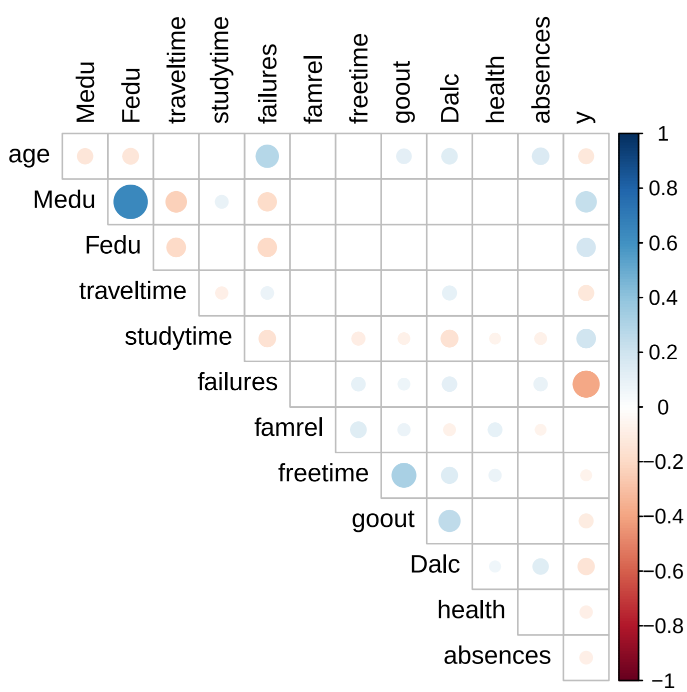
```
##               Df Sum Sq Mean Sq F value   Pr(>F)
## internet       1   1024  1024.0   14.47 0.000151 ***
## Residuals   1042  73754    70.8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##               Df Sum Sq Mean Sq F value   Pr(>F)
## address        1   1168  1168.0   16.53 5.14e-05 ***
## Residuals   1042  73610    70.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

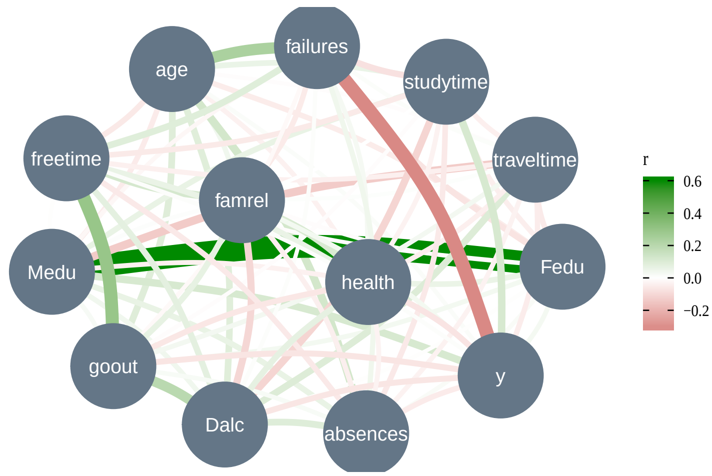
We can say the same thing for the binary classes "internet" and "address".

In the figure below we can see the correlation matrix between all the numerical variables. The spaces left blank are statistically insignificant.

We can see that variable most correlated with our response variable is "failures". "famrel" is the only one not statistically significant and "Medu" and "Fedu" seems pretty correlated too. Also note how strongly they are correlated between them.



In the next image we can instead explore the "partial" correlation between the same variables which, by definition, is the correlation of two variables while controlling for a third or more other variables. Here the effect of Fedu and Medu on the response variable seems to be less than before. Same power for "failures" instead.



In the table below we present the transformed data. The categorical variables with more than 2 classes were *One-Hot* encoded so that we now have only numerical variables.

```
## Variable            |  Mean |   SD |   IQR |          Range | Skewness | Kurtosis |    n | n_Missing
## -----------------------------------------------------------------------------------------------------
## Fjob_at_home        |  0.06 | 0.24 |  0.00 |   [0.00, 1.00] |     3.73 |    11.96 | 1044 |         0
## Fjob_health         |  0.04 | 0.19 |  0.00 |   [0.00, 1.00] |     4.75 |    20.61 | 1044 |         0
## Fjob_services       |  0.28 | 0.45 |  1.00 |   [0.00, 1.00] |     0.98 |    -1.04 | 1044 |         0
## Fjob_teacher        |  0.06 | 0.24 |  0.00 |   [0.00, 1.00] |     3.63 |    11.19 | 1044 |         0
## Mjob_at_home        |  0.19 | 0.39 |  0.00 |   [0.00, 1.00] |     1.62 |     0.62 | 1044 |         0
## Mjob_health         |  0.08 | 0.27 |  0.00 |   [0.00, 1.00] |     3.14 |     7.86 | 1044 |         0
## Mjob_services       |  0.23 | 0.42 |  0.00 |   [0.00, 1.00] |     1.29 |    -0.33 | 1044 |         0
## Mjob_teacher        |  0.12 | 0.33 |  0.00 |   [0.00, 1.00] |     2.28 |     3.19 | 1044 |         0
## reason_course       |  0.41 | 0.49 |  1.00 |   [0.00, 1.00] |     0.36 |    -1.87 | 1044 |         0
## reason_home         |  0.25 | 0.43 |  0.00 |   [0.00, 1.00] |     1.17 |    -0.62 | 1044 |         0
## reason_reputation   |  0.24 | 0.43 |  0.00 |   [0.00, 1.00] |     1.24 |    -0.48 | 1044 |         0
## sex                 |  0.43 | 0.50 |  1.00 |   [0.00, 1.00] |     0.27 |    -1.93 | 1044 |         0
## age                 | 16.73 | 1.24 |  2.00 | [15.00, 22.00] |     0.43 |     0.04 | 1044 |         0
## address             |  0.73 | 0.45 |  1.00 |   [0.00, 1.00] |    -1.02 |    -0.96 | 1044 |         0
## famsize             |  0.29 | 0.46 |  1.00 |   [0.00, 1.00] |     0.91 |    -1.17 | 1044 |         0
## Pstatus             |  0.88 | 0.32 |  0.00 |   [0.00, 1.00] |    -2.40 |     3.78 | 1044 |         0
## Medu                |  2.60 | 1.12 |  2.00 |   [0.00, 4.00] |    -0.14 |    -1.23 | 1044 |         0
## Fedu                |  2.39 | 1.10 |  2.00 |   [0.00, 4.00] |     0.12 |    -1.17 | 1044 |         0
## traveltime          |  1.52 | 0.73 |  1.00 |   [1.00, 4.00] |     1.37 |     1.48 | 1044 |         0
## studytime           |  1.97 | 0.83 |  1.00 |   [1.00, 4.00] |     0.67 | 6.62e-03 | 1044 |         0
## failures            |  0.26 | 0.66 |  0.00 |   [0.00, 3.00] |     2.78 |     7.50 | 1044 |         0
## schoolsup           |  0.11 | 0.32 |  0.00 |   [0.00, 1.00] |     2.43 |     3.93 | 1044 |         0
## famsup              |  0.61 | 0.49 |  1.00 |   [0.00, 1.00] |    -0.46 |    -1.79 | 1044 |         0
## paid                |  0.21 | 0.41 |  0.00 |   [0.00, 1.00] |     1.42 |     0.02 | 1044 |         0
## activities          |  0.49 | 0.50 |  1.00 |   [0.00, 1.00] |     0.02 |    -2.00 | 1044 |         0
## higher              |  0.91 | 0.28 |  0.00 |   [0.00, 1.00] |    -2.97 |     6.86 | 1044 |         0
```

```
## internet          |  0.79 | 0.41 |  0.00 |   [0.00, 1.00] |    -1.44 |     0.08 | 1044 |          0
## romantic          |  0.36 | 0.48 |  1.00 |   [0.00, 1.00] |     0.61 |    -1.64 | 1044 |          0
## famrel            |  3.94 | 0.93 |  1.00 |   [1.00, 5.00] |    -1.06 |     1.29 | 1044 |          0
## freetime          |  3.20 | 1.03 |  1.00 |   [1.00, 5.00] |    -0.18 |    -0.36 | 1044 |          0
## goout             |  3.16 | 1.15 |  2.00 |   [1.00, 5.00] |     0.04 |    -0.84 | 1044 |          0
## Dalc              |  1.49 | 0.91 |  1.00 |   [1.00, 5.00] |     2.16 |     4.48 | 1044 |          0
## Walc              |  2.28 | 1.29 |  2.00 |   [1.00, 5.00] |     0.63 |    -0.78 | 1044 |          0
## health            |  3.54 | 1.42 |  2.00 |   [1.00, 5.00] |    -0.50 |    -1.08 | 1044 |          0
## absences          |  4.43 | 6.21 |  6.00 |  [0.00, 75.00] |     3.74 |    26.60 | 1044 |          0
## y                 | 23.69 | 8.47 | 10.75 |  [1.41, 46.91] |     0.02 |    -0.08 | 1044 |          0
```

1.3 Modeling

Now that we are more familiar with the data and the relations between them, we can actually proceed modeling them and trying to gain some additional information.

We can actually create a model in 3 different ways

1. Binary classification

   - y > 10: pass

   - y < 10: fail

2. five-level classification based on Erasmus grade conversion system

   - 16-20: very good

   - 14-15: good

   - 12-13: satisfactory

   - 10-11: sufficient

   - 0-9 : fail

3. Regression (Predicting y)

Now, the real question is: what do we want from this? Do we want to classify and predict whether a kid is going to pass or fail the exam? It might actually be useful for social services and people whose job is to prevent kids from failing, intervening in the right moment. Do we need to classify and predict who's going to be very good rather than who's going to be sufficient at best? Maybe yes, just like above.

What I personally am more interested in is the third option. I find extremely important trying to clarify and infer the precise effect of every variable we have and that's the reason I am going to go with Linear Regression first so that we begin with the highest level of interpretability and just than trying to dive into more complex models.

1.3.1 Baseline Model - Simple Linear Regression

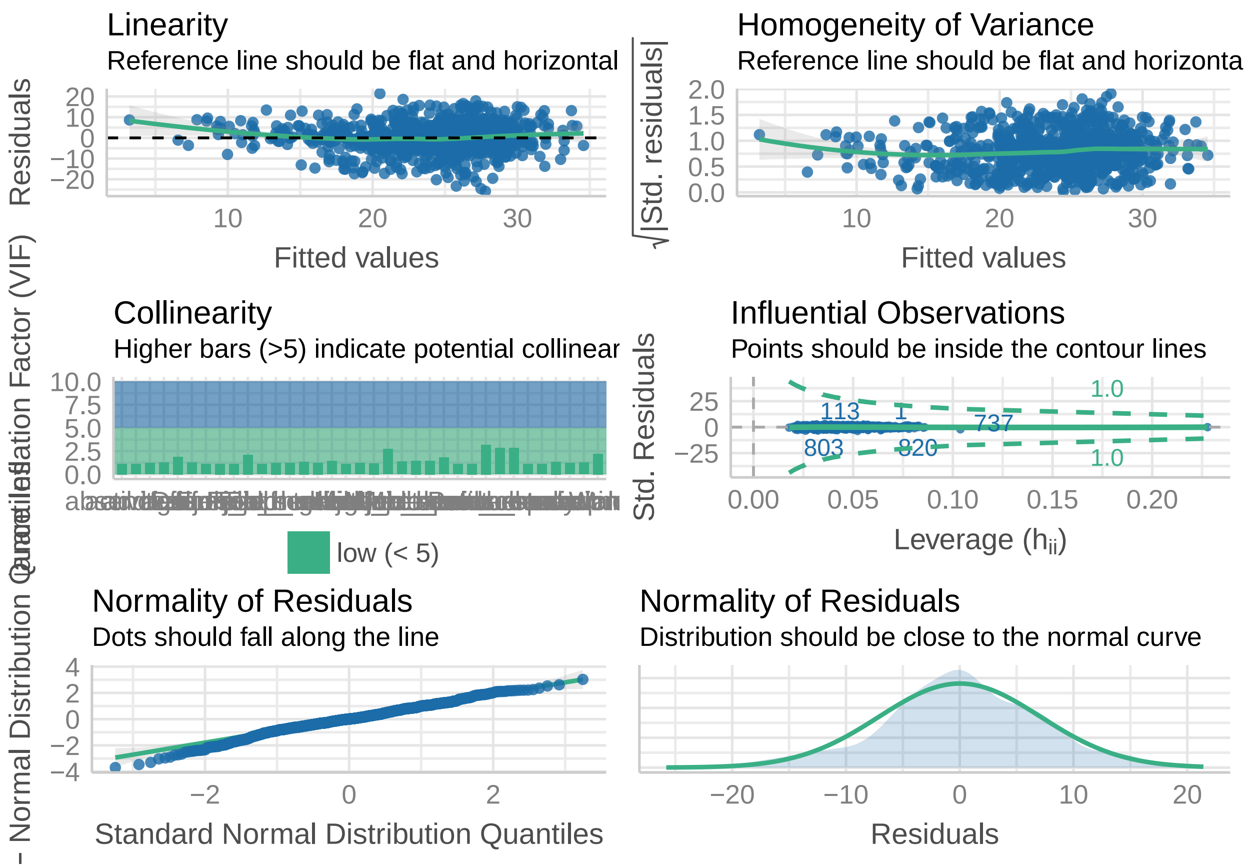
First thing first, we split the dataset in a train and a test set (80%/20%)

```
## [1] 207  36
```

```
## [1] 837  36
```

Now we run a Linear Regression with all the variables we dispone. Than we check the model assumptions.

```
## Loading required namespace: qqplotr
```



From this graphical check, everything seems to be ok except maybe for a little heteroskedasticity of the variance. We hence check all of them with the proper tests.

```
## Warning: Non-normality of residuals detected (p < .001).
```

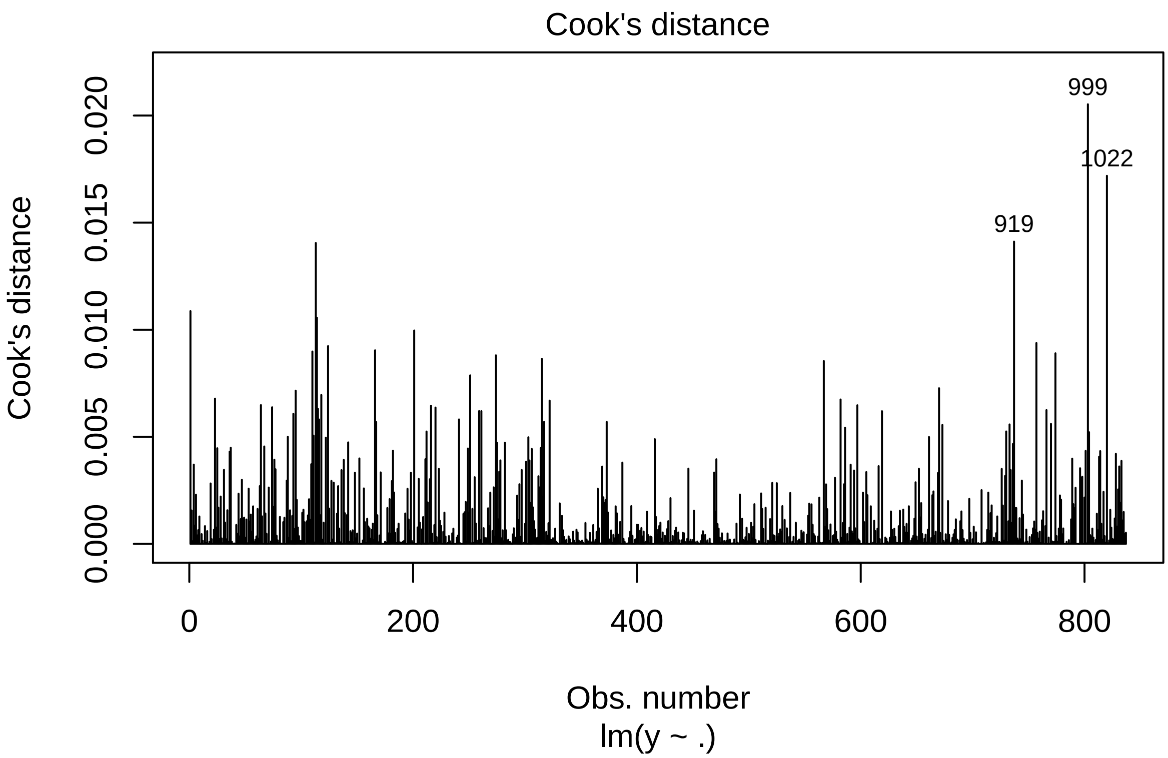
```
## Warning: Heteroscedasticity (non-constant error variance) detected (p = 0.016).
```

```
## OK: Residuals appear to be independent and not autocorrelated (p = 0.318).
```

```
## # Check for Multicollinearity
##
## Low Correlation
##
##              Term  VIF Increased SE Tolerance
##       Fjob_at_home 1.13         1.06      0.88
##        Fjob_health 1.26         1.12      0.80
##      Fjob_services 1.26         1.12      0.79
##       Fjob_teacher 1.35         1.16      0.74
##       Mjob_at_home 1.39         1.18      0.72
##        Mjob_health 1.47         1.21      0.68
##      Mjob_services 1.47         1.21      0.68
##       Mjob_teacher 1.82         1.35      0.55
##      reason_course 3.20         1.79      0.31
##        reason_home 2.84         1.68      0.35
##  reason_reputation 2.85         1.69      0.35
##                sex 1.35         1.16      0.74
```

```
##                 age 1.29          1.14       0.78
##             address 1.23          1.11       0.82
##             famsize 1.14          1.07       0.88
##             Pstatus 1.14          1.07       0.88
##                Medu 2.74          1.65       0.37
##                Fedu 2.10          1.45       0.48
##          traveltime 1.28          1.13       0.78
##           studytime 1.22          1.10       0.82
##            failures 1.27          1.13       0.79
##           schoolsup 1.12          1.06       0.89
##              famsup 1.15          1.07       0.87
##                paid 1.10          1.05       0.91
##          activities 1.12          1.06       0.89
##              higher 1.24          1.11       0.81
##            internet 1.19          1.09       0.84
##            romantic 1.13          1.06       0.89
##              famrel 1.12          1.06       0.90
##            freetime 1.26          1.12       0.79
##               goout 1.44          1.20       0.70
##                Dalc 1.89          1.37       0.53
##                Walc 2.22          1.49       0.45
##              health 1.13          1.06       0.89
##            absences 1.13          1.06       0.88
```

The model appears to be fine for what it concerns Autocorrelation of the residuals and Multicollinearity. Hypothesis of Normality of the residuals and Homoscedasticity were instead rejected. Normality of the residuals is actually not a real problem since our sample is big enough to use the properties of the Central Limit Theory and from the graph we could spot how *not* severe this normality is. We will instead address Heteroscedasticity with proper robust standard errors from now.



Here above we checked for outliers with the Cook's distance. We can spot that other than 3/5 evident severe outliers there are also a lot of mild outliers. That's why we will ignore them for now and take care of this problem later on with a Robust Regression.

1.3.2 Inference

Below we can observe the coefficients with Robust Standard Errors obtained through White's estimator. We have an intercept that is very significant, Father's job "teacher" has a pretty high coefficient of 3.2 and togheter with "services" that is instead negative they both are significant as previewsly seen. For Mother'job, as we already know, working in "health" has a pretty high coefficient too and still significant. Also "services". Next we have famsize, basically having siblings seems to be positive and significant. Furthermore, we note studytime that is positive and significant and, as we already knew, failures that has the highest coefficient so far and it is very significative. The following variables sems to be a clear case of "Spurious Correlation" since School Support is higly negative and significative. It doesn't of course suggest that receiving schooling support worsen performance but that all the people receiving it probably have previus difficulties. Same thing for the ones who receive "paid" extra lessons. Not shocking at all, the desire to continue higher studies is super positive and significant. Strange result is instead the significance of being in a sentimental relation and that being negative. It appears that kids going out a lot have significantlly worse performance and, unexplainable enough, health status seems to have a negative and significant effect on performance.

```
##
## t test of coefficients:
##
##                    Estimate Std. Error t value  Pr(>|t|)
## (Intercept)       15.567522   4.376710  3.5569 0.0003973 ***
## Fjob_at_home      -1.123453   1.188450 -0.9453 0.3447858
## Fjob_health       -0.509885   1.336606 -0.3815 0.7029503
```

```
## Fjob_services     -1.180226   0.599645 -1.9682 0.0493887 *
## Fjob_teacher       3.242395   1.331246  2.4356 0.0150838 *
## Mjob_at_home       0.210881   0.759498  0.2777 0.7813458
## Mjob_health        2.908366   1.056450  2.7530 0.0060396 **
## Mjob_services      1.340805   0.738369  1.8159 0.0697595 .
## Mjob_teacher      -0.728374   1.068001 -0.6820 0.4954377
## reason_course     -0.437629   0.886845 -0.4935 0.6218178
## reason_home        0.348718   0.924838  0.3771 0.7062302
## reason_reputation  0.598204   0.964627  0.6201 0.5353417
## sex               -0.246555   0.593358 -0.4155 0.6778689
## age                0.303060   0.234825  1.2906 0.1972224
## address            0.975543   0.645899  1.5104 0.1313450
## famsize            1.299115   0.570018  2.2791 0.0229248 *
## Pstatus           -0.126757   0.875631 -0.1448 0.8849360
## Medu               0.563027   0.374013  1.5054 0.1326239
## Fedu               0.382797   0.336049  1.1391 0.2549976
## traveltime        -0.332330   0.365255 -0.9099 0.3631711
## studytime          1.244032   0.321107  3.8742 0.0001157 ***
## failures          -3.752754   0.415992 -9.0212 < 2.2e-16 ***
## schoolsup         -3.584231   0.773425 -4.6342 4.181e-06 ***
## famsup            -0.447674   0.540951 -0.8276 0.4081607
## paid              -1.870608   0.598604 -3.1250 0.0018424 **
## activities         0.102614   0.532437  0.1927 0.8472239
## higher             3.453967   0.934247  3.6971 0.0002330 ***
## internet           1.099915   0.689265  1.5958 0.1109325
## romantic          -1.316401   0.548228 -2.4012 0.0165686 *
## famrel             0.073976   0.273548  0.2704 0.7868981
## freetime           0.330384   0.288036  1.1470 0.2517151
## goout             -0.753286   0.269896 -2.7910 0.0053792 **
## Dalc              -0.257708   0.321295 -0.8021 0.4227366
## Walc              -0.131624   0.279749 -0.4705 0.6381198
## health            -0.626948   0.180919 -3.4654 0.0005577 ***
## absences          -0.030691   0.040327 -0.7610 0.4468539
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## # Indices of model performance
##
## AIC      |      BIC |    R2 | R2 (adj.) |  RMSE | Sigma
## ---------------------------------------------------------
## 5724.869 | 5899.873 | 0.305 |     0.274 | 7.076 | 7.233
```

One thing we should notice is that the ratio between significant and unsignificant variables is quite even, meaning that we're feeding our model with a lot of useless information. That's why in the next section we are going to perform an automatic feature selection with the help of two of the most efficient methods, a mix of forward and back stepwise selection and the LASSO.

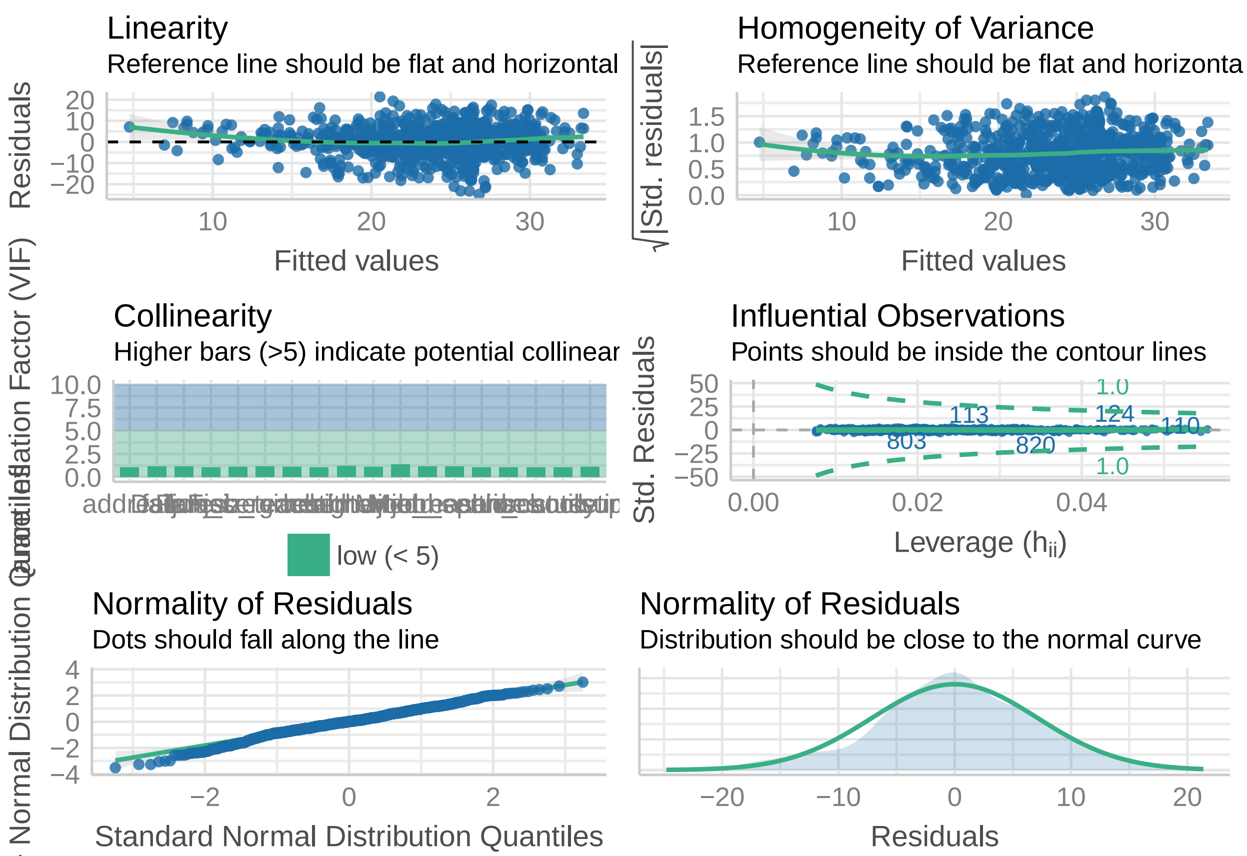
1.3.3 Stepwise Selection

The following model automatically select the best variables performing both a forward and a backward stepwise selection. We can in fact observe how the number of variables drastically decreased and that now they appear to be almost all significant (standard errors are, again, computed robustly).

```
##
## t test of coefficients:
##
```

```
##                Estimate Std. Error t value  Pr(>|t|)
## (Intercept)    21.22666    1.55815 13.6230 < 2.2e-16 ***
## Fjob_services -1.09354    0.55776 -1.9606 0.0502650 .
## Fjob_teacher   3.56986    1.26592  2.8200 0.0049189 **
## Mjob_health    2.83937    0.96060  2.9558 0.0032080 **
## Mjob_services  1.40530    0.65570  2.1432 0.0323914 *
## reason_course -0.82969    0.52891 -1.5687 0.1171087
## address        1.06036    0.61197  1.7327 0.0835257 .
## famsize        1.23151    0.53703  2.2932 0.0220917 *
## Medu           0.66178    0.25846  2.5605 0.0106311 *
## studytime      1.27785    0.30650  4.1692 3.382e-05 ***
## failures      -3.66311    0.39449 -9.2856 < 2.2e-16 ***
## schoolsup     -3.70282    0.72816 -5.0852 4.556e-07 ***
## paid          -1.98280    0.59374 -3.3395 0.0008773 ***
## higher         3.42745    0.89262  3.8397 0.0001327 ***
## internet       1.12804    0.67947  1.6602 0.0972619 .
## romantic      -1.22986    0.51724 -2.3777 0.0176485 *
## goout         -0.64756    0.23483 -2.7576 0.0059534 **
## Dalc          -0.42615    0.26283 -1.6214 0.1053204
## health        -0.62384    0.17342 -3.5973 0.0003408 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The only few noticeable differences concern the address var that is now significant being positive for kid kids who lives in urban areas. Mother education level is now positive and significant and so is having acces to internet at home. Everything else is pretty much the same.



```
## Warning: Non-normality of residuals detected (p < .001).
```

```
## Warning: Heteroscedasticity (non-constant error variance) detected (p = 0.036).
```

```
## OK: Residuals appear to be independent and not autocorrelated (p = 0.340).
```

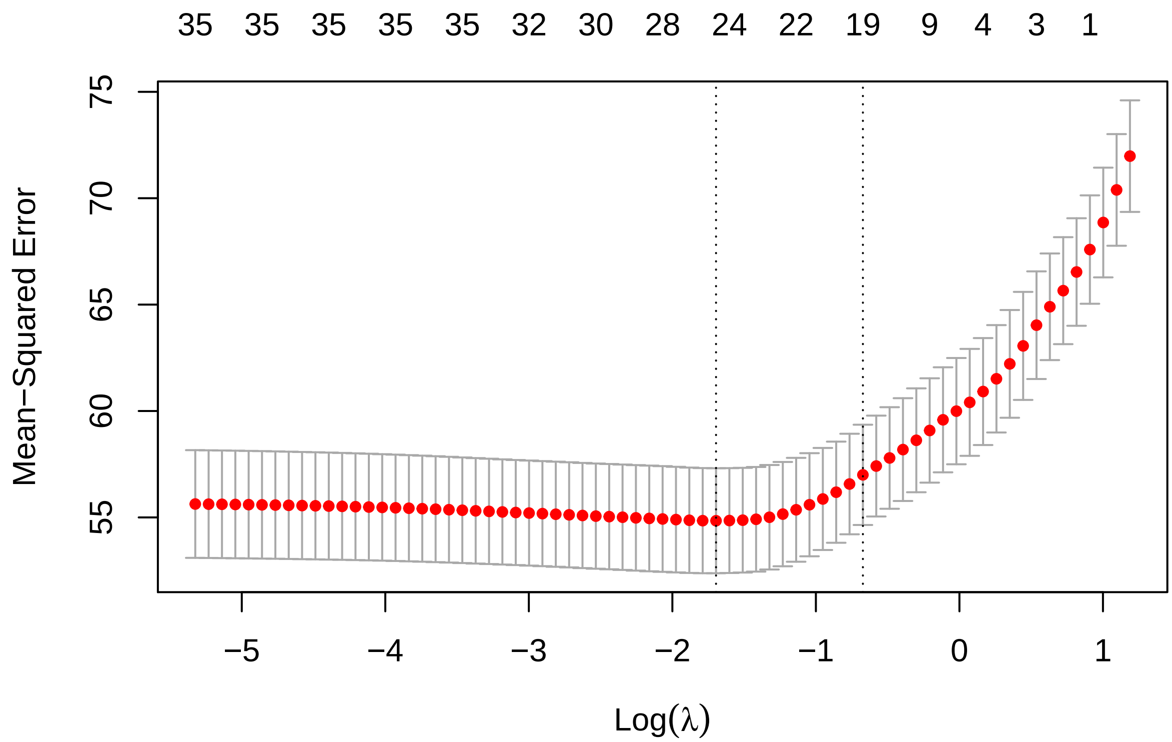
```
## # Check for Multicollinearity
##
## Low Correlation
##
##            Term  VIF Increased SE Tolerance
##   Fjob_services 1.09         1.04      0.92
##    Fjob_teacher 1.15         1.07      0.87
##     Mjob_health 1.17         1.08      0.85
##   Mjob_services 1.14         1.07      0.88
##   reason_course 1.06         1.03      0.94
##         address 1.07         1.04      0.93
##         famsize 1.03         1.01      0.97
##            Medu 1.36         1.16      0.74
##       studytime 1.09         1.04      0.92
##        failures 1.16         1.08      0.86
##       schoolsup 1.04         1.02      0.96
##            paid 1.07         1.03      0.94
##          higher 1.19         1.09      0.84
##        internet 1.12         1.06      0.89
##        romantic 1.04         1.02      0.97
##           goout 1.10         1.05      0.91
##            Dalc 1.16         1.08      0.86
##          health 1.04         1.02      0.96
```

let's always check the model assumption. As expected, same problems as before. Same solutions.

1.3.4 LASSO

Let's go forward to a more sophisticated method to select variables, the LASSO.

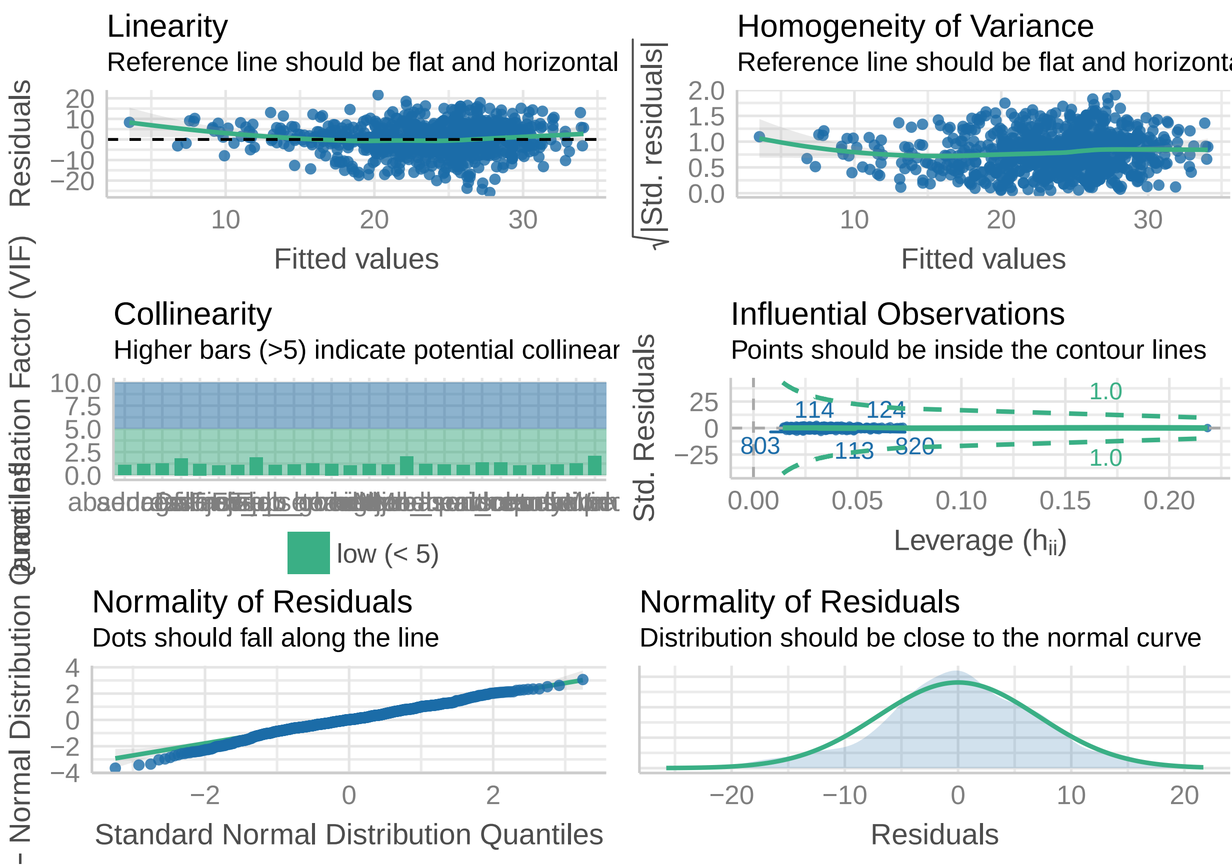
Here we can see the plot of the minimum $\log(\lambda)$ selected through cross validation on the training set.



Below we spot instead the regressors selected with that particular λ. Note that the regressors are more than the stepwise selected.

```
##  [1] "Fjob_at_home"      "Fjob_services"     "Fjob_teacher"
##  [4] "Mjob_health"       "Mjob_services"     "reason_course"
##  [7] "reason_reputation" "age"               "address"
## [10] "famsize"           "Medu"              "Fedu"
## [13] "traveltime"        "studytime"         "failures"
## [16] "schoolsup"         "famsup"            "paid"
## [19] "higher"            "internet"          "romantic"
## [22] "goout"             "Dalc"              "Walc"
## [25] "health"            "absences"
```

Let's indeed recheck the model as always.




```
## Warning: Non-normality of residuals detected (p < .001).

## Warning: Heteroscedasticity (non-constant error variance) detected (p = 0.021).

## OK: Residuals appear to be independent and not autocorrelated (p = 0.338).

## # Check for Multicollinearity
##
## Low Correlation
##
##                Term  VIF Increased SE Tolerance
##         Fjob_at_home 1.10         1.05      0.91
##        Fjob_services 1.14         1.07      0.88
##         Fjob_teacher 1.25         1.12      0.80
##          Mjob_health 1.20         1.10      0.83
##        Mjob_services 1.15         1.07      0.87
##        reason_course 1.37         1.17      0.73
##    reason_reputation 1.39         1.18      0.72
##                  age 1.25         1.12      0.80
##              address 1.21         1.10      0.83
##              famsize 1.04         1.02      0.96
##                 Medu 2.03         1.42      0.49
##                 Fedu 1.93         1.39      0.52
##           traveltime 1.25         1.12      0.80
##            studytime 1.17         1.08      0.86
##             failures 1.24         1.11      0.80
##            schoolsup 1.10         1.05      0.91
##               famsup 1.12         1.06      0.90
```

```
##             paid 1.10         1.05      0.91
##           higher 1.21         1.10      0.83
##         internet 1.14         1.07      0.88
##         romantic 1.07         1.03      0.93
##            goout 1.25         1.12      0.80
##             Dalc 1.83         1.35      0.55
##             Walc 2.09         1.45      0.48
##           health 1.07         1.04      0.93
##         absences 1.10         1.05      0.91
```

Once again let's plot the the coefficients with robust standard errors:

```
##
## t test of coefficients:
##
##                     Estimate Std. Error t value  Pr(>|t|)
## (Intercept)       17.054297   4.140410  4.1190 4.196e-05 ***
## Fjob_at_home      -0.993026   1.178029 -0.8430 0.3995023
## Fjob_services     -1.261150   0.573782 -2.1980 0.0282344 *
## Fjob_teacher       3.123946   1.268151  2.4634 0.0139700 *
## Mjob_health        2.996488   0.956581  3.1325 0.0017956 **
## Mjob_services      1.480352   0.659200  2.2457 0.0249934 *
## reason_course     -0.632362   0.579641 -1.0910 0.2756177
## reason_reputation  0.399377   0.673974  0.5926 0.5536342
## age                0.301743   0.229207  1.3165 0.1883910
## address            0.952949   0.640720  1.4873 0.1373223
## famsize            1.248596   0.548953  2.2745 0.0231964 *
## Medu               0.422144   0.328384  1.2855 0.1989790
## Fedu               0.378923   0.324795  1.1667 0.2436937
## traveltime        -0.356592   0.365791 -0.9749 0.3299252
## studytime          1.238545   0.315116  3.9304 9.204e-05 ***
## failures          -3.709631   0.409547 -9.0579 < 2.2e-16 ***
## schoolsup         -3.526902   0.761773 -4.6299 4.261e-06 ***
## famsup            -0.433821   0.530655 -0.8175 0.4138724
## paid              -1.906937   0.592955 -3.2160 0.0013515 **
## higher             3.461011   0.922747  3.7508 0.0001889 ***
## internet           1.083925   0.685671  1.5808 0.1143085
## romantic          -1.260675   0.531443 -2.3722 0.0179167 *
## goout             -0.629044   0.250675 -2.5094 0.0122878 *
## Dalc              -0.252147   0.318175 -0.7925 0.4283128
## Walc              -0.183154   0.267951 -0.6835 0.4944653
## health            -0.631318   0.175393 -3.5994 0.0003383 ***
## absences          -0.032026   0.039431 -0.8122 0.4169217
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

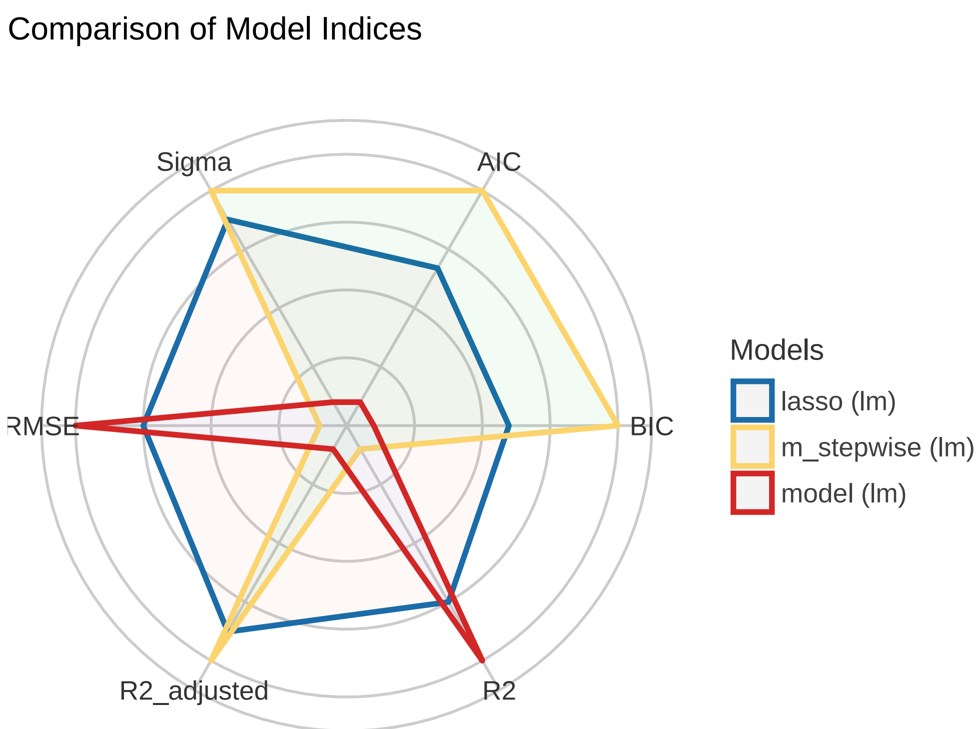
The scheme looks more like our OLS baseline model than the stepwise. I won't comment further this output since no relevant difference or surprise is spotted.

1.4 Linear Models Comparison

Now that we have assessed the three models of ours, we can proceed in comparing their performance and choose the best one. In the table below we can observe various metrics to compare their performances. There is also a "Performance Score" which ranges from 0% to 100%. Higher values indicating better model performance. Note that all score value do not necessarily sum up to 100%. Rather, calculation is based on normalizing all indices (i.e. rescaling them to a range from 0 to 1), and taking the mean value of all indices for each model. This is a rather quick heuristic, but might be helpful as exploratory index.

```
## # Comparison of Model Performance Indices
##
## Name       | Model |      AIC |      BIC |    R2 | R2 (adj.) |  RMSE | Sigma | Performance-Score
## ------------------------------------------------------------------------------------------------
## lasso      |    lm | 5709.640 | 5842.075 | 0.302 |     0.280 | 7.088 | 7.205 |            72.63%
## m_stepwise |    lm | 5700.819 | 5795.415 | 0.296 |     0.281 | 7.118 | 7.201 |            66.67%
## model      |    lm | 5724.869 | 5899.873 | 0.305 |     0.274 | 7.076 | 7.233 |            33.33%
```

From the table above we can say that the three models performs very similarly but our PerformanceScore still indicates a clear ranking between 'em, positioning the LASSO on the podium, followed by the stepwise selection and lastly by the basic OLS. We can also visualize this in the spider web below.



1.4.1 Robust Model

Now that we have assessed the best model, we can proceed trying to elaborate a Robust Linear Regression (for outliers) on the LASSO model and finally compare them. We will use Bisquare weights instead of the standard Hubers one because they're more penalizing for large outliers, as it seemed we had.

Below we have the weight our robust regression assigned to the observations. Since the weights range from [0,1], we can see how the first 10 most penalized observations were considered severe outliers from the algorithm.

```
##          resid     weight
## 999  -27.03427 0.02618108
## 333  -25.00707 0.07996344
## 243  -24.36786 0.10174085
## 1022 -23.52007 0.13368074
## 135  -21.95472 0.20003687
## 136  -21.79242 0.20734251
## 945   21.42593 0.22416459
## 334  -20.13155 0.28641421
## 388  -19.65853 0.31001753
## 149  -19.38907 0.32364354
```

1.4.2 Findings

Finally we compare all the models with all the coefficient side by side (all the SE are computed robustly).

	Baseline OLS (1)	Stepwise (2)	Lasso (3)	Robust (4)
Fjob_at_home	-1.12 (1.19)		-1.04 (1.17)	-1.29 (1.18)
Fjob_health	-0.51 (1.34)			
Fjob_services	-1.18** (0.60)	-1.09** (0.56)	-1.21** (0.58)	-1.39** (0.55)
Fjob_teacher	3.24** (1.33)	3.57*** (1.27)	3.11** (1.27)	3.74*** (1.09)
Mjob_at_home	0.21 (0.76)			
Mjob_health	2.91*** (1.06)	2.84*** (0.96)	3.04*** (0.96)	2.69*** (1.00)
Mjob_services	1.34* (0.74)	1.41** (0.66)	1.48** (0.66)	1.44** (0.59)
Mjob_teacher	-0.73 (1.07)			
reason_course	-0.44 (0.89)	-0.83 (0.53)	-0.67 (0.58)	-0.76 (0.55)
reason_home	0.35 (0.92)			
reason_reputation	0.60 (0.96)		0.38 (0.68)	0.34 (0.65)
sex	-0.25 (0.59)			
age	0.30 (0.23)		0.32 (0.23)	0.33 (0.21)
address	0.98 (0.65)	1.06* (0.61)	0.97 (0.64)	0.79 (0.58)
famsize	1.30** (0.57)	1.23** (0.54)	1.25** (0.55)	1.11** (0.51)
Pstatus	-0.13 (0.88)			
Medu	0.56 (0.37)	0.66** (0.26)	0.42 (0.33)	0.46 (0.30)
Fedu	0.38 (0.34)		0.38 (0.33)	0.50 (0.31)

traveltime	-0.33		-0.34	-0.33
	(0.37)		(0.36)	(0.33)
studytime	1.24***	1.28***	1.26***	1.24***
	(0.32)	(0.31)	(0.31)	(0.31)
failures	-3.75***	-3.66***	-3.76***	-3.56***
	(0.42)	(0.39)	(0.41)	(0.40)
schoolsup	-3.58***	-3.70***	-3.52***	-3.82***
	(0.77)	(0.73)	(0.76)	(0.71)
famsup	-0.45		-0.42	-0.59
	(0.54)		(0.53)	(0.50)
paid	-1.87***	-1.98***	-1.88***	-2.08***
	(0.60)	(0.59)	(0.59)	(0.58)
activities	0.10			
	(0.53)			
higher	3.45***	3.43***	3.49***	3.11***
	(0.93)	(0.89)	(0.92)	(0.80)
internet	1.10	1.13*	1.05	1.28*
	(0.69)	(0.68)	(0.68)	(0.66)
romantic	-1.32**	-1.23**	-1.24**	-0.81*
	(0.55)	(0.52)	(0.53)	(0.49)
famrel	0.07			
	(0.27)			
freetime	0.33		0.31	0.20
	(0.29)		(0.28)	(0.28)
goout	-0.75***	-0.65***	-0.72***	-0.65***
	(0.27)	(0.23)	(0.27)	(0.24)
Dalc	-0.26	-0.43	-0.28	-0.20
	(0.32)	(0.26)	(0.32)	(0.28)
Walc	-0.13		-0.17	-0.35
	(0.28)		(0.27)	(0.24)
health	-0.63***	-0.62***	-0.65***	-0.53***
	(0.18)	(0.17)	(0.18)	(0.17)
absences	-0.03		-0.03	-0.07**
	(0.04)		(0.04)	(0.04)
Constant	15.57***	21.23***	16.03***	16.37***
	(4.38)	(1.56)	(4.19)	(3.89)
Observations	837	837	837	837
R2	0.30	0.30	0.30	
Adjusted R2	0.27	0.28	0.28	
Residual Std. Error	7.23 (df = 801)	7.20 (df = 818)	7.20 (df = 809)	6.33 (df = 809)
F Statistic	10.02*** (df = 35; 801)	19.13*** (df = 18; 818)	13.05*** (df = 27; 809)	

From the table above we can learn some few takeaways. - The four models' performances do not differ too much altough our model performances comparison above clearly chose the LASSO as the best one. - There seems to be a certain unanimity among the models for what variables are most significant in explain the variance in Students' performances. - Comparing the Robust Regression with its regular counterpart, the LASSO, we can spot the difference in the coefficients' estimates. Although we used the most penalizing weights, those differences between coefficients are way smaller than I expected.

We can finally affirm the following for what it concerns our inference process: The parents' occupation is highly significant in explaining the variability of the student performance. Particularly, teachers fathers and mothers in health and services have a very positive impact on performance. Fathers in services have a negative effect instead. Also having siblings has a positive and unanimously significant effect on performance. Study time, failures and School supports as well as Paid are three of the always significant variables that help us explain well the variability of our model but we've already talked about them. Willingness to continue studiesis, as previusly seen, always significant and very positive although the robust coefficient is significantlly lower than the others. Same thing for being in a romantic relation and going out a lot. Having internet at home appear significant only for the stepwise and the robust so we give it for good. Again Health is negative and significant troughout all the models but with a smaller coefficient in the robust one. Finally, "absences" appear to be negative and significant only for the robust model but with a very low effect on the performance.

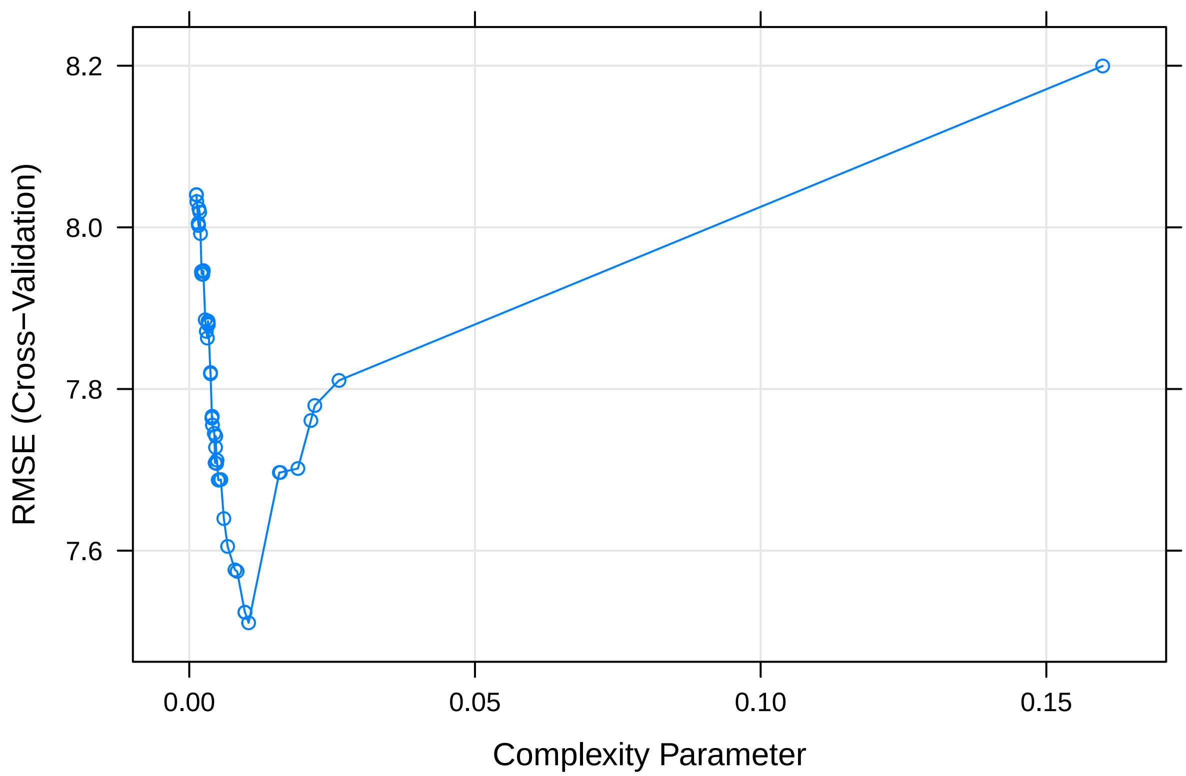
1.5 Beyond Linear Regression

Now that we are done with the inference effort, we are going to move toward models with less interpretability power but with more predictive one. In this section we're going to explore new alternatives that may capture in our data also non-linearity and interactions between our variables and then compare their performance:

We're going to use a Decision Tree and finally a Random Forests.

1.5.1 Decision Tree

We proceed with a Decision Tree, trained on train set on which a cross validation with 10 folds is applied to tune the complexity parameter.




```
## [1] 0.01039
```

We further proceed with a complexity parameter of 0.01039.

```
## [1] "Training RMSE: 7.0892287759461  Test RMSE: 7.32748115832476"
```

We then compute the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE) both for the training set and the test test. The relevant ones are of course only the test ones but in doing so, we can make sure that our model did not underfit/overfit.

MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. It's the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight.

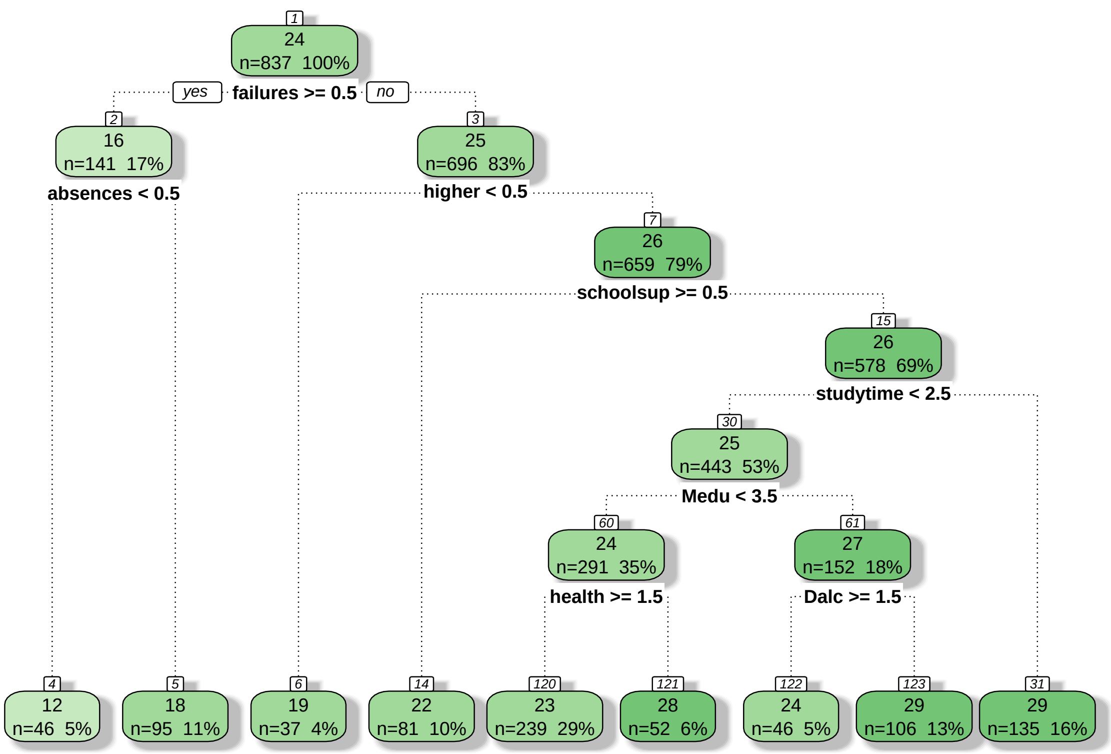
The RMSE is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit.

Taking the square root of the average squared errors has some interesting implications for RMSE. Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means the RMSE should be more useful when large errors are particularly undesirable.

```
## Loading required package: bitops

## Registered S3 method overwritten by 'rattle':
##   method         from
##   predict.kmeans parameters

## Rattle: A free graphical interface for data science with R.
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```



Rattle 2021−Jul−06 22:29:06 hainex

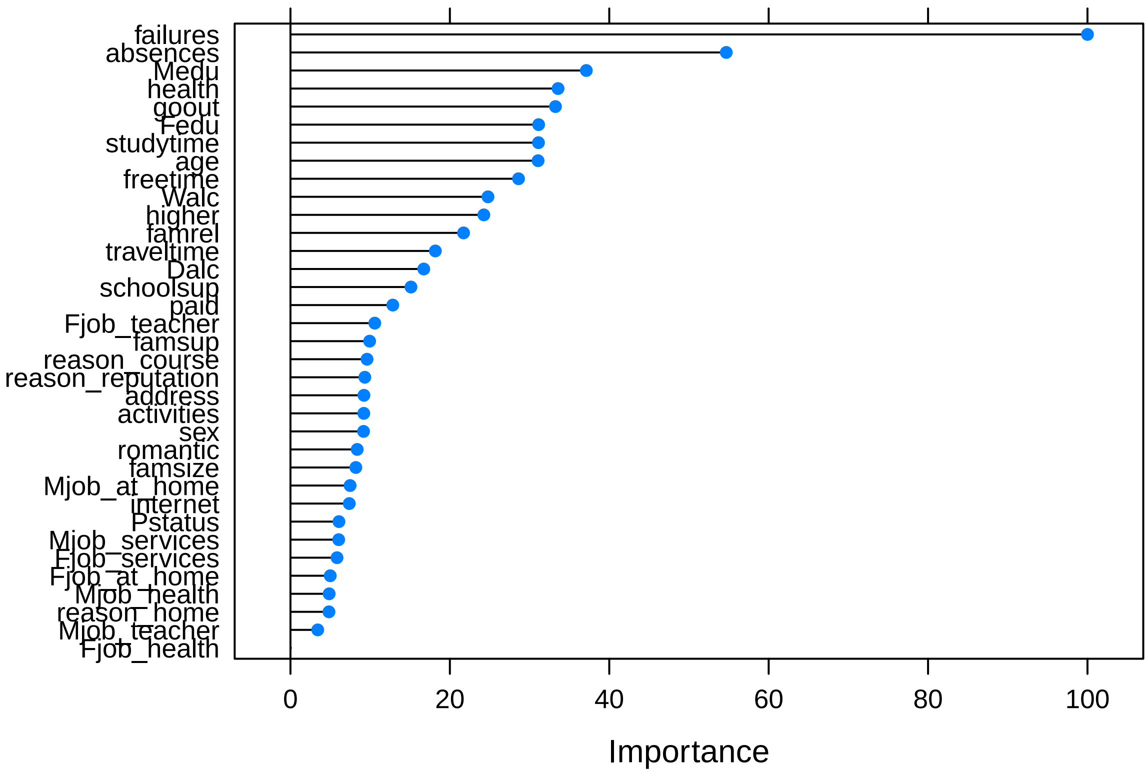
- Interestingly enough, we can spot that "Failures" is consider the best discriminant variables, followed by absences - which, I recall, was found significant only in the final robust model - and the willingness to pursue higher studies. - Other interesting point is that it did not included any of the parent's occuppation but only Mother's education and Daily Alchol consume. Two variables never significant in our previous models.

1.5.2 Random Forest

Finally we are going to dive into a random forest algorithm. The RF is trained always on training data, and it makes use of 50 Bootstraps to select the correct number of variables to include for each tree. We created a grid with 9 values with mean $\sqrt(nvars)$ and standard deviation 3.5 to choose from.

```
## [1] "Training RMSE: 3.45355346224596  Test RMSE: 6.67012787805937"
```

Altought we can't visualize a the Random Forest *per se*, we can understand the variables importance that occurred in contributing to the model building. In the image below we can in fact clearly see how `failures` is the most relevant variable by far. It is followed by `absences` which, again, it was barely significant in the last robust linear model and, at best, with a very low effect. This leaves us something to think about. Tree Based models, as stated above can better perceive non linear effects and interactions between variables. That may be the case. `Mother's education level` is again very important as well as time spent `going out` and `study time` and, unexpectedly, `Weekend alcholic use`. Some of the variables that we treated as very relevant like `higher` or the `Parents' occupations` are instead down below the rank.



Let's finally produce a table to compare all the models

```
##            OLS  Stepwise    LASSO    Robust      TREE        RF
## RMSE 7.6269968 7.5217937 7.577527 7.5577834 7.3274812 6.6701279
## MAE  6.0750000 6.0220000 6.061000 6.0240000 5.6410000 5.2170000
## R2   0.3045905 0.2962748 0.302285 0.2979893 0.3020176 0.8857574
```

1.6 Conclusions Part I

What we can learn from this final chapter of this Assignment is that Linear Models and Tree based models, as stated above, can better perceive non linear effects and interactions between variables. That, in fact, may be the case since we saw a coherence among linear models in selecting important variables and a different but also coherent fashion for what it concerns the tree based models.

The parents' occupation is highly significant in explaining the variability of the student performance. Particularly, teachers fathers and mothers in health and services have a very positive impact on performance. Also having siblings has a positive and unanimously significant effect on performance. Study time, failures and School supports as well as Paid are always significant variables that help us explain well the variability of our model. Willingness to continue studiesis, as previusly seen, is significant and very positive. Being in a romantic relation and going out a lot are negative and significant. In the tree based model instead `failures` is the most relevant variable, followed by `absences` which was barely significant in the linear models.

Talking about performances, the Linear model and Tree models do not differ that much. We have the LASSO that it's the best performative among the linear models but it's also better than the decision tree for what it concerns R2. Both Tree and the Random Forest, though, performs better in both MAE and RMSE with the Random Forest that clearly outperforms all the other models in everything. This is nothing shocking as it was largely expected.

2 PART II: UNSUPERVISED LEARNING

Now that we're done with Inference and Prediction, in this second part of this work we are going to apply some unsupervised learning techniques on our dataset to try to discover some useful insight.

The characteristic of our response variable, as stated in the first part of the assignment, let us space in deciding how to treat and proceed our data analysis. Let's recall previously that we suggested 3 alternative ways to go ahead:

1. Binary classification

   - y > 10: pass
   - y < 10: fail

2. five-level classification based on Erasmus grade conversion system

   - 16-20: very good
   - 14-15: good
   - 12-13: satisfactory
   - 10-11: sufficient
   - 0-9 : fail

3. Regression (Predicting y)

In First Assignment we proceeded with the third option because we thought more important to exploit interpretable models to make some inference.

In this second Assignment, though, it could be interesting to exploit unsupervised techniques to check whether our data alone, i.e. without none of the `grade` variables (G1,G2,G3), are capable to grasp the dimensional difference between the "groups" of students.

It would be interesting, though, if our clustering methods could, alone, be able to divide and cluster all of our students in 2 different groups, the ones who pass and the ones who fail.

To do that we need to reload our original dataset without any transformation applied and starting all over again. First, we're going to apply a K-means algorithm and then a Hierachical Clustering.

2.1 K Means

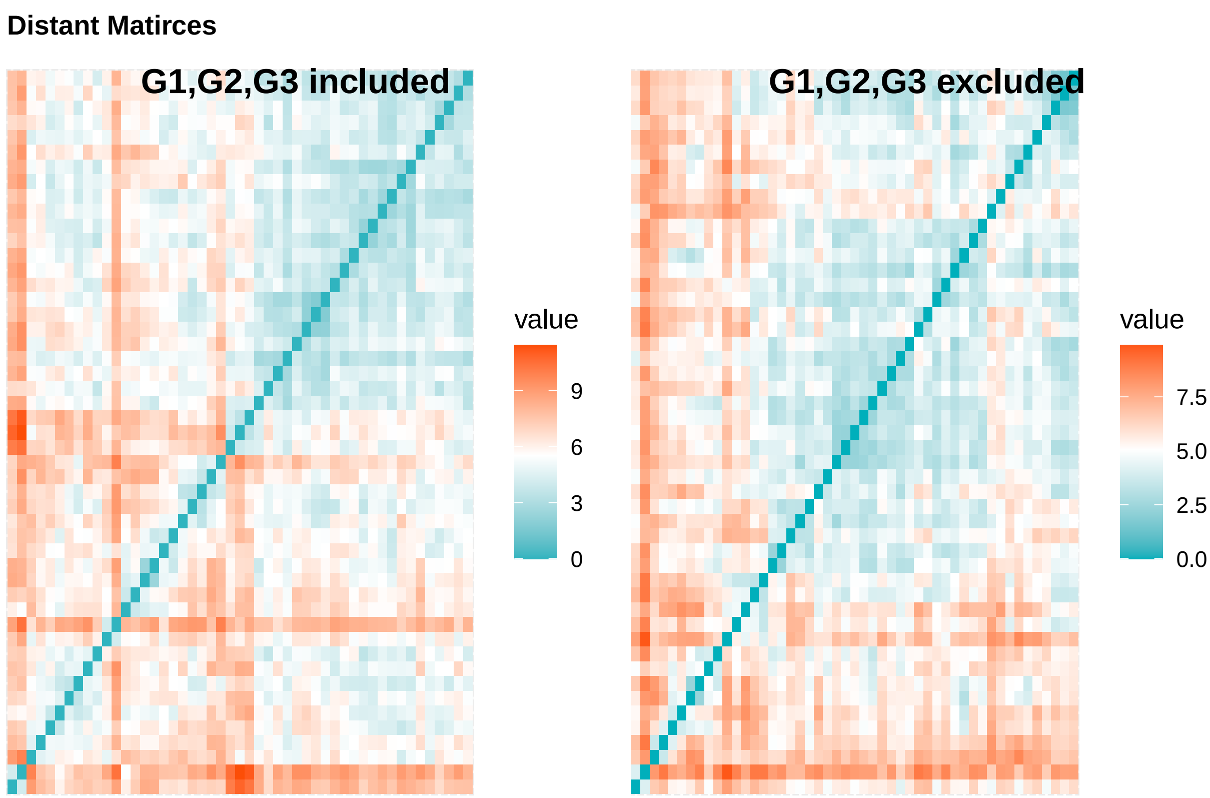
K-means clustering is the most commonly used unsupervised machine learning algorithm for partitioning a given data set into a set of k groups (i.e. k clusters), where k represents the number of groups pre-specified by the analyst. It classifies objects in multiple groups (i.e., clusters), such that objects within the same cluster are as similar as possible (i.e., high intra-class similarity), whereas objects from different clusters are as dissimilar as possible (i.e., low inter-class similarity). In k-means clustering, each cluster is represented by its center (i.e, centroid) which corresponds to the mean of points assigned to the cluster.

```
## Variable   |      Mean |   SD |  IQR |          Range | Skewness | Kurtosis |    n | n_Missing
## ------------------------------------------------------------------------------------------------
## age        |  2.32e-16 | 1.00 | 1.61 |  [-1.39, 4.25] |     0.43 |     0.04 | 1044 |         0
## Medu       | -8.30e-18 | 1.00 | 1.78 |  [-2.31, 1.24] |    -0.14 |    -1.23 | 1044 |         0
## Fedu       |  1.57e-16 | 1.00 | 1.82 |  [-2.17, 1.47] |     0.12 |    -1.17 | 1044 |         0
## traveltime |  4.58e-17 | 1.00 | 1.37 |  [-0.71, 3.39] |     1.37 |     1.48 | 1044 |         0
## studytime  | -4.60e-17 | 1.00 | 1.20 |  [-1.16, 2.43] |     0.67 | 6.62e-03 | 1044 |         0
```

```
## failures    | -9.49e-18 | 1.00 | 0.00 |  [-0.40, 4.17] |     2.78 |     7.50 | 1044 |          0
## famrel      |  1.04e-17 | 1.00 | 1.07 |  [-3.15, 1.14] |    -1.06 |     1.29 | 1044 |          0
## freetime    | -1.32e-16 | 1.00 | 0.97 |  [-2.13, 1.74] |    -0.18 |    -0.36 | 1044 |          0
## goout       |  1.05e-16 | 1.00 | 1.74 |  [-1.87, 1.60] |     0.04 |    -0.84 | 1044 |          0
## Dalc        |  8.90e-17 | 1.00 | 1.10 |  [-0.54, 3.85] |     2.16 |     4.48 | 1044 |          0
## Walc        |  1.36e-16 | 1.00 | 1.56 |  [-1.00, 2.11] |     0.63 |    -0.78 | 1044 |          0
## health      | -7.29e-19 | 1.00 | 1.40 |  [-1.79, 1.02] |    -0.50 |    -1.08 | 1044 |          0
## absences    |  3.92e-17 | 1.00 | 0.97 | [-0.71, 11.36] |     3.74 |    26.60 | 1044 |          0
```

First things first, we need to purge our dataset from categorical variables because they're not supported by K means algorithm, then we need to rescale all of our numerical variables as we've done in the table above. All the variables have 0 mean and 1 standard deviation.

After this preliminary processing, we can proceed further in capturing the (dis)similarity between the observations since the goal of clustering methods is exactly classify data samples into groups of similar objects. Through an enhanced distant matrix that use by default "euclidean distance (but other alternatives are available) we can visualize the data.



In the figure above we can see the dissimilarity matrices computed for both our dataset with our "response" variables included and excluded. Important: - The two matrices seem not to differ that much. This means that a clustering algorithm could really be able to detect the right cluster of students based on the information provided (that are not the "grades").

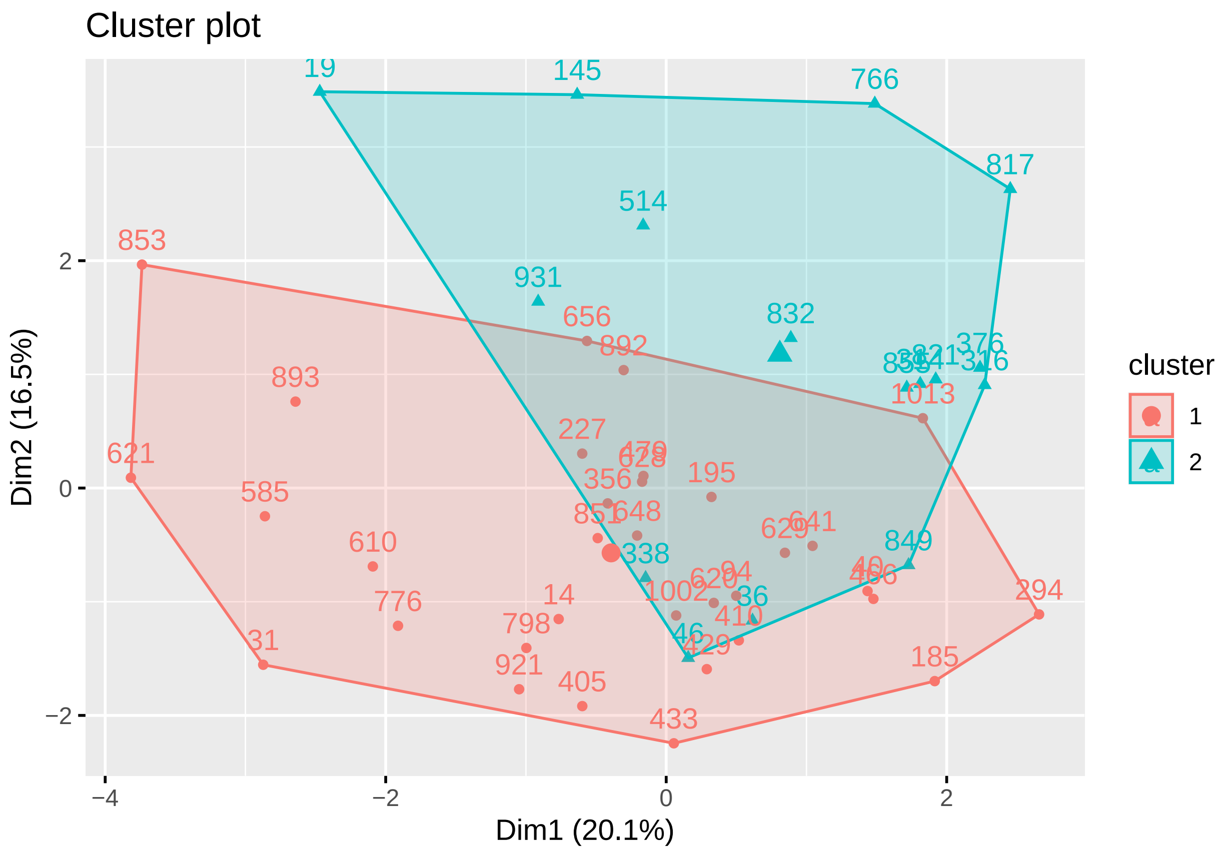
Here we will group the data into two clusters (centers = 2). The `kmeans` function also has an `nstart` option that attempts multiple initial configurations and reports on the best one. For example, adding `nstart` = 25 will generate 25 initial configurations. This approach is often recommended.

The output of k-means is a list with several bits of information. The most important being:

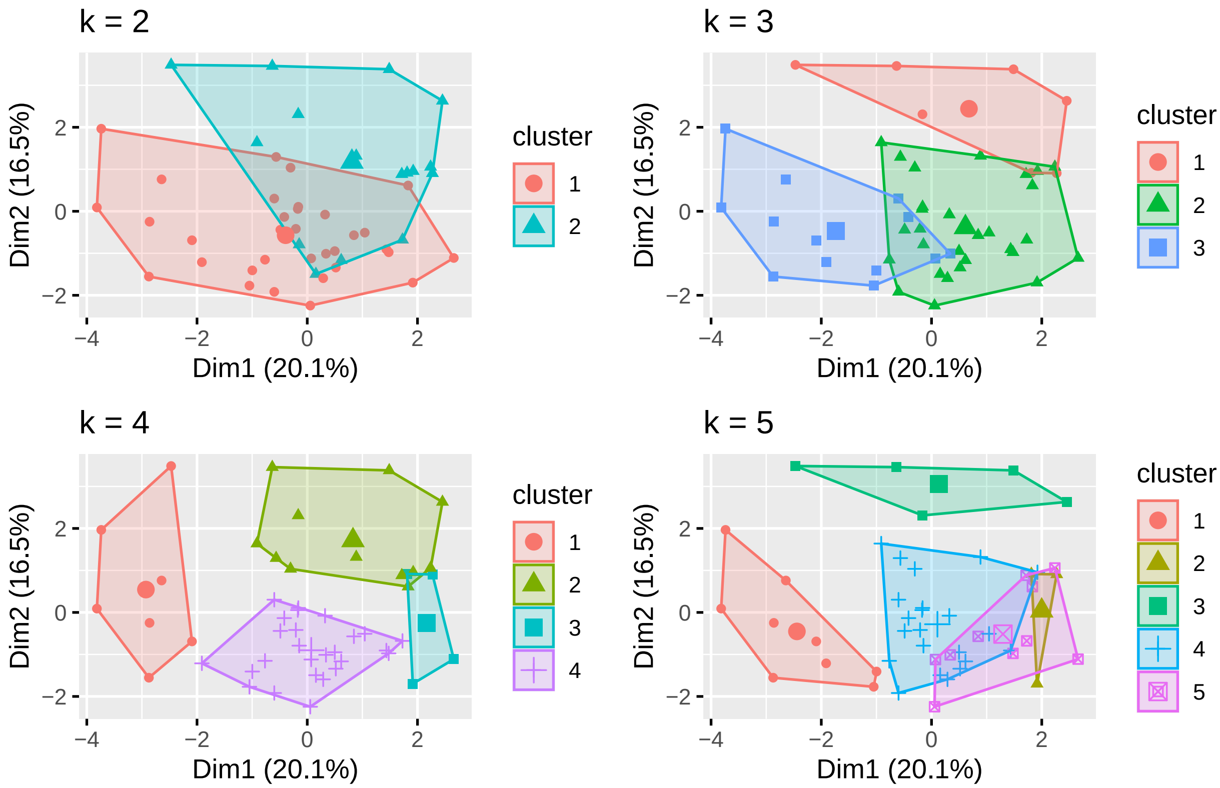
- cluster: A vector of integers (from 1:k) indicating the cluster to which each point is allocated.
- centers: A matrix of cluster centers.

- totss: The total sum of squares.
- withinss: Vector of within-cluster sum of squares, one component per cluster.
- tot.withinss: Total within-cluster sum of squares, i.e. sum(withinss).
- betweenss: The between-cluster sum of squares, i.e. $totss - tot.withinss$.
- size: The number of points in each cluster.

Th pe following image provides a nice illustration of the clusters. If there are more than two dimensions (variables) the function will automatically perform Principal Component Analysis (PCA) and plot the data points according to the first two principal components that explain the majority of the variance.



Because the number of clusters (k) must be set before we start the algorithm, it is often advantageous to use several different values of k and examine the differences in the results. We can execute the same process for 3, 4, and 5 clusters, and the results are shown in the figure:



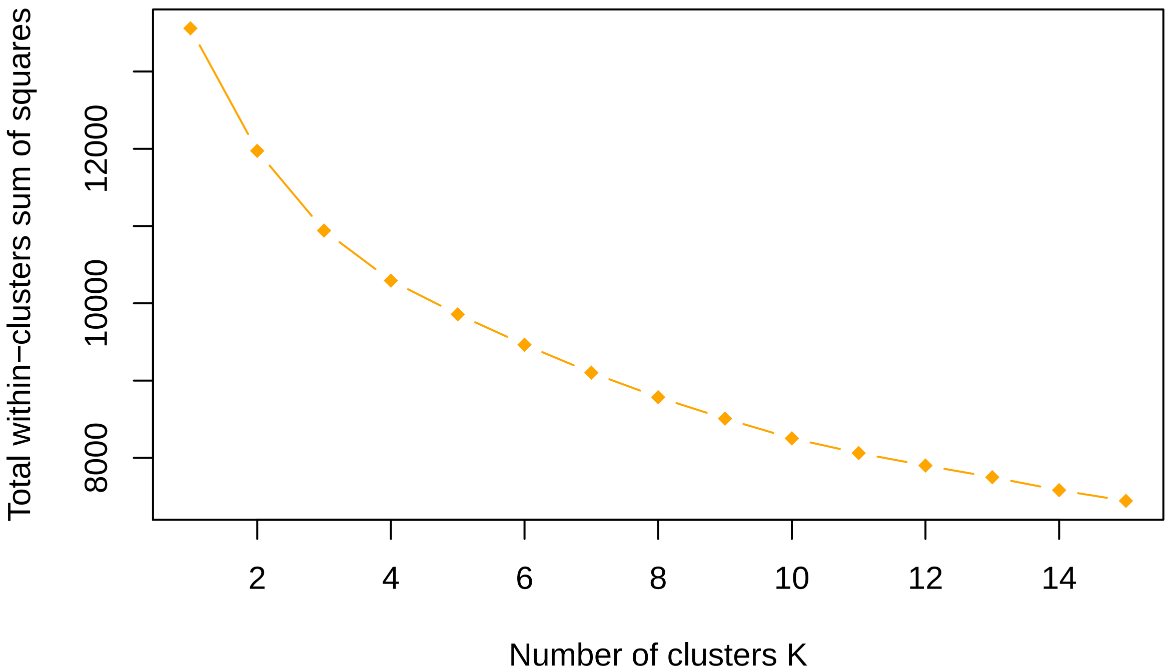
2.2 Determining Optimal Clusters

We recall that it is scientist's prerogative to specify the number of clusters to use; preferably they would like to use the optimal number of clusters. To aid in this scope, we will explore the following two popular methods for determining the optimal clusters:

- Elbow method
- Silhouette method

Recall that, the basic idea behind cluster partitioning methods, such as k-means clustering, is to define clusters such that the total intra-cluster variation (known as total within-cluster variation or total within-cluster sum of square) is minimized. The Elbow method is an heuristics consisting of plotting the explained variation as a function of the number of clusters, and picking the elbow of the curve as the number of clusters to use.

As we can see in the figure below is not always that easy as no clear "elbow" could be spotted by human sight.



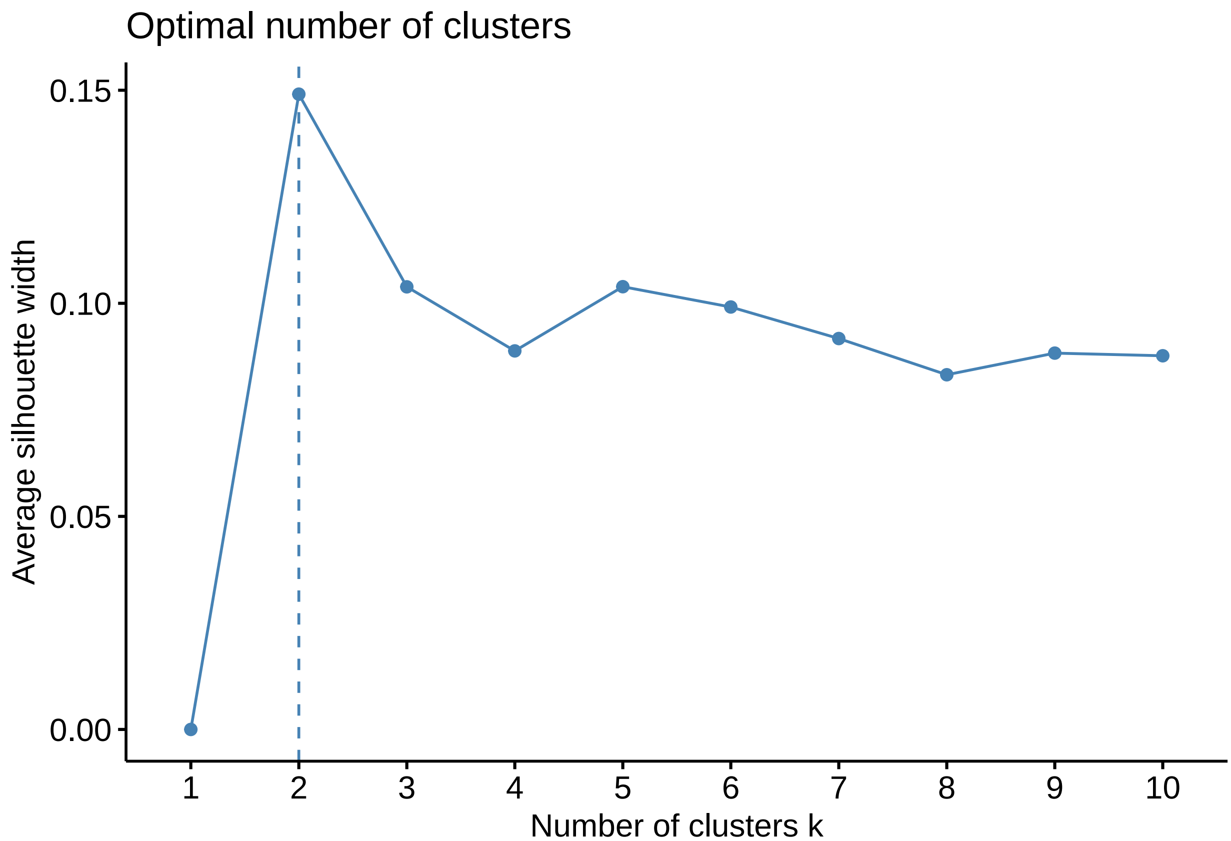
The decision of what number of cluster to choose stay unclear. Therefore we move on the next approach.

2.3 Average Silhouette Method

In short, the average silhouette approach measures the quality of a clustering. That is, it determines how well each object lies within its cluster. A high average silhouette width indicates a good clustering. The average silhouette method computes the average silhouette of observations for different values of k. The optimal number of clusters k is the one that maximizes the average silhouette over a range of possible values for k.

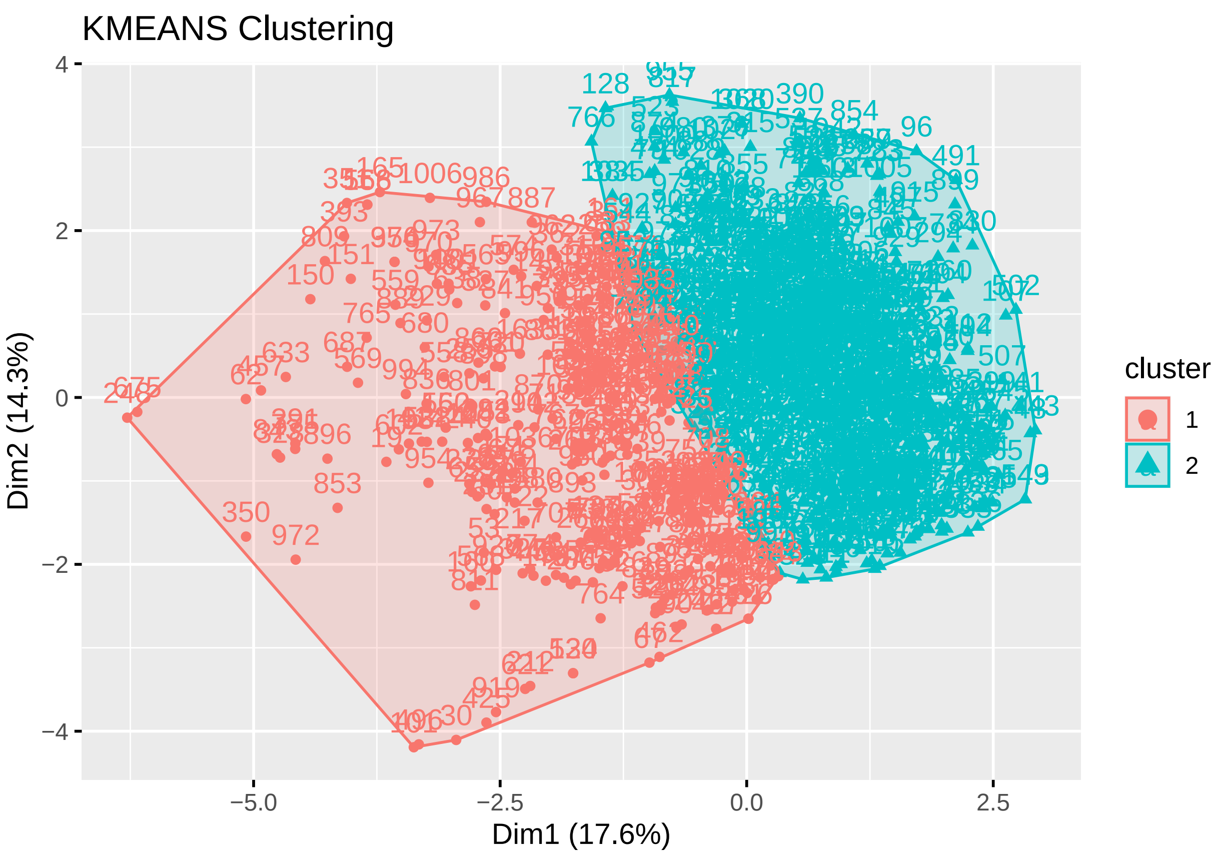
The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from -1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.

Here our silhouette values for different values of k



It's definitely easier to choose our right number of cluster (which is automatically highlighted).

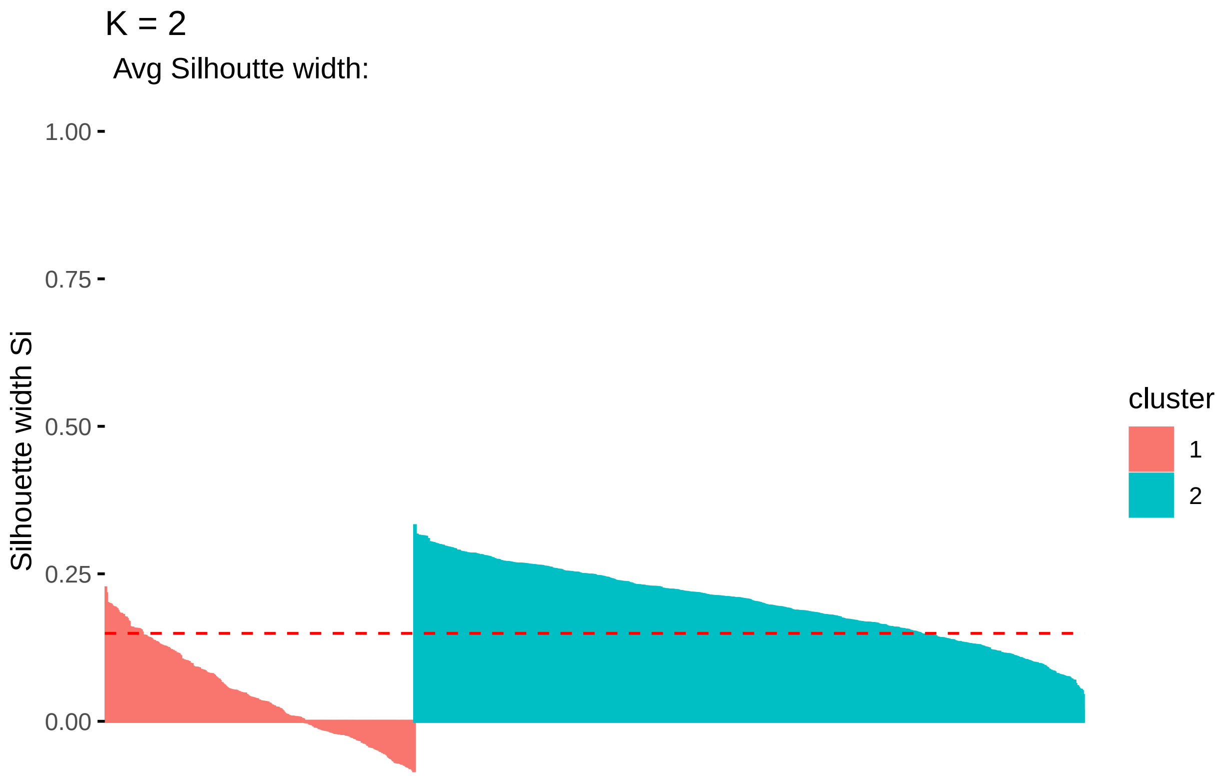
We're going to replot our graphical representation with our full dataset with 2K as suggested by the Silhouette



Method.

```
##   cluster size ave.sil.width
## 1       1  330          0.04
## 2       2  714          0.20
```

Below we can also observe a representation of the Silhouette Method itself, with the two cluster both largely above the zero value.



Finally we can summarize all our variables with in our two clusters:

```
Cluster age   Medu   Fedu traveltime studytime failures famrel freetime  goout Dalc Walc health absences
<int>  <dbl>    <dbl>    <dbl>    <dbl>  <dbl>    <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>     <dbl>
1   0.249 -0.112   -0.0255    0.241   -0.428     0.402   -0.0987  0.426  0.722  0.882  1.03   0.209     0.297
2 -0.120   0.0537   0.0123   -0.116    0.206    -0.193   0.0475  -0.205 -0.347 -0.424 -0.495 -0.100    -0.143
```

2.4 Conclusions Part II

We can easily spot that the first cluster is associated with the "Good Performance" students as we already know the effects of the single variables on the student performance. There is no single variable that is wrongly classified therefore we can successfully claim that given our dataset, the K mean algorithm - without any help from the "responses" variables - clearly defines a boundary between good and bad performative students. This is quite an achievement.

3 Appendix

```r
## ----setup, include=FALSE, echo=FALSE, warning=FALSE,error=FALSE,fig.align = "center"----------------------------------
knitr::opts_chunk$set(echo = FALSE
                      #,warning=FALSE,
                      # message=FALSE
                      )
extrafont::loadfonts()
# remotes::install_github("easystats/easystats")
library(tidyverse)
library(sandwich)
library(readr)
library(corrplot)
library(easystats)
library(hrbrthemes)
library(Hmisc)
library(GoodmanKruskal)
library(ggraph)
library(glmnet)
library(caret)
library(ggpubr)
library(olsrr)
library(GGally)
library(mltools)
library(data.table)
library(multcomp)
library(car)
library(MASS)
library(lmtest)
library(doParallel)
source('data/funct/unregister_dopar.R')
df <- read_csv("data/student.csv")


## ----bank --------------------------------------------------------------------------------------------------------
df$sex <- as.factor(df$sex)
df$address<- as.factor(df$address)
df$famsize<- as.factor(df$famsize)
df$Pstatus <- as.factor(df$Pstatus)
df$Mjob<- as.factor(df$Mjob)
df$Fjob<- as.factor(df$Fjob)
df$reason<- as.factor(df$reason)
df$guardian<- as.factor(df$guardian)
df$schoolsup<- as.factor(df$schoolsup)
df$famsup<- as.factor(df$famsup)
df$paid<- as.factor(df$paid)
df$activities<- as.factor(df$activities)
df$nursery<- as.factor(df$nursery)
df$higher <- as.factor(df$higher)
df$internet <- as.factor(df$internet)
df$romantic <- as.factor(df$romantic)
df$school <- NULL
G1 <- df$G1
G2 <- df$G2

describe_distribution(df) # numeric variables


## ----pressure, echo=F, error=FALSE, fig.align='center', warning=FALSE,fig.width=7---------------------------------------
a <- ggplot(df,
       aes(x = Fjob,
           fill = famsup)) +
  geom_bar(position = "stack")

```

```
b <- ggplot(df,
       aes(x = Mjob,
           fill = famsup)) +
  geom_bar(position = "stack")

c <- ggplot(df,
       aes(x = famsup,
           fill = paid)) +
  geom_bar(position = "fill") +
            labs(y = "Proportion")
d <- ggplot(df,
       aes(x = activities ,
           fill = address)) +
  geom_bar(position = "fill") +
            labs(y = "Proportion")
s <- ggarrange(a,b,c,d,
          labels = c("A", "B", "C","D"),
          ncol = 2, nrow = 2)

annotate_figure(s,
                top = text_grob("", color = "black", face = "bold", size = 14),
                fig.lab = "Figure 1", fig.lab.face = "bold")


## ----fig, fig.align = "center", fig.width
    =9---------------------------------------------------------------------------

annotate_figure(ggarrange(ggplot(data=df, aes(x=G3, group=Mjob, fill=Mjob)) +
    geom_density(adjust=1.5, alpha=.4) +
    theme_ipsum(base_family = 'Helvetica')
    ,ggplot(data=df, aes(x=G3, group=Fjob, fill=Fjob)) +
    geom_density(adjust=1.5, alpha=.4) +
    theme_ipsum(base_family = 'Helvetica')
    ,ggplot(data=df, aes(x=G3, group=address, fill=address)) +
    geom_density(adjust=1.5, alpha=.4) +
    theme_ipsum(base_family = 'Helvetica')
    ,ggplot(data=df, aes(x=G3, group=internet, fill=internet)) +
    geom_density(adjust=1.5, alpha=.4) +
    theme_ipsum(base_family = 'Helvetica'),
    labels = c("A", "B", "C","D")),
                top = text_grob(" ", color = "black", face = "bold", size = 14),
                fig.lab = "Figure 2", fig.lab.face = "bold")


## ----fig.align='center
    '--------------------------------------------------------------------------------------------------

ggplot(data = df) +
  geom_count(mapping = aes(x = G3, y = failures))+
  theme_ipsum(base_family = 'Helvetica')


##
    ----------------------------------------------------------------------------------------------------

df <- df %>%
            mutate(y = round((G1+G2+G3)/3,1), .keep = 'unused')

BoxCoxTrans(df$y)
df$y <- df$y^1.3


## ----fig.align='center
    '--------------------------------------------------------------------------------------------------

ggdensity(df, x = "y", fill = "lightblue", title = "General Grade") +
  stat_overlay_normal_density(color = "red", linetype = "dashed")
```

```r

## ----------------------------------------------------------------------------------------------------
leveneTest(y~Mjob,data = df)


## ----warning=FALSE,echo=FALSE,message=FALSE ------------------------------------------------------------------------------
mjob <- aov(y~Mjob,data = df)
posthoc = glht(mjob, linfct = mcp(Mjob = "Tukey"))
summary(corrected <- posthoc, test = adjusted(type = "bonferroni"))


## ----fig.width =4----------------------------------------------------------------------------------------------------
leveneTest(y~Fjob,data = df)
fjob <- aov(y~Fjob,data = df)
posthoc2 = glht(fjob, linfct = mcp(Fjob = "Tukey"))
summary(corr <- posthoc2,test = adjusted(type = "bonferroni"))


## ----------------------------------------------------------------------------------------------------
reas <- aov(y~reason,data = df)
posthoc3 = glht(reas, linfct = mcp(reason = "Tukey"))
summary(posthoc3,test = adjusted(type = "bonferroni"))

inter <- aov(y~internet,data = df)
summary(inter)

addr <- aov(y~address,data = df)
summary(addr)



## ----include=FALSE ----------------------------------------------------------------------------------------------------
corrplo <- df %>%
  correlation() %>%
  summary()

library(Hmisc)
library(corrplot)
flattenCorrMatrix <- function(cormat, pmat) {
  ut <- upper.tri(cormat)
  data.frame(
    row = rownames(cormat)[row(cormat)[ut]],
    column = rownames(cormat)[col(cormat)[ut]],
    cor  =(cormat)[ut],
    p = pmat[ut]
    )
}
r <- append (corrplo$Parameter, 'y')
r <- r[-c(11)]

res2<-rcorr(as.matrix(df[r]))
flattenCorrMatrix(res2$r, res2$P)


## ----fig.align='center '----------------------------------------------------------------------------------------------
# Insignificant correlations are leaved blank
corrplot::corrplot(res2$r, type="upper",
```

```r
        p.mat = res2$P, insig = "blank",diag = F, tl.col = 'black')


## ----fig.align='center
    '-------------------------------------------------------------------------------------------

df[r] %>%
  correlation(partial = T) %>%
  plot()


##
    -------------------------------------------------------------------------------------------------------

multiple = c("Fjob", "Mjob", "reason")

binary = c("sex",'address','higher',"famsize", "Pstatus", "schoolsup", "famsup", "paid", "
    activities", "internet", "romantic")

for (col in binary) {
  df[col] <- as.numeric(unlist(df[col]))
}
df[binary] <- ifelse(df[binary] == 1,0,1)

one_hot_enc <- as.data.frame(one_hot(as.data.table(df[multiple])))

enc_df <- df[,!(names(df) %in% multiple)]
enc_df <- enc_df[,!(names(enc_df) %in% c('guardian','nursery'))]

one_hot_enc <- one_hot_enc[,!(names(one_hot_enc) %in% c('Fjob_other','Mjob_other','reason_
    other'))] # drop the baselines cat, the ones less interpretable
one_hot_enc_alter <- one_hot_enc[,!(names(one_hot_enc) %in% c('Fjob_health','Fjob_services',
    'Fjob_at_home','Mjob_at_home','Mjob_services','reason_course','reason_home'))] # drop
    the baselines cat, the ones less interpretable

data <- cbind(one_hot_enc,enc_df)
data_altern <- cbind(one_hot_enc_alter,enc_df)
data_altern$Walc <- NULL
describe_distribution(data)


## ----traintest
    -------------------------------------------------------------------------------------------------------

set.seed(30)
split_train_test <- createDataPartition(y = data$y, p=0.8, list = F)
train <- data[split_train_test,]
test <-  data[-split_train_test,]
dim(test)
dim(train)


## ----fig.align='center
    '-------------------------------------------------------------------------------------------

model <- lm(y ~., train)
check_model(model)


##
    -------------------------------------------------------------------------------------------------------

check_normality(model)
check_heteroscedasticity(model)
check_autocorrelation(model)
check_collinearity(model)
```

```r
## ------------------------------------------------------------------------------------------------
plot(model, which = 4)


## ------------------------------------------------------------------------------------------------
lmtest::coeftest(model, vcov. = vcovHC, type = "HC1")


## ------------------------------------------------------------------------------------------------
performance(model)


## ------------------------------------------------------------------------------------------------
m_stepwise <- lm(y ~., train)
m_stepwise <- select_parameters(m_stepwise)
coeftest(m_stepwise,vcov. = vcovHC, type = "HC1")


## ------------------------------------------------------------------------------------------------
check_model(m_stepwise)


## ------------------------------------------------------------------------------------------------
check_normality(m_stepwise)
check_heteroscedasticity(m_stepwise)
check_autocorrelation(m_stepwise)
check_collinearity(m_stepwise)


## ------------------------------------------------------------------------------------------------
x = as.matrix(train[,-36])
y = train$y


## ------------------------------------------------------------------------------------------------
m_cvlasso=cv.glmnet(x,y)
plot(m_cvlasso)


## ------------------------------------------------------------------------------------------------
coef <- coef(m_cvlasso, s = m_cvlasso$lambda.min)
coefname <- coef@Dimnames[[1]][-1]
coef <- coefname[coef@i]
coef


## ------------------------------------------------------------------------------------------------
fmla <- as.formula(paste("y ~ ", paste(coef, collapse = "+")))
```

```r
lasso <- lm(fmla, data=train)
check_model(lasso)


## ----------------------------------------------------------------------------------------------------------
check_normality(lasso)
check_heteroscedasticity(lasso)
check_autocorrelation(lasso)
check_collinearity(lasso)


## ----------------------------------------------------------------------------------------------------------

coeftest(lasso, vcov. = vcovHC, type='HC1')


## ----------------------------------------------------------------------------------------------------------
compare_performance(model,lasso,m_stepwise,rank = T)


## ---- fig.align='center'
    '------------------------------------------------------------------------------------------------
plot(compare_performance(model,lasso,m_stepwise,rank = T))


## ----------------------------------------------------------------------------------------------------------
robust <- rlm(fmla,data=train, psi = psi.bisquare) # more penalizing than standard huber one


## ----------------------------------------------------------------------------------------------------------
hweights <- data.frame(resid = robust$resid, weight = robust$w)
hweights2 <- hweights[order(robust$w),]
hweights2[1:10,]


## ----Table comparison, error=FALSE, message=FALSE, warning=FALSE, paged.print=TRUE
    -------------------------------------
rob_se_pan <- list(sqrt(diag(vcovHC(model, type = "HC1"))),
                   sqrt(diag(vcovHC(m_stepwise, type = "HC1"))),
                   sqrt(diag(vcovHC(lasso, type = "HC1"))),
                   sqrt(diag(vcovHC(robust, type = "HC1")))
                   )

# stargazer::stargazer(model, m_stepwise, lasso, robust,
#                       type = 'text',
#                       digits = 2,
#                       dep.var.labels.include = F,
#                       omit.table.layout = "n",
#                       header = F,
#                       column.labels = c('Baseline OLS', 'Stepwise', 'Lasso', "Robust"),
#                       se = rob_se_pan)



## ----fig.align='center
    '------------------------------------------------------------------------------------------------
```

```r

cntr <- caret::trainControl(method = 'cv',
                            number = 10,
                            search = 'grid')

#### Comment the code to obtain the algo just because I saved it as an Rds file and load it automatically,
#### it's faster this way ####

# tree <- caret::train(y~.,
#                       data = train,
#                       method = "rpart",
#                       trControl = cntr,
#                       tuneLength = 50)

#saveRDS(tree, "rpar_model.rds")

tree <- readRDS("rpar_model.rds")
plot(tree)
print(round(tree$bestTune[[1]],5))


## ------------------------------------------------------------------------------------------------------------
train_pred = predict(tree, newdata = train)
test_pred = predict(tree, newdata = test)

print(paste0('Training RMSE: ', (rmse(train$y,train_pred)),'  ',
             'Test RMSE: ', (rmse(test$y, test_pred)),'  ',
             'Training MAE: ', (mae(train$y,train_pred)),'  ',
             'Test MAE: ', (mae(test$y, test_pred))
             )
      )

TREE <- c(rmse(test$y, test_pred),mae(test$y, test_pred),R2(train$y, train_pred))


## ------------------------------------------------------------------------------------------------------------
library(rattle)

fancyRpartPlot(tree$finalModel)


## ---- warning=FALSE, error=FALSE ------------------------------------------------------------------------

# cl <- makePSOCKcluster(7)
# registerDoParallel(cl)


cntr <- trainControl(method = 'boot_all',
                              number = 50)

tunegrid <- expand.grid(.mtry=rnorm(9,mean=sqrt(length(train))+3,sd=3.5))

#### Comment the code to obtain the algo just because I saved it as an Rds file and load it automatically,
#### it's faster this way ####

# rf <- caret::train(y ~ .,
#                         data = train,
#                         method = "rf",
#                         trControl = cntr,
#                         tuneGrid= tunegrid,
#                         allowParallel = T)
```

```
# saveRDS(rf, "rf_model.rds")

rf <- readRDS("rf_model.rds")

#stopCluster(cl)


## ------------------------------------------------------------------------------------------------

train_pred = predict(rf, newdata = train)
test_pred = predict(rf, newdata = test)

print(paste0('Training RMSE: ', (rmse(train$y,train_pred)),'  ',
             'Test RMSE: ', (rmse(test$y, test_pred)),'  ',
             'Training MAE: ', (mae(train$y,train_pred)),'  ',
             'Test MAE: ', (mae(test$y, test_pred))
             )
      )
RF <- c(rmse(test$y, test_pred),mae(test$y, test_pred),R2(train$y, train_pred))


## ------------------------------------------------------------------------------------------------

plot(varImp(rf))


## ------------------------------------------------------------------------------------------------

test_pred = predict(model, newdata = test)
train_pred = predict(model, newdata = train)
OLS <- c(rmse(test$y, test_pred),mae(test$y, test_pred),R2(train$y, train_pred))
test_pred = predict(m_stepwise, newdata = test)
train_pred = predict(m_stepwise, newdata = train)
Stepwise <- c(rmse(test$y, test_pred),mae(test$y, test_pred),R2(train$y, train_pred))
test_pred = predict(lasso, newdata = test)
train_pred = predict(lasso, newdata = train)
LASSO <- c(rmse(test$y, test_pred),mae(test$y, test_pred),R2(train$y, train_pred))
test_pred = predict(robust, newdata = test)
train_pred = predict(robust, newdata = train)
Robust <- c(rmse(test$y, test_pred),mae(test$y, test_pred),R2(train$y, train_pred))

metrics <- data.frame(OLS,Stepwise,LASSO,Robust,TREE,RF)
row.names(metrics) <- c("RMSE","MAE","R2")
metrics


## ---- include=FALSE, message=FALSE
    ------------------------------------------------------------------------
library(grid)
library(gridExtra)
library(cluster)
library(factoextra)
library(png)
library(dendextend)

df <- read_csv("data/student.csv")
df$sex <- as.factor(df$sex)
df$address<- as.factor(df$address)
df$famsize<- as.factor(df$famsize)
df$Pstatus <- as.factor(df$Pstatus)
df$Mjob<- as.factor(df$Mjob)
df$Fjob<- as.factor(df$Fjob)
df$reason<- as.factor(df$reason)
df$guardian<- as.factor(df$guardian)
df$schoolsup<- as.factor(df$schoolsup)
```

```r
df$famsup<- as.factor(df$famsup)
df$paid<- as.factor(df$paid)
df$activities<- as.factor(df$activities)
df$nursery<- as.factor(df$nursery)
df$higher <- as.factor(df$higher)
df$internet <- as.factor(df$internet)
df$romantic <- as.factor(df$romantic)
df$school <- NULL
G1 <- df$G1
G2 <- df$G2


## ---------------------------------------------------------------------------------------------------
y <-cut(df$G3, seq(0,20,4), labels=c("F","D","C","B","A"),include.lowest=T)


## ---------------------------------------------------------------------------------------------------
dfnum <- df %>%
  correlation() %>%
  summary()

dfnums <- append(dfnum$Parameter,'G3')
dfnum <- as.data.frame(scale(df[dfnums]))
df_votefree <- dfnum[,-c(14,15,16)]
describe_distribution(dfnum[,-c(14,15,16)])


## ---- fig.width=7, fig.height=4.5-------------------------------------------------------------------------------------
set.seed(42)
sample <- createDataPartition(y = y, p=0.95, list = F)
df_sample <-  dfnum[-sample,]
y_sample <- y[-sample]
distance <- get_dist(df_sample)
d_1 <- fviz_dist(distance, gradient = list(low = "#00AFBB", mid = "white", high = "#FC4E07"),show_labels = F)
df_sample_2 <-  df_votefree[-sample,]
y_sample <- y[-sample]
distance <- get_dist(df_sample_2)
d_2 <- fviz_dist(distance, gradient = list(low = "#00AFBB", mid = "white", high = "#FC4E07"),show_labels = F)

annotate_figure(ggarrange(d_1,d_2,labels = c("G1,G2,G3 included","G1,G2,G3 excluded")),
                top = text_grob("", color = "black", face = "bold", size = 14),
                fig.lab = "Distant Matirces", fig.lab.face = "bold")


## ---------------------------------------------------------------------------------------------------
k2 <- kmeans(df_sample, centers = 2, nstart = 25)


## ---------------------------------------------------------------------------------------------------
fviz_cluster(k2, data = df_sample_2)


## ----fig.align='center',fig.width=7-------------------------------------------------------------------------------------
k3 <- kmeans(df_sample_2, centers = 3, nstart = 25)
k4 <- kmeans(df_sample_2, centers = 4, nstart = 25)
```

```r
k5 <- kmeans(df_sample_2, centers = 5, nstart = 25)

# plots to compare
p1 <- fviz_cluster(k2, geom = "point", data = df_sample_2) + ggtitle("k = 2")
p2 <- fviz_cluster(k3, geom = "point",  data = df_sample_2) + ggtitle("k = 3")
p3 <- fviz_cluster(k4, geom = "point",  data = df_sample_2) + ggtitle("k = 4")
p4 <- fviz_cluster(k5, geom = "point",  data = df_sample_2) + ggtitle("k = 5")

library(gridExtra)
grid.arrange(p1, p2, p3, p4, nrow = 2)


## ----fig.align='center'-------------------------------------------------------------------------------------------
set.seed(42)

k <- 15

wss<-sapply(1:k ,function(k){kmeans(df_votefree,k,nstart=50,iter.max=15)$tot.withinss})

plot(1:k, wss, type="b", pch=18, xlab="Number of clusters K", ylab="Total within-clusters sum of squares", col="orange")


## ----fig.align='center'-------------------------------------------------------------------------------------------
fviz_nbclust(df_votefree, kmeans, method = "silhouette")


## ---------------------------------------------------------------------------------------------------------------
#kmeans2 <- kmeans(df_votefree,2)
ris2 <- eclust(df_votefree,"kmeans",k=2)
avg_s_2 <- fviz_silhouette(ris2) + labs(title= "K = 2",
                                        subtitle= " Avg Silhoutte width:")


## ---------------------------------------------------------------------------------------------------------------
avg_s_2


## ---------------------------------------------------------------------------------------------------------------
final <- kmeans(df_votefree, 2, nstart = 25)


## ---------------------------------------------------------------------------------------------------------------
df_votefree %>%
  mutate(Cluster = final$cluster) %>%
  group_by(Cluster) %>%
  summarise_all("mean")
```