Concrete compressive strength prediction with machine learning

Pedro Bernardino Alves Moreira

April 22, 2020

Abstract

Compressive strength is the main characteristic of concrete. The correct prediction of this parameter means cost and time reduction. This work built predictive models for 6 different ages of concrete samples (3, 7, 14, 28, 56, and 100 days). A set of data obtained in previous studies was used, a total of 1030 samples, with 9 variables: compressive strength, age, and 7 ingredients (water, cement, fine aggregate, coarse aggregate, fly ash, blast furnace slag, and superplasticizers). Another 6 variables were added to represent the proportions of the main ingredients in each sample (water/cement, fine aggregate/cement, coarse aggregate/cement, fine aggregate/coarse aggregate, water/coarse aggregate, and water/fine aggregate). The predictive models were developed in *R* language, using the *caret* package with the *Parallel Random Forest* algorithm and repeated cross-validation technique to optimize the parameters. The results were satisfactory and compatible with other studies using the same data set. The most important model, 28 days old, obtained *RMSE* of 4.717. The 3-day model obtained the best result, *RMSE* of 3.310. The worst result was the 56-day model, with *RMSE* of 5.939. The work showed that the compressive strength of concrete can be predicted. The choice of creating a model for each age, instead of using age as a predictor, allowed to get compatible results with the available data at each age. It was a promising alternative since good results were achieved by training with just one algorithm. This work facilitates exploration and new efforts to predict the compressive strength of concrete, it can be replicated using different algorithms or the combination of several.

Contents

List of Figures

List of Tables

1 Introduction

Compressive strength is the main characteristic of concrete, measured by tests of international standards that consist of the breaking of specimens. Measurement at 28 days is mandatory and represents the grade of the concrete. Knowing in advance what the result will be obtained for a given age, based on the proportions of its ingredients, is of great interest to concrete manufacturers, construction companies, and civil engineers.

This compressive strength is a nonlinear function of its ingredients and age, making it difficult to establish an analytical formula, although some formulas have already been proposed. Hasan (2011) proposed a mathematical model to predict from the results of tests of 7 and 14 days, and Kabir (2012) from 7 days. However, machine learning techniques can be used to model this characteristic from real sample data, using only the ingredients.

Many previous studies use the same dataset used by Yeh (1998) to predict the compressive strength of concrete. Alshamiri (2020) got good results with the regularized extreme learning machine (RELM) technique, and Hameed (2020) got even better results with the Artificial Neural Networks and cross-validation technique. This set of samples is so well known that there are many pages on the internet of unpublished studies that use it and have good results, such as Abban (2016), Raj (2018), Modukuru (2020) and Pierobon (2018). At the end of the work, the results found are compared to the works cited here.

Unlike previous studies with this dataset, this work does data preparation differently. The age of the concrete is the most unique feature that contributes to its compressive strength. For this reason, age is treated separately in the machine learning models, creating models for each age group.

2 Materials and Methods

2.1 Materials

The methodology was carried out using RStudio software (RStudio Team 2020), an integrated virtual environment for code development in *R* (R Core Team 2020). Throughout the process, all code executed was documented in the same order as its execution in Appendix 3, and reference was always made to codes throughout the text. All relevant information related to the operating system and installed packages has been presented in Appendix 1. In addition, an online and open-source repository was created in *Github*, housing all the code used to generate this work, the link was made available in Appendix 2.

2.2 Reproducibility

In order to guarantee reproducibility, whenever there was some code that could use probabilistic operations, a *seed* was defined before its execution, ensuring that when run on another machine, with the same version of *R* and the same *seed*, get the same result. The *seeds* can be checked throughout Appendix 3 or directly on *Github*.

2.3 Obtaining the data

The data was downloaded from the University of California Irvine website ("Concrete Compressive Strength Data Set" 2008) (8.1.1). In total there are 1030 samples with 9 columns. The samples were renamed and an id column was added to facilitate data manipulation (8.1.2). The columns were reordered to put the new id column in the first position (8.1.3). The first samples can be viewed in the table 1.

Table 1: First 6 samples

ID	Cement kg/m^3	B.F.S. kg/m^3	Fly ash kg/m^3	Water kg/m^3	Superp. kg/m^3	C.Aggregate kg/m^3	F.Aggregate kg/m^3	Day	Comp.Str. MPa
1	540.0	0.0	0	162	2.5	1040.0	676.0	28	79.99
2	540.0	0.0	0	162	2.5	1055.0	676.0	28	61.89
3	332.5	142.5	0	228	0.0	932.0	594.0	270	40.27
4	332.5	142.5	0	228	0.0	932.0	594.0	365	41.05
5	198.6	132.4	0	192	0.0	978.4	825.5	360	44.30
6	266.0	114.0	0	228	0.0	932.0	670.0	90	47.03

2.4 Data preparation

The preparation of the data consisted of transforming the sample set in order to maintain only relevant data for the subsequent studies. Data that were considered irrelevant or that had the potential to add undesirable noise to the analysis were removed. In addition, the relevant data has been transformed to better fit the studies in the next steps.

2.4.1 Initial data cleaning

Initially, there were 25 duplicate samples that were removed, resulting in a new total of 1005 samples (8.2.1).

The data show the variables in the columns and samples in the rows. However it was found that some samples are identical in proportions of ingredients, changing only the value of age and compressive strength, for example, samples 653, 654, 678 and 681, shown in the table 2.

Table 2: Samples with same composition

ID	Cement kg/m^3	B.F.S. kg/m^3	Fly ash kg/m^3	Water kg/m^3	Superp. kg/m^3	C.Aggregate kg/m^3	F.Aggregate kg/m^3	Day	Comp.Str. MPa
653	102	153	0	192	0	887	942	3	4.57
678	102	153	0	192	0	887	942	7	7.68
681	102	153	0	192	0	887	942	28	17.28
654	102	153	0	192	0	887	942	90	25.46

In addition, there are also samples with the same values and proportions of ingredients, but with different compressive strength, probably due to differences in the building process. This is the case, for example, of samples 472, 473 and 474, shown in the table 3.

To facilitate the analysis of the samples, all samples that are the same in relation to the ingredients, have been assigned the same *id*. In addition, as the compressive strength at 28 days is the parameter to determine the grade of the concrete, only the elements containing that day among its samples were maintained. In the case of the same samples but with different results of compressive strength, the values were averaged. After all these changes (8.2.4), the new total of samples was reduced to 970, containing 416 different settings for the proportions of ingredients.

Table 3: Same samples with different results

ID	Cement kg/m^3	B.F.S. kg/m^3	Fly ash kg/m^3	Water kg/m^3	Superp. kg/m^3	C.Aggregate kg/m^3	F.Aggregate kg/m^3	Day	Comp.Str. MPa
472	446	24	79	162	11.6	967	712	28	57.03
473	446	24	79	162	11.6	967	712	28	44.42
474	446	24	79	162	11.6	967	712	28	51.02

The result can be seen in the table 4. All samples with equal ingredient settings have the same id, and when they had different results for the same days, they were transformed into just one sample, with the arithmetic mean in the compressive strength.

Table 4: Previous samples after processing

ID	Cement kg/m^3	B.F.S. kg/m^3	Fly ash kg/m^3	Water kg/m^3	Superp. kg/m^3	C.Aggregate kg/m^3	F.Aggregate kg/m^3	Day	Comp.Str. MPa
472	446	24	79	162	11.6	967	712	28	50.82
653	102	153	0	192	0.0	887	942	3	4.57
653	102	153	0	192	0.0	887	942	7	7.68
653	102	153	0	192	0.0	887	942	28	17.28
653	102	153	0	192	0.0	887	942	90	25.46

2.4.2 Age selection

As previously described, the main age for analysis of compressive strength is 28 days, but other ages can also be used to build predictive models. However, it is necessary to verify how relevant the data of these other ages are. Starting with the distribution of the samples in relation to each age (8.2.6) shown in the figure 1.

It was observed that the ages of 90, 91 and 100 days probably represent extremes to each other in the ingredient configurations, since they are relatively close ages but with very different values, especially for 90 and 91.

This hypothesis was verified using the principal component analysis method, applied to samples of these 3 ages (8.2.7). The figure 2 shows how the samples relate to each other (which are similar or different) and revealed how each variable contributes to the analysis. The first two dimensions represent 37% and 24% $ respectively of the variance.

Another important point considered, of the concrete nature itself, is the fact that the growth rate of its compressive strength decreases with time, reaching a certain stability value. The figure 3 shows the compressive strength over the days for samples with more than 5 data, that is, data available for at least 6 different ages (8.2.8).

For the reasons presented in the figures 1, 2 and 3, it was considered that the ages of 90, 91 and 100 days can be grouped to improve reading and decrease sample noise. They were converted to the same value, the age of 100 days was chosen (8.2.9). As shown in the figure 3, the resistance only increases, after 100 days the resistance to compression will be greater than or equal to the value of 90 or 91 days.

Figure 1: Boxplot - Compressive strength (MPa) vs age (days)

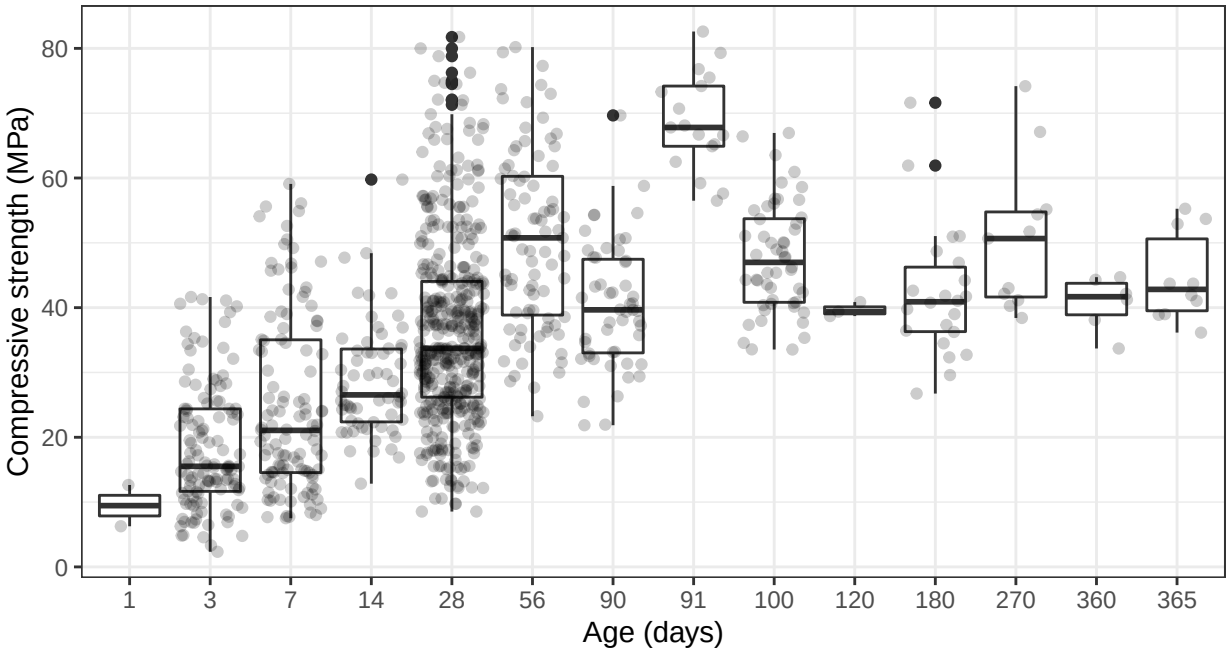


Figure 2: Principal component analysis - 90, 91 e 100 days

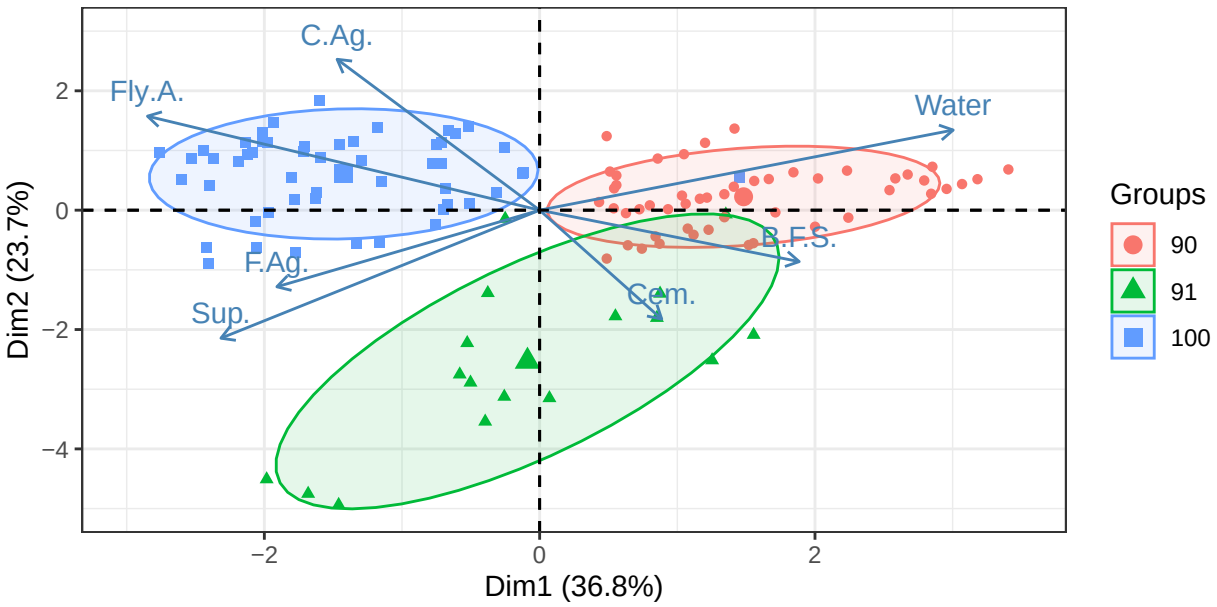
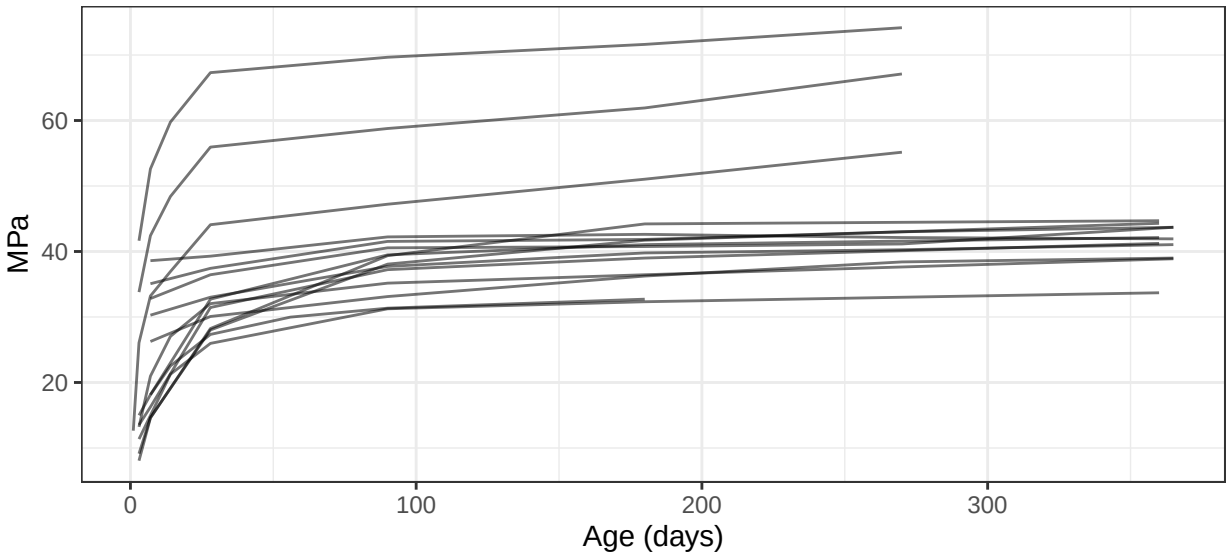
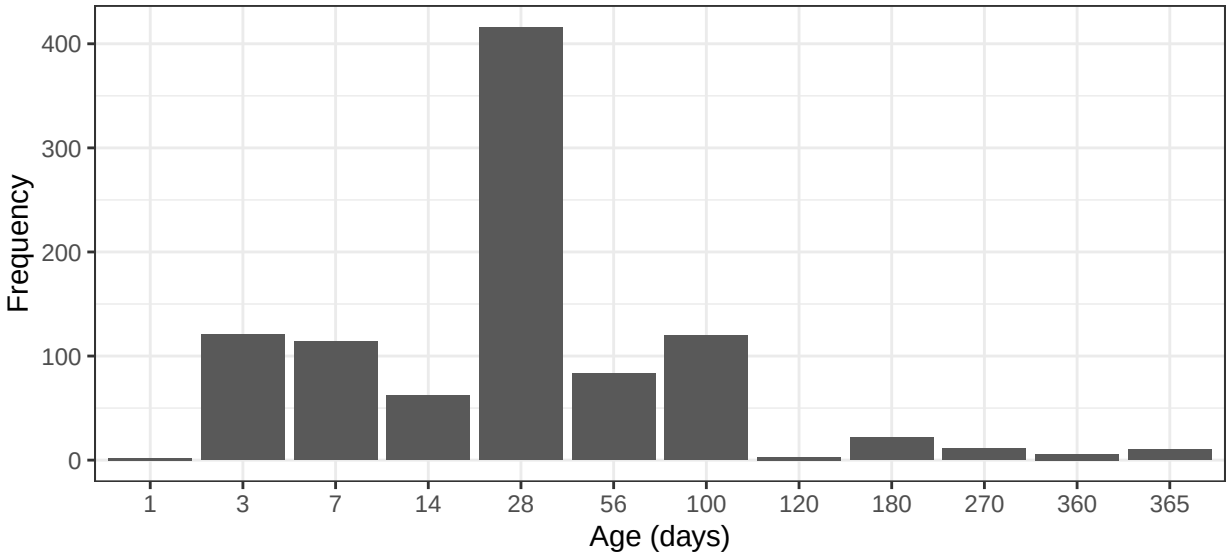


Figure 3: Compressive strength through time



Another topic analyzed in the selection of ages was the observed frequency of each age value after this transformation from 90, 91 in 100 days, shown in the figure 4. Some values of days have very low concentrations of samples, at the risk of damaging more than helping to create the models, so they were removed (8.2.11). The criterion adopted was to maintain only ages with a frequency greater than 50, only the values of 3, 7, 14, 28, 56 and 100 days.

Figure 4: Ages frequency



2.4.3 Data reorganization

The samples were grouped to maintain only one sample from each set of configuration of the proportions of the ingredients, adding new variables/columns for resistance at each age (8.2.12). The result in the first

samples after this processing is shown in the table 5.

Table 5: First 6 samples after reorganization

ID	Cement kg/m^3	B.F.S kg/m^3	Fly ash kg/m^3	Water kg/m^3	Superp. kg/m^3	Coarse Ag. kg/m^3	Fine Ag. kg/m^3	3 days MPa	7 days MPa	14 days MPa	28 days MPa	56 days MPa	100 days MPa
1	540.0	0.0	0	162	2.5	1040.0	676.0				79.99		
2	540.0	0.0	0	162	2.5	1055.0	676.0				61.89		
3	332.5	142.5	0	228	0.0	932.0	594.0		30.28		33.02		37.72
5	198.6	132.4	0	192	0.0	978.4	825.5	9.13	14.64		28.02		38.07
6	266.0	114.0	0	228	0.0	932.0	670.0				45.85		47.03
7	380.0	95.0	0	228	0.0	932.0	594.0		32.82		36.45		40.56

The number of samples and distinct samples after all this manipulation remained the same, a total of 416 (8.2.14).

2.4.4 Adding new variables

To finish the data preparation, new columns were added to the dataset (8.2.15). Starting with the concrete class, for example if the compressive strength is between 25 and 30, it receives the class *C25*. The inclusion of the class was important because the compressive strength in *MPa* is a continuous variable, which will be used in the regression models, but the class as a discrete variable can provide another visualization of the data. The approximate mix of concrete was also added, which represents the proportions of aggregates (fine and coarse) for cement. Other proportions between the main ingredients were also added. The new variables are presented in the table 6.

Table 6: New features

ID	Class	Approximated Mix	Water / Cement	Fine Ag. / Cement	Coarse Ag. / Cement	Fine Ag. / Coarse Ag.	Water / Coarse Ag.	Water / Fine Ag.
1	C75	1:1:2	0.3000	1.2519	1.9259	0.6500	0.1558	0.2396
2	C60	1:1:2	0.3000	1.2519	1.9537	0.6408	0.1536	0.2396
3	C30	1:2:3	0.6857	1.7865	2.8030	0.6373	0.2446	0.3838
5	C25	1:4:5	0.9668	4.1566	4.9265	0.8437	0.1962	0.2326
6	C45	1:3:4	0.8571	2.5188	3.5038	0.7189	0.2446	0.3403
7	C35	1:2:2	0.6000	1.5632	2.4526	0.6373	0.2446	0.3838

2.5 Data visualization

In order to assess the need for further manipulation before building the models, in this step the 416 samples already processed were visualized and analyzed.

2.5.1 Descriptive statistics

The table 7 presents the statistical data of the continuous variables (8.3.1). The *Null* line represents the number of zeroed values for the ingredients, and the *NA* line represents the number of missing data. As the samples were filtered to maintain only sets of samples with known values of compressive strength at 28 days, the number of *NAs* is zero for that age. The figure 5 presents the statistical data of the discrete variables (8.3.2).

Table 7: Descriptive statistics - continuous variables

	Samples	Null	NA	Min	Max	Range	Sum	Median	Mean	SE mean	CI mean	Variance	Std.Dev.	Coef.Var
Cement	416	0	0	102.00	540.00	438.00	109373.10	257.70	262.92	5.10	10.02	10817.50	104.01	0.40
B.F.S.	416	174	0	0.00	359.40	359.40	35824.60	94.25	86.12	4.32	8.49	7755.00	88.06	1.02
Fly ash	416	202	0	0.00	200.10	200.10	26389.00	71.25	63.44	3.26	6.40	4407.81	66.39	1.05
Water	416	0	0	121.80	247.00	125.20	76335.60	185.00	183.50	0.94	1.86	370.73	19.25	0.10
Superplast.	416	107	0	0.00	32.20	32.20	2871.30	7.60	6.90	0.26	0.52	28.85	5.37	0.78
Coarse agg.	416	0	0	801.00	1145.00	344.00	397799.90	953.35	956.25	4.12	8.10	7063.06	84.04	0.09
Fine agg.	416	0	0	594.00	992.60	398.60	317809.80	769.65	763.97	3.59	7.06	5371.89	73.29	0.10
3 days	121	0	295	2.33	41.64	39.31	2210.82	15.52	18.27	0.87	1.72	91.64	9.57	0.52
7 days	114	0	302	7.51	59.09	51.58	2845.52	21.06	24.96	1.29	2.55	188.81	13.74	0.55
14 days	62	0	354	12.84	59.76	46.92	1782.56	26.54	28.75	1.10	2.19	74.62	8.64	0.30
28 days	416	0	0	8.54	81.75	73.21	15101.13	33.72	36.30	0.70	1.38	206.30	14.36	0.40
56 days	83	0	333	23.25	80.20	56.95	4178.77	50.77	50.35	1.52	3.02	190.82	13.81	0.27
100 days	120	0	296	21.86	82.60	60.74	5701.90	45.61	47.52	1.17	2.31	163.40	12.78	0.27
Water / Cement	416	0	0	0.27	1.88	1.62	340.60	0.73	0.82	0.02	0.03	0.11	0.34	0.41
Fine agg. / Cement	416	0	0	1.14	9.24	8.10	1415.82	2.94	3.40	0.07	0.14	1.96	1.40	0.41
Coarse agg. / Cement	416	0	0	1.55	8.70	7.14	1761.06	3.67	4.23	0.08	0.16	2.70	1.64	0.39
Fine agg. / Coarse agg.	416	0	0	0.53	1.16	0.63	335.42	0.80	0.81	0.01	0.01	0.01	0.11	0.14
Water / Coarse agg.	416	0	0	0.12	0.29	0.17	80.66	0.19	0.19	0.00	0.00	0.00	0.03	0.16
Water / Fine agg.	416	0	0	0.13	0.38	0.26	101.26	0.24	0.24	0.00	0.00	0.00	0.04	0.17

2.5.2 Correlation between ingredients and compressive strength

The figure 6 shows the correlation of variables for each set of ages (8.3.3). The figure 7 presents the same data, but instead of correlating them all, it only correlates with the compressive strength, showing the values in more detail (8.3.4).

The interpretation of the figure 7 suggests that the strength of the concrete is positively related mainly to the cement and superplasticizer ingredients and negatively to the water and fine aggregate. The smaller the amount of cement for aggregates and water, the more negatively they are correlated with compressive strength.

The figures 8 and 9 show the relationship between the main ingredients (known as mix) in relation to the compressive strength (8.3.5 and 8.3.6). The interpretation of these figures shows that the greater the amount of cement in relation to the other ingredients, the greater the resistance to compression.

2.5.3 Variables distribution

The figure10 shows the distribution of variables in the samples (8.3.7). It was calculated using data only at 28 days.

The figure 11 shows the distribution of ingredients and compressive strength for each set of ages (8.3.8), in case of the 28 days it presents the same information as the figure 10. It shows that the resistance to compression gradually increases over time, as expected. Furthermore it is seen that the concentration of the ingredients can vary a lot when stratified by ages.

Figure 5: Descriptive statistics - categorical variables

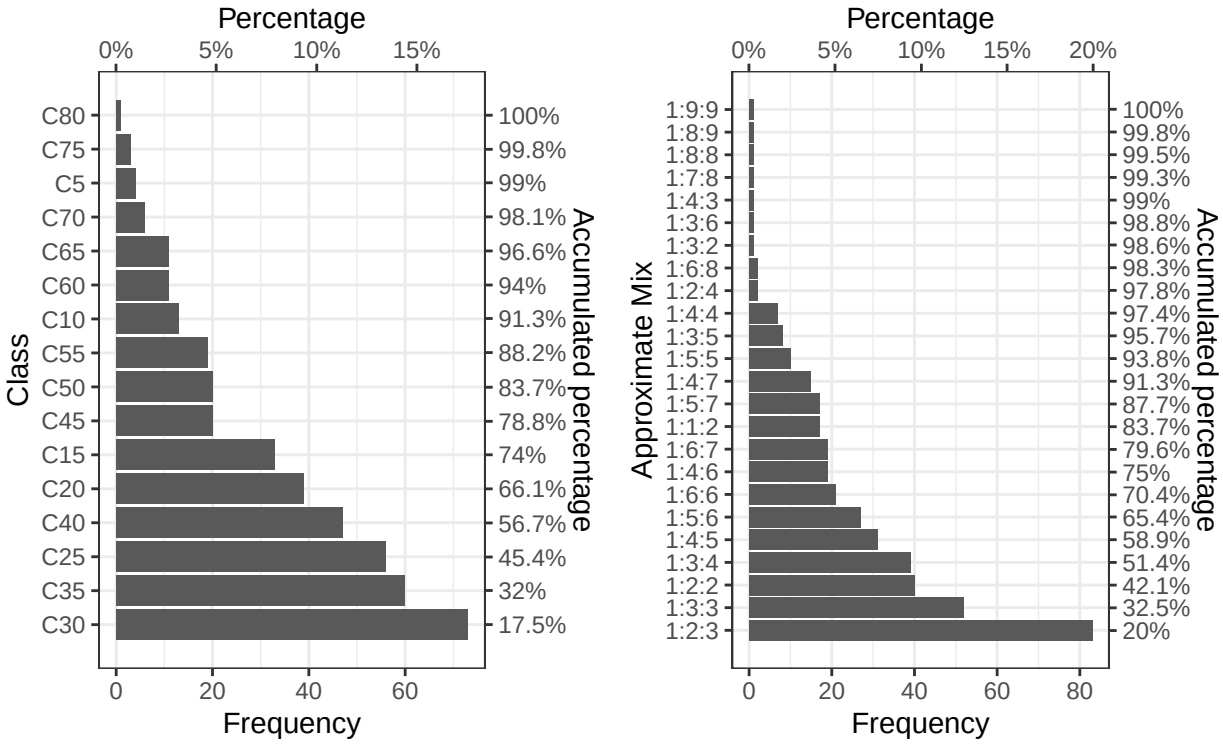


Figure 6: Correlations at each age

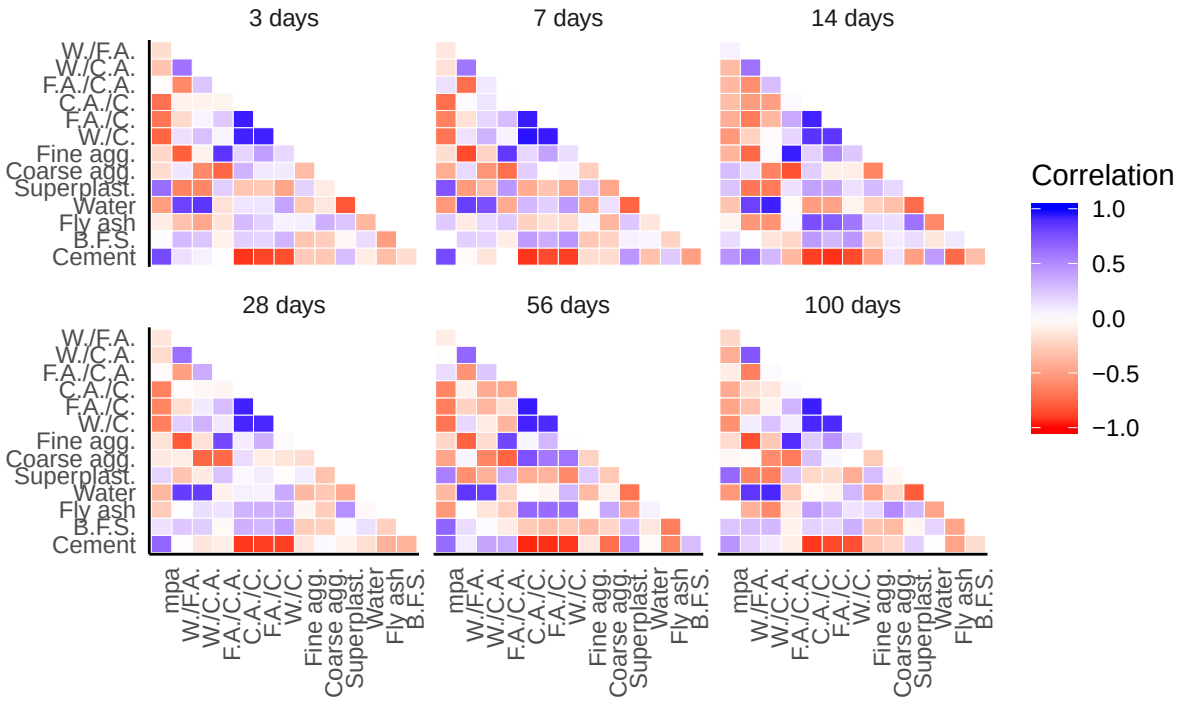


Figure 7: Correlation of variables with compressive strength over time

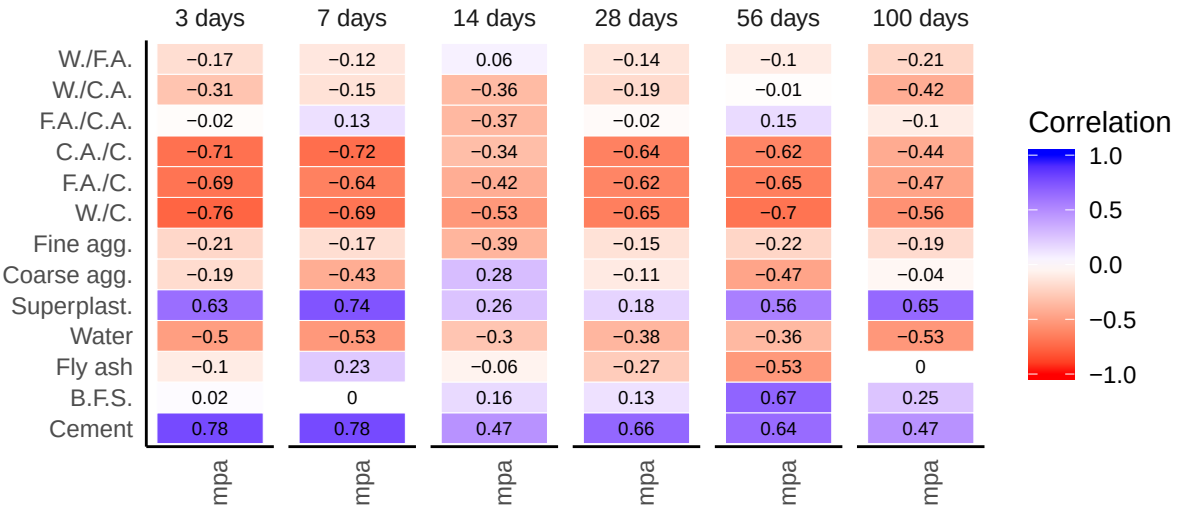


Figure 8: Relationship between approximated mix, water, MPa and age

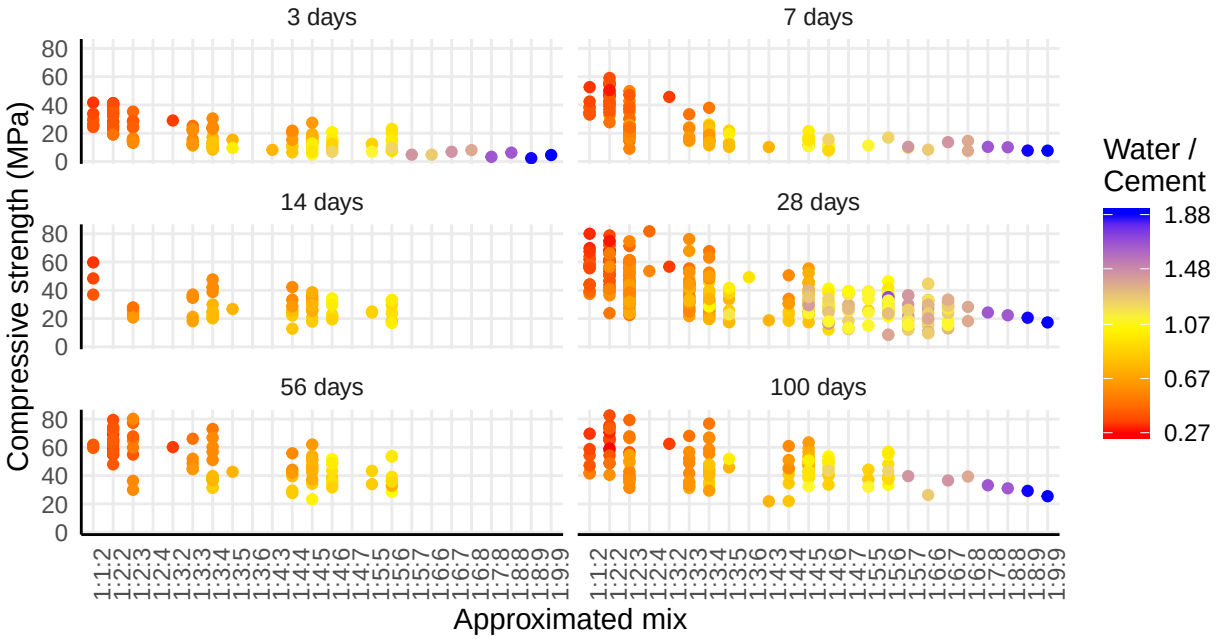
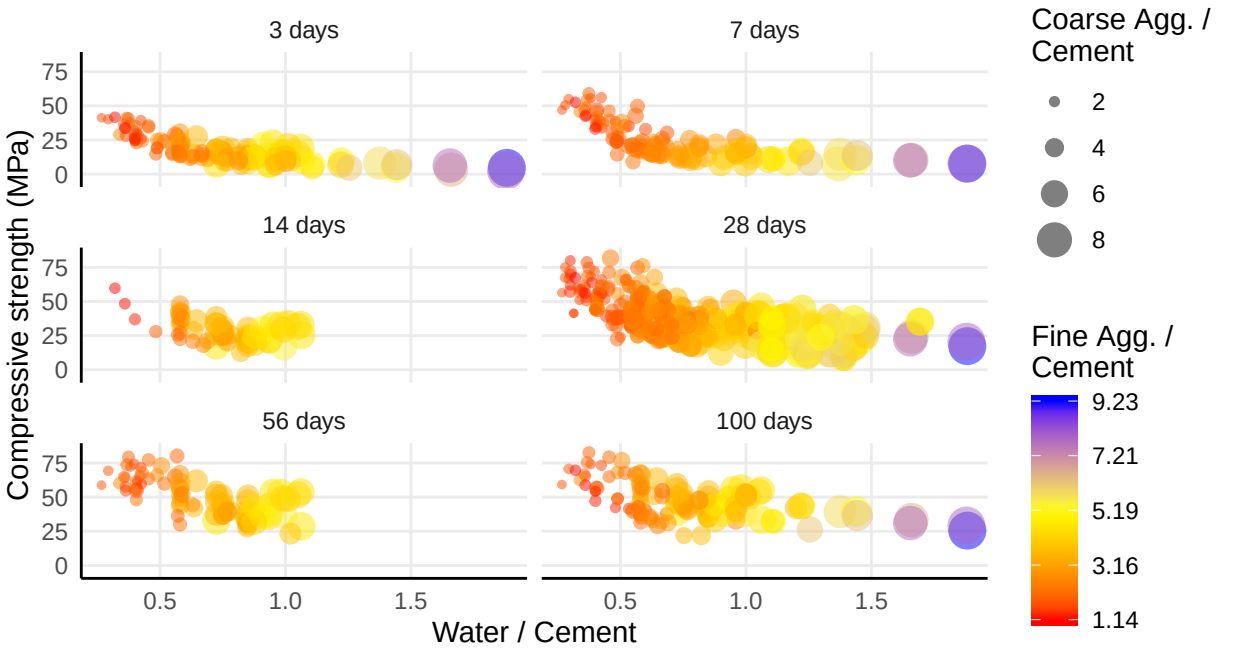


Figure 9: Relationship between concrete main features



2.5.4 Principal component analysis

In the figure 12, using an alternative classification, the principal component analysis was performed on the ingredients (8.3.9). The classification separates concrete into 4 different compressive strength groups, low up to *20 MPa*, normal up to *40 MPa*, medium up to *70 MPa* and high above that. It is possible to notice that the groups overlap, but there is a differentiation between the high and low group.

2.6 Machine learning models

The development of machine learning models was carried out with the *caret* package (Kuhn 2020) and based on Irizarry (2019) and Kuhn (2008).

2.6.1 Pre-processing and data separation

As the approximate mix is a categorical variable, it was converted into dummy variables (8.4.1), going from 22 columns (id, class, compressive strength and 19 more *features*) to 45 columns, an addition of 23 variables, one for each approximate mix.

The samples were separated based on age. One dataset was created for each age value, totaling 6 different sets (8.4.2). For illustrative purposes, the first 18 of 45 columns of the first 6 samples from the 28-day set are shown in the table 8.

Figure 10: Variables distribution

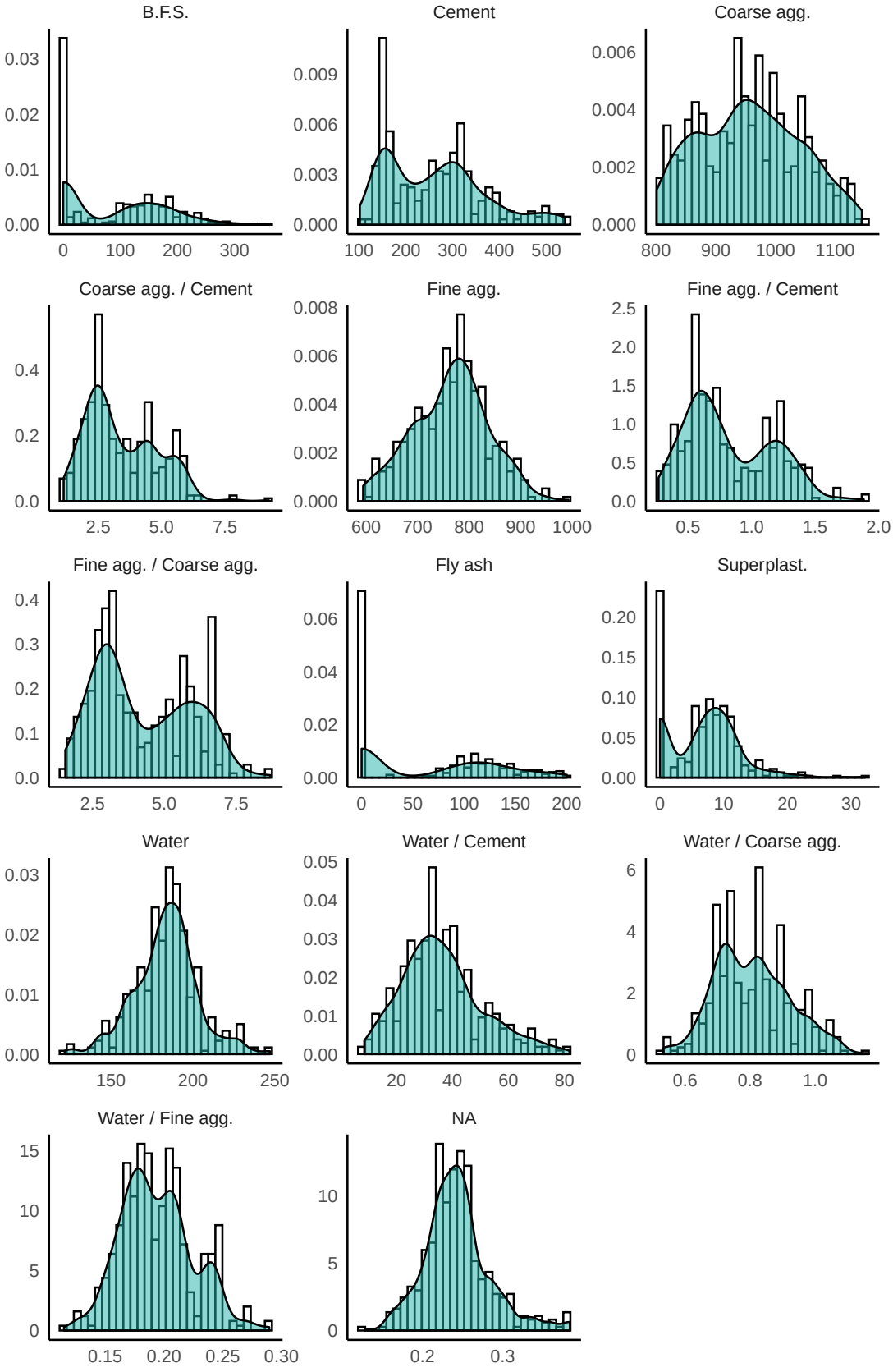


Figure 11: Variables distribution grouped by age

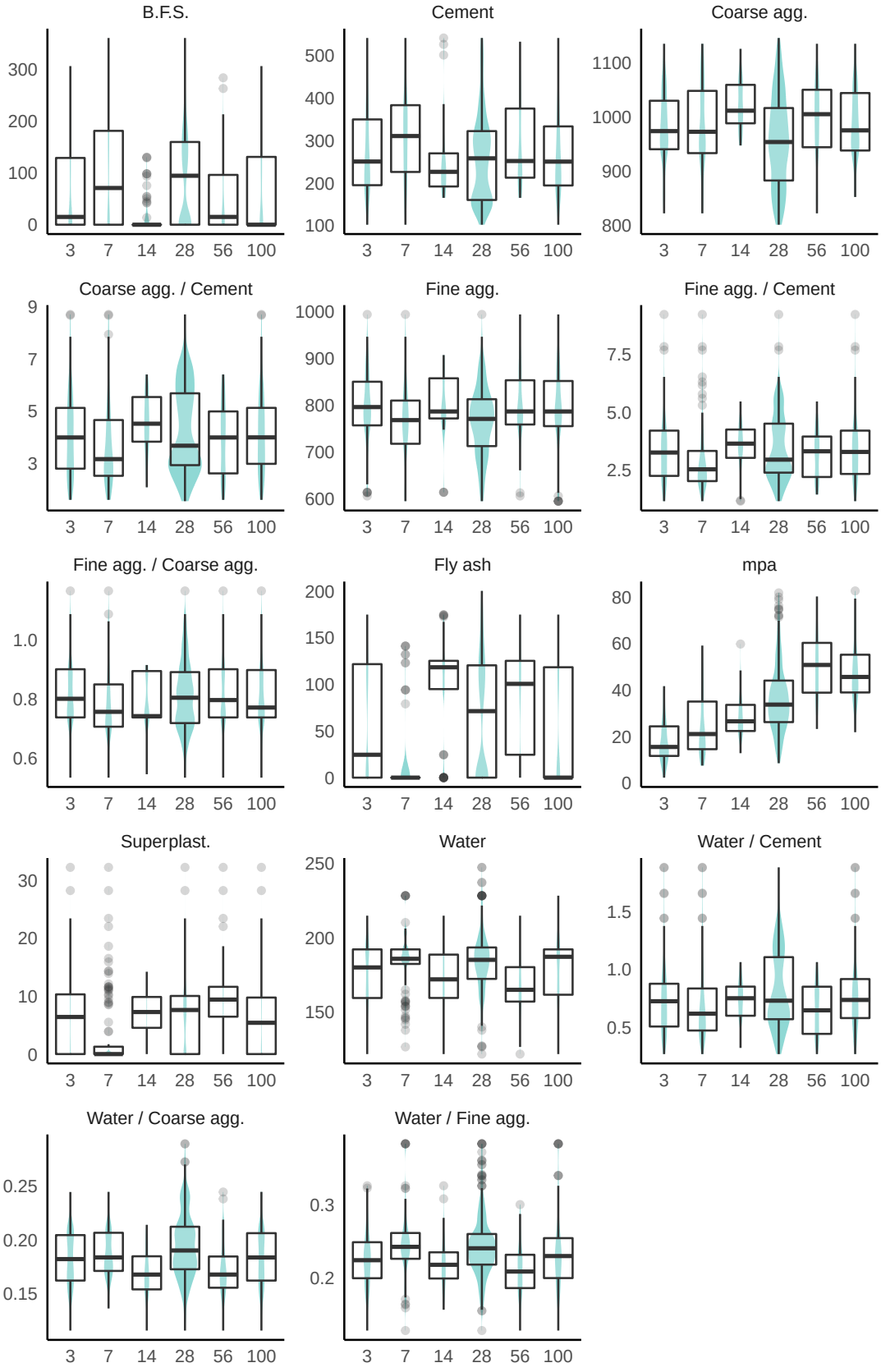


Figure 12: Principal component analysis on ingredients

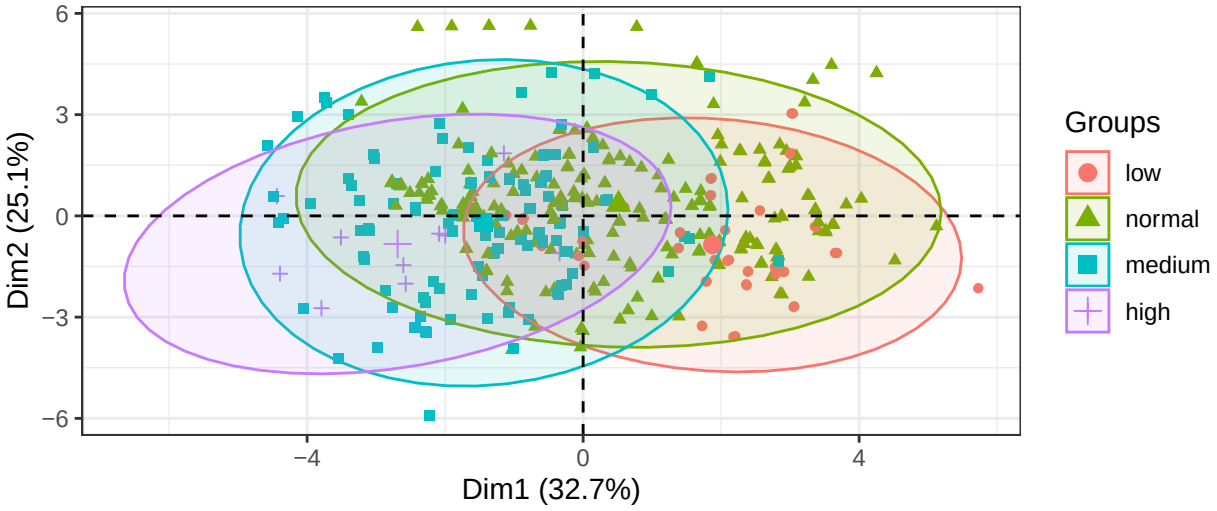


Table 8: First 18 columns of the 6 first samples of 28 days

ID	Cement kg/m^3	B.F.S. kg/m^3	Fly ash kg/m^3	water kg/m^3	Superp. kg/m^3	Coarse Agg. kg/m^3	Fine Agg. kg/m^3	MPa MPa	Class	Wat./ Ci.	F.Agg./ Ce.	C.Agg./ Ce.	F.Agg./ C.Agg.	Wat./ C.Agg.	Wat./ F.Agg.	App Mix 1:1:2	App Mix 1:2:2
1	540.0	0.0	0	162	2.5	1040.0	676.0	79.99	C75	0.30	1.25	1.93	0.65	0.16	0.24	1	0
2	540.0	0.0	0	162	2.5	1055.0	676.0	61.89	C60	0.30	1.25	1.95	0.64	0.15	0.24	1	0
3	332.5	142.5	0	228	0.0	932.0	594.0	33.02	C30	0.69	1.79	2.80	0.64	0.24	0.38	0	0
5	198.6	132.4	0	192	0.0	978.4	825.5	28.02	C25	0.97	4.16	4.93	0.84	0.20	0.23	0	0
6	266.0	114.0	0	228	0.0	932.0	670.0	45.85	C45	0.86	2.52	3.50	0.72	0.24	0.34	0	0
7	380.0	95.0	0	228	0.0	932.0	594.0	36.45	C35	0.60	1.56	2.45	0.64	0.24	0.38	0	1

For each of the 6 sets, the existence or not of variables with variance close to zero and their subsequent removal was verified (8.4.4). Many of the 23 variables added referring to the approximate concrete mix were removed due to this fact. In addition to them, in the case of the 7-day set, the fly ash variable was also removed. Then it was verified that there are no variables with high correlation, above 0.999, in any of the 6 data sets (8.4.5). After these steps, the sample sets presented 24, 21, 23, 23, 24 and 25 columns respectively for ages in increasing sequence.

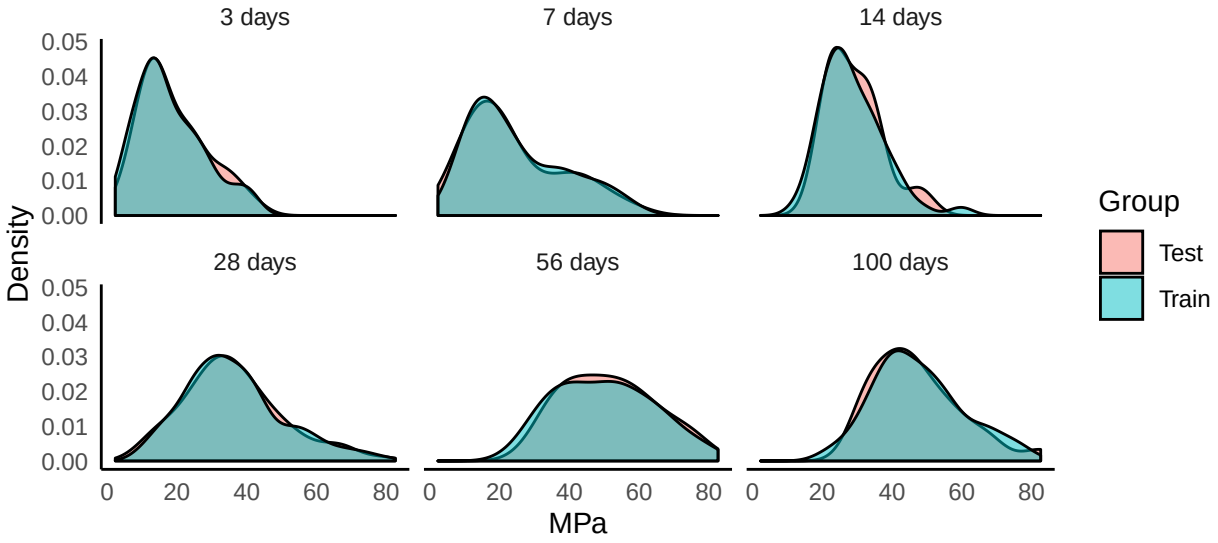
The stage of centralization and normalization of the variables was carried out later, together with the application of the models, as it is simpler to do this with the *caret* package. If performed at this time, it would be necessary to manually undo these transformations in the predictions. The *caret* allows you to transform before training the models and already transforms the results back.

Each data set was separated into test and training sets, *20%* and *80%* respectively (8.4.6). The figure 13 shows the distribution of data between the sets in relation to the compressive strength for each model (8.4.7).

2.6.2 Performance measures

The performance evaluation of the models was performed by the Root Mean Square Error (*RMSE*). The *RMSE* is the measure used in all the works mentioned in the introduction and will allow the comparison of the models in the discussion.

Figure 13: Distribution of test and train data



2.6.3 Naive models

Before creating the real models, for comparison purposes, naive models were created. They simply predict that the compressive strength of the test set is the average compressive strength of the training set (8.4.8). In other words, naive models are simply the best guess possible. The results can be checked in the table 9.

Table 9: Naive models

Age	Mean MPa (train)	RMSE (train)	RMSE (test)
3	18.08887	9.591344	9.303229
7	25.12383	13.731569	13.443646
14	28.63980	8.786823	7.593319
28	36.33605	14.361021	14.283824
56	50.06555	13.968077	12.702112
100	47.57000	12.758042	12.614652

2.6.4 Choice of algorithms

The *caret* (Kuhn 2020) package exposes more than 200 different algorithms for building *machine learning* models. The package documentation presents an initial code ("Models Clustered by Tag Similarity" 2020) as a suggestion to select a portfolio of the most distinct algorithms possible in relation to some pre-selected algorithm, but for agility and due to technical limitations, it was chosen to use an algorithm with the highest probability to achieve the best possible result. According to Fernandez-Delgado et al. (2014), who compared 179 algorithms in 121 different databases, the algorithm most likely to achieve the best possible results is the *Parallel Random Forest* (called *prRF* in *caret*).

2.6.5 Regression models

As new variables were added throughout the processing (the relationships between the ingredients and a few more *dummy vars* for each age set), 5 possibilities for configuring the *features* for the models were studied:

1. All *features*;
2. All *features* without the *dummy vars*;
3. Only the original *features*;
4. Only new *features*;
5. Only new *features*, without *dummy vars*;

Building a model for each of these configurations using the set at 28 days (8.4.10), showed that the best option is configuration 2, the *dummy vars* were completely discarded, but the other new variables were kept. For illustrative purposes, the 10 table shows the first 6 samples from the 28-day model training set. The samples of the other models, of the test and training sets are similar, the only difference being in the 7-day model, which excludes fly ash due to the variance close to zero, performed previously.

Table 10: First 6 samples of train data of the 28 days model

Features													Outcome
Cement kg/m^3	B.F.S. kg/m^3	Fly ash kg/m^3	water kg/m^3	Superp. kg/m^3	Coarse Agg. kg/m^3	Fine Agg. kg/m^3	Wat./ Ce.	F.Agg./ Ce.	C.Agg./ Ce.	F.Agg./ C.Agg.	Wat./ C.Agg.	Wat./ F.Agg.	y MPa
540.0	0.0	0	162	2.5	1040.0	676.0	0.30	1.25	1.93	0.65	0.16	0.24	79.99
540.0	0.0	0	162	2.5	1055.0	676.0	0.30	1.25	1.95	0.64	0.15	0.24	61.89
266.0	114.0	0	228	0.0	932.0	670.0	0.86	2.52	3.50	0.72	0.24	0.34	45.85
380.0	95.0	0	228	0.0	932.0	594.0	0.60	1.56	2.45	0.64	0.24	0.38	36.45
475.0	0.0	0	228	0.0	932.0	594.0	0.48	1.25	1.96	0.64	0.24	0.38	39.29
198.6	132.4	0	192	0.0	978.4	825.5	0.97	4.16	4.93	0.84	0.20	0.23	28.02

For each age set, a model was created using the *Parallel Random Forest* algorithm, previously defined (8.4.12). For each of the 6 models, the parameter *mtry* was optimized, and *repeated cross-validation* was performed, dividing into 10 or 30 parts and repeating 10 times.

3 Results

The test *RMSE* for each model in ascending order of age was 3.31, 4.36, 4.62, 4.72, 5.94 and 5.85 respectively. The table 11 presents the details and results of each model (8.5.1). The figure 14 compares the actual and predicted values (8.5.2), and the following tables show the best and worst results for each model (8.5.3).

Table 11: Regression models results

Model	mtry	CV	Repetitions	RMSE (train)	RMSE (test)
3 days	6	30	10	3.905196	3.310370
7 days	2	10	10	4.475981	4.361987
14 days	13	30	10	5.136687	4.620515
28 days	11	30	10	5.847334	4.716698
56 days	8	30	10	6.702565	5.939163
100 days	8	10	10	6.381940	5.851088

Figure 14: Actual vs Predicted values in each model

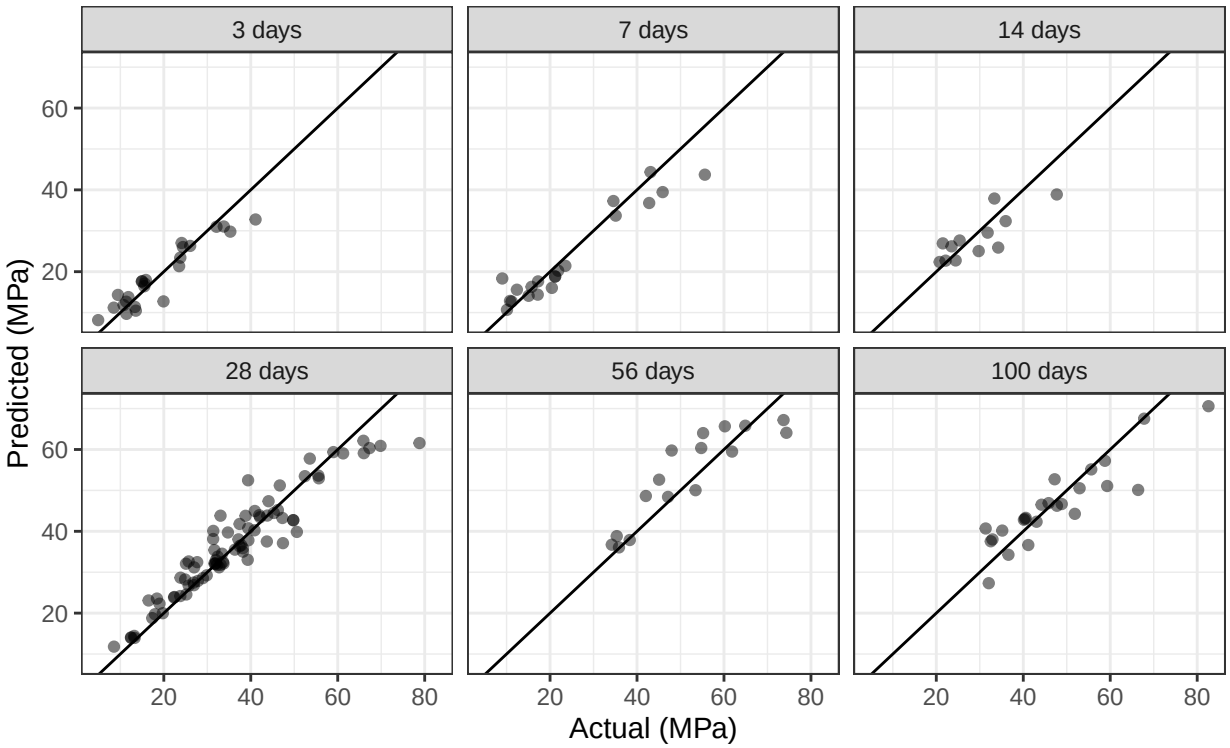


Table 12: Model of 3 days

Best 10			Worst 10		
Actual	Predicted	Error	Actual	Predicted	Error
26.06	26.297746	0.2377457	41.10	32.758262	-8.341738
23.80	23.432367	-0.3676333	19.93	12.732172	-7.197828
15.52	16.436292	0.9162920	35.30	29.781989	-5.518011
10.76	11.830651	1.0706507	9.45	14.316742	4.866742
32.11	30.977619	-1.1323807	4.90	8.154245	3.254245
11.36	12.660044	1.3000440	13.57	10.464572	-3.105428
15.62	17.079698	1.4596980	24.10	27.016321	2.916321
24.39	26.018261	1.6282610	33.80	31.045796	-2.754204
11.41	9.688879	-1.7211213	8.49	11.196910	2.706910
11.85	13.788864	1.9388640	14.99	17.677379	2.687379

Table 13: Model of 7 days

Best 10			Worst 10		
Actual	Predicted	Error	Actual	Predicted	Error
17.24	17.61947	0.3794733	55.60	43.71082	-11.889177
15.75	16.29318	0.5431847	9.01	18.32999	9.319987
10.09	10.67438	0.5843837	45.90	39.45277	-6.447230
15.07	14.10010	-0.9698955	42.80	36.78526	-6.014737
43.11	44.35190	1.2419040	20.42	16.03647	-4.383533
35.10	33.69951	-1.4004926	12.37	15.58829	3.218292
11.17	12.71080	1.5408013	17.17	14.38220	-2.787795
21.86	20.23260	-1.6274033	34.57	37.26745	2.697450
10.79	12.85092	2.0609233	21.16	18.79017	-2.369827
23.52	21.42631	-2.0936880	21.18	18.84231	-2.337694

Table 14: Model of 14 days

Best 10			Worst 10		
Actual	Predicted	Error	Actual	Predicted	Error
22.14	22.69248	0.5524823	47.71	38.87416	-8.835845
20.77	22.36561	1.5956093	34.24	25.88502	-8.354984
24.45	22.74412	-1.7058757	21.50	26.91740	5.417403
25.37	27.62261	2.2526130	29.75	25.02926	-4.720744
31.81	29.53499	-2.2750130	33.36	37.88168	4.521675
23.51	26.19615	2.6861493	35.96	32.35358	-3.606417
35.96	32.35358	-3.6064167	23.51	26.19615	2.686149
33.36	37.88168	4.5216750	31.81	29.53499	-2.275013
29.75	25.02926	-4.7207443	25.37	27.62261	2.252613
21.50	26.91740	5.4174030	24.45	22.74412	-1.705876

Table 15: Model of 28 days

Best 10			Worst 10		
Actual	Predicted	Error	Actual	Predicted	Error
27.83	27.86776	0.0377573	78.80	61.57401	-17.225986
43.80	43.84951	0.0495087	39.38	52.47227	13.092267
26.92	26.82812	-0.0918800	33.04	43.81819	10.778193
31.87	32.06794	0.1979360	50.60	39.85258	-10.747415
19.77	19.97517	0.2051693	47.40	37.11150	-10.288501
31.88	32.13467	0.2546680	69.84	60.88636	-8.953635
59.00	59.36857	0.3685727	31.38	40.09934	8.719345
23.79	24.17376	0.3837583	49.77	42.71779	-7.052213
28.99	28.55592	-0.4340790	49.77	42.72394	-7.046057
32.24	31.79166	-0.4483390	25.10	32.07616	6.976158

Table 16: Model of 56 days

Best 10			Worst 10		
Actual	Predicted	Error	Actual	Predicted	Error
35.85	36.06413	0.2141317	47.97	59.74774	11.777736
38.33	37.86955	-0.4604540	74.36	64.08547	-10.274529
64.90	65.81326	0.9132646	55.20	64.00472	8.804718
47.13	48.41224	1.2822413	45.08	52.62057	7.540568
61.86	59.49031	-2.3696857	42.03	48.63652	6.606522
34.20	36.72639	2.5263850	73.70	67.19523	-6.504768
53.46	50.04835	-3.4116453	54.77	60.36449	5.594488
35.34	38.82656	3.4865560	60.20	65.67068	5.470680
60.20	65.67068	5.4706798	35.34	38.82656	3.486556
54.77	60.36449	5.5944884	53.46	50.04835	-3.411645

Table 17: Model of 100 days

Best 10			Worst 10		
Actual	Predicted	Error	Actual	Predicted	Error
67.80	67.53752	-0.2624793	66.42	50.11266	-16.307345
55.64	55.13289	-0.5071103	82.60	70.59383	-12.006168
43.06	42.33515	-0.7248493	31.35	40.69152	9.341519
45.84	46.89673	1.0567320	59.30	51.07318	-8.226824
47.74	46.24704	-1.4929567	51.86	44.27586	-7.584144
58.78	57.23738	-1.5426163	47.22	52.72019	5.500191
48.85	46.67286	-2.1771403	32.92	38.02416	5.104161
44.21	46.50377	2.2937710	32.53	37.56541	5.035406
36.59	34.26929	-2.3207093	35.17	40.18011	5.010112
52.96	50.52831	-2.4316947	32.07	27.31722	-4.752779

4 Discussion

The models built present satisfactory results and prove that the compressive strength of concrete can be predicted relatively easily. The alternative adopted to create a model for each set of age proved to be a valid method, managing to stratify to obtain specific results for each set. The studies cited in the introduction using the same dataset have similar results, as expected. The 18 table presents the results of these works (8.6.1), and the table 19 presents the values found for easy comparison (8.6.2) .

Table 18: Comparison of other works

Author	Year	Algorithm	RMSE
Pierobon	2018	Ensemble com 5 algorítimos	4.150
Hameed	2020	Artificial Neural Networks	4.736
Raj	2018	Gradient Boosting Regressor	4.957
Modukuru	2020	Random Forest Regressor	5.080
Alshamiri	2020	Regularized Extreme Learning Machine	5.508
Abban	2016	Support Vector Machines with Radial Basis Function Kernel	6.105

Table 19: Final result

Model	RMSE
3 days	3.310370
7 days	4.361987
14 days	4.620515
28 days	4.716698
56 days	5.939163
100 days	5.851088

Following the line of reasoning of this work, it can be performed with different algorithms, the results found here used only one (*Parallel Random Forest*), even though it was theoretically the "best" found, other algorithms can present even better results. Another option is to create an *ensemble* of various algorithms, just like Pierobon (2018), but with the separation of age sets proposed here. In addition, it can be performed with a larger dataset, ideally with the same number of samples in each age set, a more homogeneous distribution of compressive strength, and less variance between samples.

5 Literature cited

Abban, Daniel. 2016. "Concrete Compresive Strength." October 2016. https://rpubs.com/brother_abban/220101.

Alshamiri, Tian-Feng e Kim, Abobakr e Yuan. 2020. "Non-Tuned Machine Learning Approach for Predicting the Compressive Strength of High-Performance Concrete." *Materials* 13 (February): 1023. https://doi.org/10.3390/ma13051023.

"Concrete Compressive Strength Data Set." 2008. University of California Irvine. March 2008. https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength.

Fernandez-Delgado, Manuel, E. Cernadas, S. Barro, and Dinani Amorim. 2014. "Do We Need Hundreds of Classifiers to Solve Real World Classification Problems?" *Journal of Machine Learning Research* 15 (October): 3133–81.

Hameed, Mohamed, Mohammed e Khalid. 2020. "Prediction of Compressive Strength of High-Performance Concrete: Hybrid Artificial Intelligence Technique." In, 323–35. https://doi.org/10.1007/978-3-030-38752-5_26.

Hasan, Ahsanul, Md e Kabir. 2011. "Prediction of Compressive Strength of Concrete from Early Age Test Result." In. https://doi.org/10.13140/RG.2.1.3270.7684.

Irizarry, R. A. 2019. *Introduction to Data Science: Data Analysis and Prediction Algorithms with R*. Chapman & Hall/Crc Data Science Series. CRC Press. https://books.google.com.br/books?id=xb29DwAAQBAJ.

Kabir, Md e Miah, Ahsanul e Hasan. 2012. "Predicting 28 Days Compressive Strength of Concrete from 7 Days Test Result," January, 18–22. https://www.researchgate.net/publication/258255513_Predicting_28_Days_Compressive_Strength_of_Concrete_from_7_Days_Test_Result.

Kuhn, Max. 2008. "Building Predictive Models in R Using the Caret Package." *Journal of Statistical Software, Articles* 28 (5). https://doi.org/10.18637/jss.v028.i05.

Kuhn, Max et al. 2020. *Caret: Classification E Regression Training*. https://cran.r-project.org/web/packages/caret/index.html.

"Models Clustered by Tag Similarity." 2020. 2020. http://topepo.github.io/caret/models-clustered-by-tag-similarity.html.

Modukuru, Pranay. 2020. "Concrete Compressive Strength Prediction Using Machine Learning." 2020. https://towardsdatascience.com/concrete-compressive-strength-prediction-using-machine-learning-4a531b3c43f3.

Pierobon, Gabriel. 2018. "A Comprehensive Machine Learning Workflow with Multiple Modelling Using Caret and caretEnsemble in R." September 2018. https://towardsdatascience.com/a-comprehensive-machine-learning-workflow-with-multiple-modelling-using-caret-and-caretensemble-in-fcbf6d80b5f2.

Raj, Pavan. 2018. "Predicting Compressive Strength of Concrete." June 2018. https://www.kaggle.com/pavanraj159/predicting-compressive-strength-of-concrete.

R Core Team. 2020. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

RStudio Team. 2020. *RStudio: Integrated Development Environment for R*. Boston, MA: RStudio, Inc. http://www.rstudio.com/.

Yeh, I-Cheng. 1998. "Modeling of Strength of High-Performance Concrete Using Artificial Neural Networks." Cement and Concrete Research, 28(12), 1797-1808." *Cement and Concrete Research* 28 (December): 1797–1808. https://doi.org/10.1016/S0008-8846(98)00165-3.

6 Appendix 1 - Virtual environment

6.1 Operational system

platform	x86_64-apple-darwin15.6.0
arch	x86_64
os	darwin15.6.0
system	x86_64, darwin15.6.0
status	
major	3
minor	6.2
year	2019
month	12
day	12
svn rev	77560
language	R
version.string	R version 3.6.2 (2019-12-12)
nickname	Dark and Stormy Night

6.2 Packages

caret	6.0.85
cowplot	1.0.0
dplyr	0.8.4
factoextra	1.0.6
gdata	2.18.0
ggplot2	3.2.1
gridExtra	2.3
kableExtra	1.1.0
knitr	1.28
pastecs	1.3.21
purrr	0.3.3
questionr	0.7.0
reshape2	1.4.3
tidyr	1.0.2
tidyverse	1.3.0

7 Appendix 2 - Online repository

https://github.com/pedrobern/concrete-compressive-strength-prediction

8 Appendix 3 - Code

8.1 Obtaining the data

8.1.1 Data download

```
# Data download
url_base <- "https://archive.ics.uci.edu"
url <-  "/ml/machine-learning-databases/concrete/compressive/Concrete_Data.xls"
download.file(paste0(url_base, url), "data.xls")
dat <- read.xls("data.xls")
n_inicial_samples <- nrow(dat)
colnames(dat)
```

8.1.2 Renaming the columns

```
# Renaming the columns
colnames(dat) <- c(
  "cement",
  "blast_furnace_slag",
  "fly_ash",
  "water",
  "superplasticizers",
  "coarse_aggregate",
  "fine_aggregate",
  "day",
  "mpa"
  )
dat$id <- seq.int(nrow(dat))
```

8.1.3 Reordering the data

```
# Reordering the data
col_order <- c(
  "id",
  "cement",
  "blast_furnace_slag",
  "fly_ash",
  "water",
  "superplasticizers",
  "coarse_aggregate",
  "fine_aggregate",
  "day",
  "mpa"
)
dat <- dat[, col_order]
```

8.1.4 Defining column names and units

```r
# Defining column names and units
colNames <- c("ID", "Cement", "B.F.S.", "Fly ash", "Water",
              "Superp.", "C.Aggregate", "F.Aggregate", "Day", "Comp.Str.")
dfUnits <- c("", "$kg/m^3$", "$kg/m^3$","$kg/m^3$","$kg/m^3$",
              "$kg/m^3$","$kg/m^3$","$kg/m^3$","","$MPa$")
```

8.1.5 Table - First samples

```r
# Table - First samples
caption <- "First 6 samples"
kable(
    dat[1:6,],
    col.names = dfUnits,
    escape = F,
    booktabs = T,
    caption = caption,
    linesep = "\\addlinespace",
    align = "c"
    ) %>%
    add_header_above(header = colNames, line = F, align = "c") %>%
    kable_styling(latex_options = c("HOLD_position", "scale_down"))
```

8.2 Data preparation

8.2.1 Removing duplicated samples

```r
# Removing duplicated samples
n_distinct_samples <- dat %>% select(-c(id)) %>% n_distinct()
n_duplicated_samples <- n_inicial_samples - n_distinct_samples
dat <- dat[!duplicated(select(dat, -c(id))),]
n_samples <- nrow(dat)
```

8.2.2 Table - Samples with same composition

```r
# Table - Samples with same composition
same_samples <- dat %>%
  filter(id %in% c(653, 678, 654, 681))
caption <- "Samples with same composition"
kable(
  same_samples[order(same_samples$day),],
  col.names = dfUnits,
  escape = F,
  booktabs = T,
  caption = caption,
  linesep = "\\addlinespace",
  align = "c",
  row.names = FALSE
  ) %>%
  add_header_above(header = colNames, line = F, align = "c") %>%
  kable_styling(latex_options = c("HOLD_position", "scale_down"))
```

8.2.3 Table - Same samples with different results

```
# Table - Same samples with different results
same_samples_2 <- dat %>%
  filter(id %in% c(472, 473, 474))
caption <- "Same samples with different results"
kable(
  same_samples_2[order(same_samples_2$day),],
  col.names = dfUnits,
  escape = F,
  booktabs = T,
  caption = caption,
  linesep = "\\addlinespace",
  align = "c",
  row.names = FALSE
  ) %>%
  add_header_above(header = colNames, line = F, align = "c") %>%
  kable_styling(latex_options = c("scale_down"))
```

8.2.4 Data cleaning

```
# Data cleaning
dat <- dat %>%
  group_by(
    cement,
    blast_furnace_slag,
    fly_ash,
    water,
    superplasticizers,
    coarse_aggregate,
    fine_aggregate,
  ) %>%
  filter("28" %in% day) %>%
  mutate(id = id[which.min(id)]) %>%
  ungroup() %>%
  group_by(id, day) %>%
  mutate(mpa = mean(mpa)) %>%
  ungroup()
dat <- dat[!duplicated(select(dat, -c(id))),]
dat$id<-factor(dat$id)
n_samples <- nrow(dat)
n_distinct_samples <- n_distinct(dat$id)
```

8.2.5 Table - Previous samples after processing

```
# Table - Previous samples after processing
same_samples <- dat %>%
  filter(id == 653 | id == 472 & day == 28)
caption <- "Previous samples after processing"
kable(
  same_samples[order(same_samples$id, same_samples$day),],
```

```
  col.names = dfUnits,
  escape = F,
  booktabs = T,
  caption = caption,
  linesep = "\\addlinespace",
  align = "c",
  row.names = FALSE,
  digits = 2
  ) %>%
  add_header_above(header = colNames, line = F, align = "c") %>%
  kable_styling(latex_options = c("HOLD_position", "scale_down"))
```

8.2.6 Figure - Compressive strength (MPa) vs age (days)

```
# Figure - Compressive strength (MPa) vs age (days)
cap <- "Boxplot - Compressive strength (MPa) vs age (days)"
ylabel <- "Compressive strength (MPa)"
xlabel <- "Age (days)"
dat %>%
  ggplot(aes(x=factor(day), y=mpa)) +
  geom_boxplot() +
  geom_jitter(alpha=0.2)  +
  theme_bw() +
  ylab(ylabel) +
  xlab(xlabel)
```

8.2.7 Figure - Principal component analysis - 90, 91 e 100 days

```
# Figure - Principal component analysis - 90, 91 e 100 days
dat_90_91_100 <- dat %>%
  ungroup() %>%
  filter(day %in% c(90, 91, 100)) %>%
  select(-c(id, mpa))
cap <- "Principal component analysis - 90, 91 e 100 days"
colnames(dat_90_91_100) <- c(
   "Cem.", "B.F.S.", "Fly.A.","Water","Sup.","C.Ag.","F.Ag.","day")
pca <- prcomp(select(dat_90_91_100, -c(day)), scale = TRUE)
habillage <- dat_90_91_100$day
fviz_pca_biplot(
  pca,
  geom.ind = "point",
  habillage=habillage,
  addEllipses = TRUE,
  ellipse.level=0.75) +
  ggtitle("") +
  theme_bw() +
  coord_cartesian(xlim = c(-3, 3.5), ylim = c(3, -5))
```

8.2.8 Figure - Compressive strength through time

```
# Figure - Compressive strength through time
cap <- "Compressive strength through time"
ylabel <- "MPa"
xlabel <- "Age (days)"
dat_duplicated_only <- dat %>%
  group_by(id) %>%
  filter(n()>5) %>%
  select(id, mpa, day)
dat_duplicated_only %>%
  ggplot(aes(day, mpa, fill=id, alpha = 0.5)) +
  geom_line() +
  xlab(xlabel) +
  ylab(ylabel) +
  theme_bw() +
  theme(legend.position = "none")
```

8.2.9 Joining 90, 91 and 100 days data

```
# Joining 90, 91 and 100 days data
ind_90 <- dat$id[which(dat$day == "90")]
ind_91 <- dat$id[which(dat$day == "91")]
ind_100 <- dat$id[which(dat$day == "100")]
sum_duplicated <- sum(duplicated(c(ind_90, ind_91, ind_100))) # 0
dat <- dat %>%
  ungroup() %>%
  mutate(day = ifelse(day %in% c(91, 90), 100, day))
```

8.2.10 Figure - Ages frequency

```
# Figure - Ages frequency
cap <- "Ages frequency"
ylabel <- "Frequency"
xlabel <- "Age (days)"
dat %>%
  ggplot(aes(x = factor(day))) +
  geom_bar() +
  theme_bw() +
  xlab(xlabel) +
  ylab(ylabel)
```

8.2.11 Removing ages with frequency lower than 50

```
# Removing ages with frequency lower than 50
dat <- dat[dat$day %in% c(3, 7, 14, 28, 56, 100),]
```

8.2.12 Data reorganization

```r
# Data reorganization
dat <- dat %>%
  group_by_at(vars(-mpa)) %>%
  mutate(row_id = 1:n()) %>% ungroup() %>%
  spread(day, mpa, sep = "_") %>%
  select(-row_id)
```

8.2.13 Table - First 6 samples after reorganization

```r
# Table - First 6 samples after reorganization
caption <- "First 6 samples after reorganization"
colNames2 = c("ID", "Cement", "B.F.S", "Fly ash", "Water",
              "Superp.", "Coarse Ag.", "Fine Ag.", "3 days", "7 days",
              "14 days", "28 days", "56 days", "100 days")
dfUnits2 <- c("", "$kg/m^3$", "$kg/m^3$","$kg/m^3$","$kg/m^3$",
               "$kg/m^3$", "$kg/m^3$","$kg/m^3$","$MPa$","$MPa$",
               "$MPa$","$MPa$","$MPa$","$MPa$")
kable(
    head(dat[order(dat$id),]),
    col.names = dfUnits2,
    escape = F,
    booktabs = T,
    caption = caption,
    linesep = "\\addlinespace",
    align = "c"
    ) %>%
    add_header_above(header = colNames2, line = F, align = "c") %>%
    kable_styling(latex_options = c("HOLD_position", "scale_down"))
```

8.2.14 Total samples

```r
# Total samples
n_samples <- nrow(dat) # 416
n_distinct_samples <- n_distinct(dat$id) # 416
```

8.2.15 Adding New features

```r
# Adding New features
concrete_class <- function(mpa){
  if (mpa >= 10) {
    s <- as.character(mpa)
    first <- substr(s, start = 1, stop = 1)
    second <- ifelse(substr(s, start = 2, stop = 2) >= 5, 5, 0)
  }
  else {
    first <- ""
    second <- "5"
  }
  paste("C", first, second, sep = "")
}
mix <- function(c, f_ag, c_ag){
```

```
  paste(
    1,
    round(f_ag/c, 0),
    round(c_ag/c, 0),
    sep = ":")
}
dat <- dat %>%
  mutate(class = sapply(day_28, concrete_class)) %>%
  mutate(class = as.factor(class)) %>%
  mutate(mix_app = factor(
    mix(cement, fine_aggregate, coarse_aggregate))) %>%
  mutate(`water_/_cement` = water / cement) %>%
  mutate(`fine_aggregate_/_cement` = fine_aggregate/cement) %>%
  mutate(`coarse_aggregate_/_cement` = coarse_aggregate/cement) %>%
  mutate(`fine_aggregate_/_coarse_aggregate` = fine_aggregate/coarse_aggregate) %>%
  mutate(`water_/_coarse_aggregate` = water/coarse_aggregate) %>%
  mutate(`water_/_fine_aggregate` = water/fine_aggregate)
lvl <- levels(dat$class)
dat$class <- factor(
  dat$class,
  levels=c( "C5", sort(lvl[lvl!="C5"], decreasing=F)))
```

8.2.16 Table - New features

```
# Table - New features
caption <- "New features"
colNames7 = c("ID", "Class","Approximated Mix",
              "Water / Cement", "Fine Ag. / Cement",
              "Coarse Ag. / Cement", "Fine Ag. / Coarse Ag.",
              "Water / Coarse Ag.", "Water / Fine Ag.")
kable(
    head(dat[order(dat$id),][,c(1,15:22)]),
    col.names = colNames7,
    escape = F,
    booktabs = T,
    caption = caption,
    linesep = "\\addlinespace",
    align = "c",
    digits = 4
    ) %>%
  kable_styling(latex_options = c("HOLD_position", "scale_down")) %>%
  column_spec(2, width = "1.5cm") %>%
  column_spec(3, width = "2cm") %>%
  column_spec(4:9, width = "1.7cm")
```

8.3 Data visualization

8.3.1 Table - Descriptive statistics - continuous variables

```
# Table - Descriptive statistics - continuous variables
summ <- t(
  stat.desc(select(dat, -c(id, class, mix_app))))
```

```
caption <- "Descriptive statistics - continuous variables"
colnames(summ) <- c("Samples", "Null", "NA", "Min", "Max", "Range",
                    "Sum", "Median", "Mean", "SE mean",
                    "CI mean","Variance", "Std.Dev.", "Coef.Var")
rownames(summ) = c("Cement", "B.F.S.", "Fly ash", "Water",
                   "Superplast.", "Coarse agg.", "Fine agg.", "3 days", "7 days",
                   "14 days", "28 days", "56 days", "100 days",
                   "Water / Cement", "Fine agg. / Cement",
                   "Coarse agg. / Cement", "Fine agg. / Coarse agg.",
                   "Water / Coarse agg.", "Water / Fine agg.")
kable(
  summ,
  escape = F,
  booktabs = T,
  caption = caption,
  linesep = "\\addlinespace",
  align = "c",
  digits = c(0,0,0,2,2,2,2,2,2,2,2,2,2,2)
  ) %>%
  kable_styling(latex_options = c("HOLD_position", "scale_down")) %>%
  column_spec(c(1,10:15), width = "1.5cm")
```

8.3.2 Figure - Descriptive statistics - categorical variables

```
# Figure - Descriptive statistics - categorical variables
cap <- "Descriptive statistics - categorical variables"
name1 <- "Percentage"
name2 <- "Accumulated percentage"
ylabel <- "Frequency"
xlabel1 <- "Class"
xlabel2 <- "Approximate Mix"
format_percent = function(n){
  paste(n, "%", sep = "")
}
format_class <- function(cls){
  str_remove_all(cls, "C")
}
f_class <- freq(dat$class, cum = TRUE, sort = "dec", total = F) %>%
  select(n, "%", "%cum") %>%
  mutate(class = row.names(.)) %>%
  mutate(class_n = as.numeric(format_class(class)))
f_cls_labels = function(n) {
  f_class$class[n]
}
f_cls_acc_labels = function(n){
  paste(f_class$`%cum`[n], "%", sep = "")
}
p1 <- f_class %>%
  ggplot(aes(x = as.integer(reorder(class_n, -n)), y = n)) +
  geom_bar(stat = 'identity') +
  scale_y_continuous(
      sec.axis = sec_axis(~./length(dat$class) * 100,
                          name = name1,
```

```
                          labels = format_percent)) +
  scale_x_continuous(labels = f_cls_labels, breaks = 1:16, limits = c(0.5,16.5),
                     sec.axis = sec_axis(~., breaks = 1:16,
                                         name = name2,
                                         labels = f_cls_acc_labels)) +
  theme_bw() +
  theme(panel.grid.minor.y = element_blank()) +
  xlab(xlabel1) +
  ylab(ylabel) +
  coord_flip()
format_mix <- function(mix){
  str_remove_all(mix, ":")
}
f_mix <- freq(dat$mix_app, cum = TRUE, sort = "dec", total = F) %>%
  select(n, "%", "%cum") %>%
  mutate(mix = row.names(.)) %>%
  mutate(mix_n = as.numeric(format_mix(mix)))
f_mix_labels = function(n) {
  f_mix$mix[n]
}
f_mix_acc_labels = function(n){
  paste(f_mix$`%cum`[n], "%", sep = "")
}
p2 <- f_mix %>%
  ggplot(aes(x = as.integer(reorder(mix_n, -n)), y = n)) +
  geom_bar(stat = 'identity') +
  scale_y_continuous(
      sec.axis = sec_axis(~./length(dat$mix_app) * 100,
                          name = name1,
                          labels = format_percent)) +
  scale_x_continuous(labels = f_mix_labels, breaks = 1:24, limits = c(0.5,24.5),
                     sec.axis = sec_axis(~., breaks = 1:24,
                                         name = name2,
                                         labels = f_mix_acc_labels)) +
  theme_bw() +
  theme(panel.grid.minor.y = element_blank()) +
  xlab(xlabel2) +
  ylab(ylabel) +
  coord_flip()
grid.arrange(p1, p2, ncol=2)
```

8.3.3 Figure - Correlation grouped by age

```
# Figure - Correlation grouped by age
cor_dat <- dat %>% select(-c(id))
cap <- "Correlations at each age"
f_lvl <- c("3 days", "7 days", "14 days","28 days", "56 days", "100 days")
name <- "Correlation"
colnames_dat <- c("Cement", "B.F.S.", "Fly ash", "Water",
                  "Superplast.", "Coarse agg.", "Fine agg.",
                  "3", "7", "14", "28", "56", "100",
                  "class", "mix_app", "W./C.", "F.A./C.",
                  "C.A./C.", "F.A./C.A.",
```

```
                "W./C.A.", "W./F.A.")
colnames(cor_dat) <- colnames_dat
cor_dat <- cor_dat %>%
  gather("day", "mpa", c("3", "7", "14", "28", "56", "100")) %>%
  drop_na()
cor_day <- function(d){
  res <- cor_dat %>%
    filter(day == d) %>%
    select(-c(day, class, mix_app)) %>%
    cor(.)
  res[upper.tri(res)] <- NA
  return(res)
}
cor_dats <- list(cor_day(3), cor_day(7), cor_day(14),
                 cor_day(28), cor_day(56), cor_day(100))
melt_day <- function(df, d){
  df %>%
    melt() %>%
    mutate(day = d)
}
melt_dats <- list(melt_day(cor_dats[1], 3), melt_day(cor_dats[2], 7),
                  melt_day(cor_dats[3], 14), melt_day(cor_dats[4],28),
                  melt_day(cor_dats[5], 56), melt_day(cor_dats[6],100))
melt_dat_final <- melt_dats %>%
  reduce(rbind) %>%
  filter(value != 1)
melt_dat_final$day <- factor(melt_dat_final$day)
levels(melt_dat_final$day) <- f_lvl
melt_dat_final %>%
  ggplot(aes(x=reorder(Var1, desc(Var1)), y=Var2, fill=value)) +
  geom_tile(color = "white") +
  facet_wrap(~day, ncol=3) +
  scale_fill_gradient2(low = "red", high = "blue", mid = "white",
   midpoint = 0, limit = c(-1,1),name=name, na.value="white") +
  xlab("") +
  ylab("") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  theme(panel.border = element_blank(), panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"))
```

8.3.4 Figure - Correlation over time

```
# Figure - Correlation over time
cap <- "Correlation of variables with compressive strength over time"
name <- "Correlation"
melt_dat_final %>%
  filter(Var1 == "mpa" | Var2 == "mpa") %>%
  ggplot(aes(x=reorder(Var1, desc(Var1)), y=Var2, fill=value)) +
  geom_tile(color = "white") +
  facet_wrap(~day, ncol=6) +
  scale_fill_gradient2(low = "red", high = "blue", mid = "white",
```

```
   midpoint = 0, limit = c(-1,1),name=name, na.value="white") +
  xlab("") +
  ylab("") +
  geom_text(aes(label = round(value, 2)), size = 2.5) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  theme(panel.border = element_blank(), panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"))
```

8.3.5 Figure - Relationship between approximated mix, water, MPa and age

```
# Figure - Relationship between approximated mix, water, MPa and age
cap <- "Relationship between approximated mix, water, MPa and age"
d <- " days"
xlabel <- "Approximated mix"
ylabel <- "Compressive strength (MPa)"
label <- "Water /\nCement"
mix_dat <- dat %>%
  select(c(day_3,day_7,day_14,day_28,day_56,day_100,
           mix_app, `water_/_cement`,
           `fine_aggregate_/_cement`,
           `coarse_aggregate_/_cement`))

labs <- paste(c("3","7","14","28","56","100"), d, sep="")
mix_dat <- mix_dat %>%
  gather("day", "mpa", -c(mix_app, `water_/_cement`,
                          `fine_aggregate_/_cement`,
                          `coarse_aggregate_/_cement`)) %>%
  drop_na()
lvls <- paste("day_",c("3","7","14","28","56","100"), sep="")
mix_dat$day <- factor(mix_dat$day, levels=lvls)
levels(mix_dat$day) <- labs
min_x <- min(mix_dat$`water_/_cement`)
max_x <- max(mix_dat$`water_/_cement`)
s_x <- max_x - min_x
mix_dat %>%
  ggplot(aes(x=mix_app, y=mpa, colour = `water_/_cement`)) +
  geom_point()  +
  facet_wrap(~ day, ncol=2) +
  theme_bw() +
  ylab(ylabel) +
  xlab(xlabel) +
  scale_shape_manual(values=c(16, 2, 8)) +
  scale_colour_gradient2(low = "red", mid = "yellow", high = "blue",
                         midpoint = s_x / 2 + min_x ,limits = c(min_x, max_x),
                         breaks = c(round(min_x, 2),
                                    round(s_x*0.25 + min_x,2),
                                    round(s_x*0.5 + min_x,2),
                                    round(s_x * 0.75 + min_x,2),
                                    round(max_x, 2))) +
  labs(colour = label) +
  theme_minimal() +
```

```
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  theme(panel.grid.minor = element_blank(),
      axis.line = element_line(colour = "black"))
```

8.3.6 Figure - Relationship between concrete main features

```
# Figure - Relationship between concrete main features
cap <- "Relationship between concrete main features"
d <- " days"
xlabel <- "Water / Cement"
ylabel <- "Compressive strength (MPa)"
colour <- "Fine Agg. /\nCement"
size <- "Coarse Agg. /\nCement"
min_x_2 <- min(mix_dat$`fine_aggregate_/_cement`)
max_x_2 <- max(mix_dat$`fine_aggregate_/_cement`)
s_x_2 <- max_x_2 - min_x_2
mix_dat %>%
  ggplot(aes(x=`water_/_cement`, y=mpa,
             colour = `fine_aggregate_/_cement`,
             size = `coarse_aggregate_/_cement`)) +
  geom_point(alpha = 0.5)  +
  facet_wrap(~ day, ncol=2) +
  theme_bw() +
  ylab(ylabel) +
  xlab(xlabel) +
  scale_colour_gradient2(low = "red", mid = "yellow", high = "blue",
                         midpoint = (s_x_2 / 2) + min_x_2,
                         limits = c(min_x_2, max_x_2),
                         breaks = c(round(min_x_2, 2),
                                    round(s_x_2*0.25 + min_x_2,2),
                                    round(s_x_2*0.5 + min_x_2,2),
                                    round(s_x_2 * 0.75 + min_x_2,2),
                                    round(max_x_2 - 0.01, 2))
                         ) +
  labs(colour = colour, size = size) +
  ylim(c(-5,85)) +
  scale_radius() +
  theme_minimal() +
  theme(panel.grid.minor = element_blank(),
      axis.line = element_line(colour = "black"))
```

8.3.7 Figure - Variables distribution

```
# Figure - Variables distribution

cap <- "Variables distribution"
colnames_dat <- c("Cement", "B.F.S.", "Fly ash", "Water",
                  "Superplast.", "Coarse agg.", "Fine agg.",
                  "Water / Cement", "Fine agg. / Cement",
                  "Coarse agg. / Cement", "Fine agg. / Coarse agg.",
                  "Water / Coarse agg.", "Water / Fine agg.")
dist_dat <- dat %>%
```

```
  select(-c(id, class, day_3, day_7, day_14, day_56, day_100, mix_app))

colnames(dist_dat) <- colnames_dat
dist_dat <- dist_dat %>%
  gather("Var", "value") %>%
  mutate(value = as.numeric(value))
dist_dat %>%
  ggplot(aes(value)) +
  geom_histogram(aes(y = ..density..),
                 colour = "black",
                 fill = "white") +
  geom_density(alpha = .5, fill = "lightseagreen") +
  facet_wrap(~ Var, ncol=3, scale = "free") +
  theme_minimal() +
  xlab("") +
  ylab("") +
  theme(panel.border = element_blank(), panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"))
```

8.3.8 Figure - Variables distribution grouped by age

```
# Figure - Variables distribution grouped by age

days_labs <- c("3","7","14","28","56","100")
cap <- "Variables distribution grouped by age"
colnames_dat <- c("Cement", "B.F.S.", "Fly ash", "Water",
                  "Superplast.", "Coarse agg.", "Fine agg.",
                  days_labs,
                  "Water / Cement", "Fine agg. / Cement",
                  "Coarse agg. / Cement", "Fine agg. / Coarse agg.",
                  "Water / Coarse agg.", "Water / Fine agg.")
dist_dat_2 <- dat %>%
  select(-c(id, class, mix_app))
colnames(dist_dat_2) <- colnames_dat
dist_dat_2 <- dist_dat_2 %>%
  gather("day", "mpa", days_labs) %>%
  drop_na() %>%
  gather("Var", "value", -c("day")) %>%
  mutate(day = factor(day, levels = days_labs))
dist_dat_2 %>%
  ggplot(aes(x = day, y = value)) +
  geom_violin(color = NA,
                fill = "lightseagreen",
                alpha = .5,
                na.rm = TRUE,
                scale = "count") +
  geom_boxplot(alpha = 0.2) +
  facet_wrap(~ Var, ncol=3, scale = "free") +
  theme_minimal() +
  theme(panel.border = element_blank(), panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black")) +
```

```
  xlab("") +
  ylab("")
```

8.3.9 Figure - Principal component analysis on ingredients

```
# Figure - Principal component analysis on ingredients

cap <- "Principal component analysis on ingredients"
colnames_dat <- c(
  "Cem.", "B.F.S.", "F.A.", "Wat.", "Sup.", "C.Agg.",
  "F.Agg.","WxC", "F.Agg.xC","C.Agg.xC","F.Agg.xC",
  "WxC.Agg.","WxF.Agg."
)
class_list <- list(low="0", normal="20", medium="40", high="70")
dat_pca <- dat %>%
  select(-c(id, class, mix_app,
            "day_3", "day_7", "day_14", "day_28", "day_56", "day_100"))
colnames(dat_pca) <- colnames_dat
class_2 <- dat$day_28
class_2[class_2 < 20] = "0"
class_2[class_2 >= 20 & class_2 < 40] = "20"
class_2[class_2 >= 40 & class_2 < 70] = "40"
class_2[class_2 >= 70] = "70"
class_2 <- factor(class_2)
levels(class_2) <- class_list
pca <- prcomp(dat_pca, scale = TRUE)
fviz_pca_ind(
  pca,
  geom.ind = "point",
  habillage=class_2,
  addEllipses = T,
  ellipse.level=0.95) +
  ggtitle("") +
  theme_bw()
```

8.4 Machine learning models

8.4.1 Dummy variables

```
# Dummy variables
dummies <- dummyVars( ~ mix_app, data = dat)
dummyDat <- data.frame(predict(dummies, newdata = dat))
dummyDat$id <- dat$id
dat <- dat %>%
  select(-c(mix_app)) %>%
  full_join(., dummyDat)
```

8.4.2 Data preparation

```r
# Data preparation
names(dat) <- gsub(x = names(dat), pattern = "/", replacement = ".")
dat_3 <- dat %>%
  select(-c("day_7", "day_14", "day_28", "day_56", "day_100")) %>%
  drop_na() %>%
  rename_at("day_3",~"mpa")
dat_7 <- dat %>%
  select(-c("day_3", "day_14", "day_28", "day_56", "day_100")) %>%
  drop_na() %>%
  rename_at("day_7",~"mpa")
dat_14 <- dat %>%
  select(-c("day_3", "day_7", "day_28", "day_56", "day_100")) %>%
  drop_na() %>%
  rename_at("day_14",~"mpa")
dat_28 <- dat %>%
  select(-c("day_3", "day_7", "day_14", "day_56", "day_100")) %>%
  drop_na() %>%
  rename_at("day_28",~"mpa")
dat_56 <- dat %>%
  select(-c("day_3", "day_7", "day_14", "day_28", "day_100")) %>%
  drop_na() %>%
  rename_at("day_56",~"mpa")
dat_100 <- dat %>%
  select(-c("day_3", "day_7", "day_14", "day_28", "day_56")) %>%
  drop_na() %>%
  rename_at("day_100",~"mpa")
```

8.4.3 Table - First 18 columns of the 6 first samples of 28 days

```r
# Table - First 18 columns of the 6 first samples of 28 days
colNames = c("ID", "Cement", "B.F.S.", "Fly ash", "water",
              "Superp.", "Coarse Agg.", "Fine Agg.", "MPa", "Class",
              "Wat./", "F.Agg./", "C.Agg./",
              "F.Agg./","Wat./", "Wat./", "App Mix" = 2)
dfUnits <- c("", "$kg/m^3$", "$kg/m^3$","$kg/m^3$","$kg/m^3$",
              "$kg/m^3$", "$kg/m^3$","$kg/m^3$","$MPa$","", "Ci.",
              "Ce.", "Ce.", "C.Agg.", "C.Agg.", "F.Agg.", "1:1:2", "1:2:2")
caption <- "First 18 columns of the 6 first samples of 28 days"
dat_table <- dat_28[1:18]
kable(
    head(dat_table[order(dat_table$id),]),
    col.names = dfUnits,
    escape = F,
    booktabs = T,
    caption = caption,
    linesep = "\\addlinespace",
    digits = 2,
    align = "c"
    ) %>%
    add_header_above(header = colNames, line = F, align = "c") %>%
    kable_styling(latex_options = c("HOLD_position", "scale_down"))
```

8.4.4 Removing near zero variance columns

```
# Removing near zero variance columns
nzv_3 <- nearZeroVar(dat_3)
nzv_7 <- nearZeroVar(dat_7)
nzv_14 <- nearZeroVar(dat_14)
nzv_28 <- nearZeroVar(dat_28)
nzv_56 <- nearZeroVar(dat_56)
nzv_100 <- nearZeroVar(dat_100)
dat_3 <- dat_3[,-nzv_3]
dat_7 <- dat_7[,-nzv_7]
dat_14 <- dat_14[,-nzv_14]
dat_28 <- dat_28[,-nzv_28]
dat_56 <- dat_56[,-nzv_56]
dat_100 <- dat_100[,-nzv_100]
```

8.4.5 High correlation variables verification

```
# High correlation variables verification
descr_cor_3 <- cor(select(dat_3, -c(mpa, id, class)))
descr_cor_7 <- cor(select(dat_7, -c(mpa, id, class)))
descr_cor_14 <- cor(select(dat_14, -c(mpa, id, class)))
descr_cor_28 <- cor(select(dat_28, -c(mpa, id, class)))
descr_cor_56 <- cor(select(dat_56, -c(mpa, id, class)))
descr_cor_100 <- cor(select(dat_100, -c(mpa, id, class)))
high_cor_3 <- sum(abs(descr_cor_3[upper.tri(descr_cor_3)]) > .999)
high_cor_7 <- sum(abs(descr_cor_7[upper.tri(descr_cor_7)]) > .999)
high_cor_14 <- sum(abs(descr_cor_14[upper.tri(descr_cor_14)]) > .999)
high_cor_28 <- sum(abs(descr_cor_28[upper.tri(descr_cor_28)]) > .999)
high_cor_56 <- sum(abs(descr_cor_56[upper.tri(descr_cor_56)]) > .999)
high_cor_100 <- sum(abs(descr_cor_100[upper.tri(descr_cor_100)]) > .999)
```

8.4.6 Test and train data split

```
# Test and train data split
reg <- list(
  dat_3 %>% select(-c(mpa, class, id)) %>% mutate(y = dat_3$mpa),
  dat_7 %>% select(-c(mpa, class, id)) %>% mutate(y = dat_7$mpa),
  dat_14 %>% select(-c(mpa, class, id)) %>% mutate(y = dat_14$mpa),
  dat_28 %>% select(-c(mpa, class, id)) %>% mutate(y = dat_28$mpa),
  dat_56 %>% select(-c(mpa, class, id)) %>% mutate(y = dat_56$mpa),
  dat_100 %>% select(-c(mpa, class, id)) %>% mutate(y = dat_100$mpa)
)
reg_seed <- c(
  1111, # 3
  1, # 7
  22, # 14
  11111, # 28
  111, # 56
  11 # 100
  )
```

```
split_reg <- function(n){
  set.seed(reg_seed[[n]], sample.kind="Rounding")
  createDataPartition(reg[[n]]$y, p = .8, list = F)
}
trainIndex_reg <- lapply(list(1,2,3,4,5,6), split_reg)
gen_dat <- function(n){
  regIndex <- trainIndex_reg[[n]]
  list(train = reg[[n]][regIndex,], test = reg[[n]][-regIndex,])
}
dats_reg <- lapply(list(1,2,3,4,5,6), gen_dat)
names(dats_reg) <- c("d3", "d7", "d14", "d28", "d56", "d100")
```

8.4.7 Distribution of test and train data

```
# Distribution of test and train data
cap <- "Distribution of test and train data"
d_lab <- " days"
ylabel <- "Density"
g1 <- "Train"
g2 <- "Test"
dens <- function(d, n){
  dens <- full_join(
      d$train %>%
        select(y) %>%
        mutate(Group = g1, day = str_sub(n, 2)),
      d$test %>%
        select(y) %>%
        mutate(Group = g2, day = str_sub(n, 2))
    )
  dens
}
densities <- imap(dats_reg, dens) %>%
  reduce(rbind) %>%
  mutate(day_f = factor(day, levels=c('3','7','14','28', '56', '100')))
labs <- paste(c("3","7","14","28","56","100"), d_lab, sep="")
levels(densities$day_f) <- labs
densities %>%
  ggplot(aes(y, fill = Group)) +
  geom_density(alpha = 0.5) +
  facet_wrap(~day_f, ncol=3) +
  xlab("MPa") +
  ylab(ylabel) +
  theme_bw() +
  theme_minimal() +
  theme(panel.border = element_blank(), panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"))
```

8.4.8 Naive model

```
# Naive model
res <- function(d, n){
  data.frame(
    day = str_sub(n, 2),
    mean_mpa = mean(d$train$y)
    )
}
ing_model <- imap(dats_reg, res)
get_rmse <- function(d, n){
   data.frame(
    rmse_train = RMSE(d$train$y, ing_model[[n]]$mean_mpa),
    rmse_test =  RMSE(d$test$y, ing_model[[n]]$mean_mpa),
    day = str_sub(n, 2)
    )
}
rmses <- imap(dats_reg, get_rmse) %>%
  reduce(rbind)
ing_model_df <- ing_model %>% reduce(rbind)
df_performance_reg <- full_join(ing_model_df, rmses)
```

8.4.9 Table - Naive models

```
# Table - Naive models
colNames = c("Age", "Mean $MPa$ (train)", "RMSE (train)", "RMSE (test)")
caption <- "Naive models"
kable(
    df_performance_reg,
    col.names = colNames,
    escape = F,
    booktabs = T,
    caption = caption,
    linesep = "\\addlinespace",
    align = "c"
    )  %>%
  kable_styling(latex_options = c("HOLD_position"))
```

8.4.10 Features selection

```
# Features selection
data1 <- dats_reg$d28$train # full
data2 <- dats_reg$d28$train[c(1:13, 21)] # no dummy vars
data3 <- dats_reg$d28$train[c(1:7, 21)] # only original variables
data4 <- dats_reg$d28$train[c(8:21)] # only new variables
data5 <- dats_reg$d28$train[c(8:13, 21)] # only new varibels, no dummy vars
test1 <- dats_reg$d28$test # full
test2 <- dats_reg$d28$test[c(1:13, 21)] # no dummy vars
test3 <- dats_reg$d28$test[c(1:7, 21)] # only original variables
test4 <- dats_reg$d28$test[c(8:21)] # only new variables
test5 <- dats_reg$d28$test[c(8:13, 21)] # only new varibels, no dummy vars
# parRF
modelLookup("parRF")
control_parRF <- trainControl(method='repeatedcv',number=10,repeats=5,search='grid')
```

```
fit_parRF<- function(data, n) {
  set.seed(1, sample.kind = "Rounding")
  tunegrid_parRF<- expand.grid(mtry = seq(1, length(data), n))
  train(y ~ .,
    data = data,
    preProcess = c("center","scale"),
    method='parRF',
    tuneGrid=tunegrid_parRF,
    trControl=control_parRF)
}
fit_parRF1 <- fit_parRF(data1, 3)
ggplot(fit_parRF1)
min(fit_parRF1$results$RMSE) # 6.26108
p1_parRF1 <- predict(fit_parRF1, newdata = test1)
RMSE(p1_parRF1, test1$y) # 4.887264
fit_parRF2 <- fit_parRF(data2, 1)
ggplot(fit_parRF2)
min(fit_parRF2$results$RMSE) # 6.268789
p_parRF2 <- predict(fit_parRF2, newdata = test2)
RMSE(p_parRF2, test2$y) # 4.812994
fit_parRF3 <- fit_parRF(data3, 1)
ggplot(fit_parRF3)
min(fit_parRF3$results$RMSE) # 6.361905
p_parRF3 <- predict(fit_parRF3, newdata = test3)
RMSE(p_parRF3, test3$y) # 5.083467
fit_parRF4 <- fit_parRF(data4, 1)
ggplot(fit_parRF4)
min(fit_parRF4$results$RMSE) # 8.705984
p_parRF4 <- predict(fit_parRF4, newdata = test4)
RMSE(p_parRF4, test4$y) # 8.083962
fit_parRF5 <- fit_parRF(data5, 1)
ggplot(fit_parRF5)
min(fit_parRF5$results$RMSE) # 8.691755
p_parRF5 <- predict(fit_parRF5, newdata = test5)
RMSE(p_parRF5, test5$y) # 7.991537
```

8.4.11 Table - First 6 samples of train data of the 28 days model

```
# Table - First 6 samples of train data of the 28 days model
caption <- "First 6 samples of train data of the 28 days model"
colNames = c("Cement", "B.F.S.", "Fly ash", "water",
             "Superp.", "Coarse Agg.", "Fine Agg.",
             "Wat./", "F.Agg./", "C.Agg./",
             "F.Agg./","Wat./", "Wat./", "y")
dfUnits <- c("$kg/m^3$", "$kg/m^3$","$kg/m^3$","$kg/m^3$",
             "$kg/m^3$", "$kg/m^3$","$kg/m^3$","Ce.",
             "Ce.", "Ce.", "C.Agg.", "C.Agg.", "F.Agg.", "MPa")
colNames2 = c("Features"=13, "Outcome"=1)
kable(
    head(data2),
    col.names = dfUnits,
    escape = F,
    booktabs = T,
```

```
    caption = caption,
    linesep = "\\addlinespace",
    digits = 2,
    align = "c"
    ) %>%
    add_header_above(header = colNames, line = F, align = "c") %>%
    add_header_above(header = colNames2, line = T, align = "c") %>%
    kable_styling(latex_options = c("HOLD_position", "scale_down"))
```

8.4.12 Regression models

```
# Regression models
# no dummy vars:
data3 <- dats_reg$d3$train[c(1:13, 22)]
data7 <- dats_reg$d7$train[c(1:12, 19)]
data14 <- dats_reg$d14$train[c(1:13, 21)]
data28 <- dats_reg$d28$train[c(1:13, 21)]
data56 <- dats_reg$d56$train[c(1:13, 22)]
data100 <- dats_reg$d100$train[c(1:13, 23)]
test3 <- dats_reg$d3$test[c(1:13, 22)]
test7 <- dats_reg$d7$test[c(1:12, 19)]
test14 <- dats_reg$d14$test[c(1:13, 21)]
test28 <- dats_reg$d28$test[c(1:13, 21)]
test56 <- dats_reg$d56$test[c(1:13, 22)]
test100 <- dats_reg$d100$test[c(1:13, 23)]
get_trControl <- function(n, r){
  trainControl(method='repeatedcv',number=n,repeats=r,search='grid')
}
trControl3 <- get_trControl(30, 10)
trControl7 <- get_trControl(10, 10)
trControl14 <- get_trControl(30, 10)
trControl28 <- get_trControl(30, 10)
trControl56 <- get_trControl(30, 10)
trControl100 <- get_trControl(10, 10)
get_tuneGrid <- function(data){
  expand.grid(mtry = seq(1, length(data), 1))
}
tuneGrid3 <- get_tuneGrid(data3)
tuneGrid7 <- get_tuneGrid(data7)
tuneGrid14 <- get_tuneGrid(data14)
tuneGrid28 <- get_tuneGrid(data28)
tuneGrid56 <- get_tuneGrid(data56)
tuneGrid100 <- get_tuneGrid(data100)
set.seed(1, sample.kind = "Rounding")
fit_3 <- train(y ~ .,
            data = data3,
            preProcess = c("center","scale"),
            method='parRF',
            tuneGrid=tuneGrid3,
            trControl=trControl3)
p_3 <- predict(fit_3, newdata = test3)
RMSE_test_3 <- RMSE(p_3, test3$y)
RMSE_test_3 # 3.31037
```

```
fit_3$bestTune # mtry = 6
set.seed(1, sample.kind = "Rounding")
fit_7 <- train(y ~ .,
            data = data7,
            preProcess = c("center","scale"),
            method='parRF',
            tuneGrid=tuneGrid7,
            trControl=trControl7)
p_7 <- predict(fit_7, newdata = test7)
RMSE_test_7 <- RMSE(p_7, test7$y)
RMSE_test_7 # 4.361987
fit_7$bestTune # mtry = 2
set.seed(1, sample.kind = "Rounding")
fit_14 <- train(y ~ .,
            data = data14,
            preProcess = c("center","scale"),
            method='parRF',
            tuneGrid=tuneGrid14,
            trControl=trControl14)
p_14 <- predict(fit_14, newdata = test14)
RMSE_test_14 <- RMSE(p_14, test14$y)
RMSE_test_14 # 4.620515
fit_14$bestTune # mtry = 13
set.seed(1, sample.kind = "Rounding")
fit_28 <- train(y ~ .,
            data = data28,
            preProcess = c("center","scale"),
            method='parRF',
            tuneGrid=tuneGrid28,
            trControl=trControl28)
p_28 <- predict(fit_28, newdata = test28)
RMSE_test_28 <- RMSE(p_28, test28$y)
RMSE_test_28 # 4.716698
fit_28$bestTune # mtry = 11
set.seed(1, sample.kind = "Rounding")
fit_56 <- train(y ~ .,
            data = data56,
            preProcess = c("center","scale"),
            method='parRF',
            tuneGrid=tuneGrid56,
            trControl=trControl56)
p_56 <- predict(fit_56, newdata = test56)
RMSE_test_56 <- RMSE(p_56, test56$y)
RMSE_test_56 # 5.939163
fit_56$bestTune # mtry = 8
set.seed(1, sample.kind = "Rounding")
fit_100 <- train(y ~ .,
            data = data100,
            preProcess = c("center","scale"),
            method='parRF',
            tuneGrid=tuneGrid100,
            trControl=trControl100)
p_100 <- predict(fit_100, newdata = test100)
```

```r
RMSE_test_100 <- RMSE(p_100, test100$y)
RMSE_test_100 # 5.851088
fit_100$bestTune # mtry = 8
```

8.5 Results

8.5.1 Table - Models details

```r
# Table - Models details
colNames <- c("Model", "mtry", "CV", "Repetitions",
              "RMSE (train)", "RMSE (test)")
caption <- "Regression models results"
day <- c("3 days", "7 days", "14 days", "28 days", "56 days", "100 days")
dat_reg_models <- data.frame(
  dia = day,
  mtry = c(fit_3$bestTune$mtry, fit_7$bestTune$mtry, fit_14$bestTune$mtry,
           fit_28$bestTune$mtry, fit_56$bestTune$mtry, fit_100$bestTune$mtry),
  number = c(trControl3$number, trControl7$number, trControl14$number,
             trControl28$number,trControl56$number,trControl100$number),
  repeats = c(trControl3$repeats, trControl7$repeats, trControl14$repeats,
              trControl28$repeats,trControl56$repeats,trControl100$repeats),
  RMSE_train = c(min(fit_3$results$RMSE), min(fit_7$results$RMSE),
                 min(fit_14$results$RMSE), min(fit_28$results$RMSE),
                 min(fit_56$results$RMSE), min(fit_100$results$RMSE)),
  RMSE_test = c(RMSE_test_3, RMSE_test_7, RMSE_test_14,
                RMSE_test_28, RMSE_test_56, RMSE_test_100)
)
kable(
    dat_reg_models,
    col.names = colNames,
    escape = F,
    booktabs = T,
    caption = caption,
    linesep = "\\addlinespace",
    align = "c"
    ) %>%
  kable_styling(latex_options = c("HOLD_position"))
```

8.5.2 Figure - Models comparison

```r
# Figure - Models comparison
cap <- "Actual vs Predicted values in each model"
models <- c("3 days", "7 days", "14 days", "28 days", "56 days", "100 days")
xlabel <- "Actual (MPa)"
ylabel <- "Predicted (MPa)"
preds <- list(p_3, p_7, p_14, p_28, p_56, p_100)
actuals <- list(test3$y, test7$y, test14$y, test28$y, test56$y, test100$y)
gen_res_df <- function(ind){
  data.frame(actual = actuals[[ind]], pred = preds[[ind]], model = models[[ind]])
}
res_df <- lapply(1:6, gen_res_df)
res_df <- bind_rows(res_df)
```

```
res_df$model <- factor(res_df$model,levels=models)
res_df %>%
  ggplot(aes(actual, pred)) +
  facet_wrap(~ model, ncol=3) +
  geom_point(alpha=0.5) +
  theme_bw() +
  geom_abline(slope=1, intercept=0) +
  xlab(xlabel) +
  ylab(ylabel)
```

8.5.3 Results tables of 10 best and worst results

```
# Results tables of 10 best and worst results
colNames1 = c("Actual", "Predicted", "Error", "","Actual", "Predicted", "Error")
colNames2 = c("Best 10"=3,"", "Worst 10"=3)
caption_0 <- "Model of "
get_X <- function(pred, actual, X=10){
  diff <- abs(pred - actual)
  diff_2 <- pred - actual
  ind_min <- which(diff <= max(sort(diff, decreasing = F)[1:X]), arr.ind = T)
  ind_max <- which(diff >= min(sort(diff, decreasing = T)[1:X]), arr.ind = T)
  df_min <- data.frame(
    actual_min=actual[ind_min],
    pred_min=pred[ind_min],
    diff_min=diff[ind_min],
    diff_min_2=diff_2[ind_min]
  )
  df_max <- data.frame(
    actual_max=actual[ind_max],
    pred_max=pred[ind_max],
    diff_max=diff[ind_max],
    diff_max_2=diff_2[ind_max]
  )
  df_max <- df_max[order(-df_max$diff_max),]
  df_min <- df_min[order(df_min$diff_min),]
  df_null <- data.frame(null_col = rep(c(""), X))
  res <- cbind(df_min,df_null, df_max)
  res <- res %>% select(-c(diff_max, diff_min))
  rownames(res) <- c()
  res
}
gen_kable <- function(ind){
  df <- get_X(preds[[ind]], actuals[[ind]])
  caption <- paste0(caption_0,models[ind])
  kable(
    df,
    col.names = colNames1,
    escape = F,
    booktabs = T,
    caption = caption,
    linesep = "\\addlinespace",
    align = "c"
    )  %>%
```

```
  column_spec(4, width = "1cm",) %>%
  add_header_above(header = colNames2, line = T, align = "c") %>%
  kable_styling(latex_options = c("HOLD_position"))
}
gen_kable(1)
gen_kable(2)
gen_kable(3)
gen_kable(4)
gen_kable(5)
gen_kable(6)
```

8.6 Discussion

8.6.1 Table - Comparison of other works

```
# Table - Comparison of other works
colNames = c("Author","Year" ,"Algorithm", "RMSE")
caption <- "Comparison of other works"
works <- data.frame(
  autor = c("Pierobon", "Hameed", "Raj","Modukuru" ,"Alshamiri", "Abban"),
  ano = c("2018", "2020", "2018","2020" ,"2020", "2016"),
  modelo = c("Ensemble com 5 algorítimos", "Artificial Neural Networks",
          "Gradient Boosting Regressor","Random Forest Regressor" ,
          "Regularized Extreme Learning Machine",
          "Support Vector Machines with Radial Basis Function Kernel"),
  RMSE = c("4.150", "4.736", "4.957","5.080","5.508", "6.105")
)
kable(
  works,
  col.names = colNames,
  escape = F,
  booktabs = T,
  caption = caption,
  linesep = "\\addlinespace",
  align = "l"
  )  %>%
    kable_styling(latex_options = c("HOLD_position"))
```

8.6.2 Table - Final results

```
# Table - Final results
colNames <- c("Model", "RMSE")
caption <- "Final result"
day <- c("3 days", "7 days", "14 days", "28 days", "56 days", "100 days")
dat_reg_models <- data.frame(
  dia = day,
  RMSE_test = c(RMSE_test_3, RMSE_test_7, RMSE_test_14,
                RMSE_test_28, RMSE_test_56, RMSE_test_100)
)
kable(
    dat_reg_models,
    col.names = colNames,
```

```
    escape = F,
    booktabs = T,
    caption = caption,
    linesep = "\\addlinespace",
    align = "c"
    )  %>%
  kable_styling(latex_options = c("HOLD_position"))
```