Interpretability and Explainability in Machine Learning

Biel Caballero Vergés, Svenja Menzenbach and Kleber Enrique Reyes Illescas

2023-12-29

Data preparation

```
concrete <- as.data.frame(read_excel("Concrete_Data.xls"))
DescVars <- names(concrete)
names(concrete) <- c("Cement","Slag","FlyAsh","Water","Superplast",
                     "CoarseAggr","FineAggr","Age","Strength")
```

1. Fit a Random Forest

a. Compute the Variable Importance by the reduction of the impurity at the splits defined by each variable.

```
model_rf_imp <- ranger(
  Strength ~ .,
  data = train_set,
  importance='impurity'
)
print(model_rf_imp)
```

```
## Ranger result
##
## Call:
##  ranger(Strength ~ ., data = train_set, importance = "impurity")
##
## Type:                             Regression
## Number of trees:                  500
## Sample size:                      700
## Number of independent variables:  8
## Mtry:                             2
## Target node size:                 5
## Variable importance mode:         impurity
## Splitrule:                        variance
## OOB prediction error (MSE):       34.53954
## R squared (OOB):                  0.877664
```

b. Compute the Variable Importance by out-of-bag random permutations.

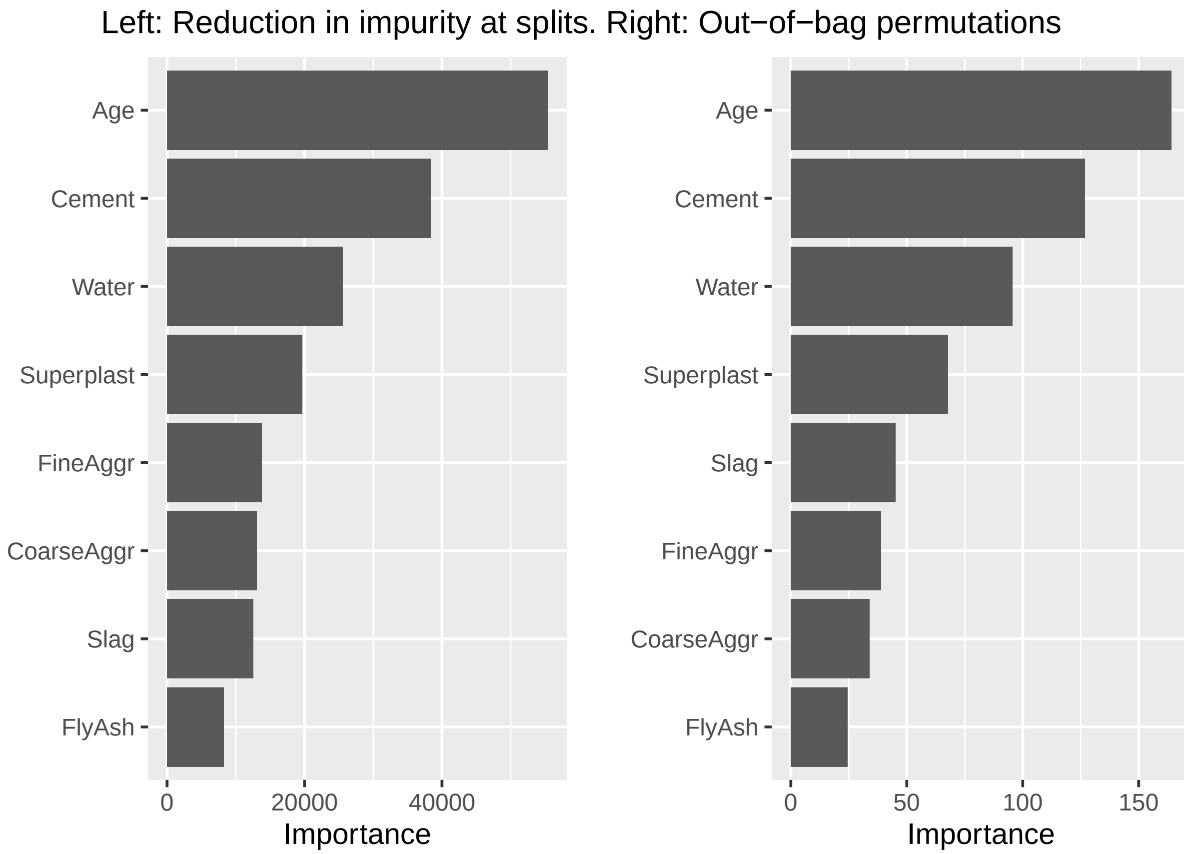
```r
model_rf_perm <- ranger(
  Strength ~ .,
  data = train_set,
  importance='permutation'
)
print(model_rf_perm)
```

```
## Ranger result
##
## Call:
##  ranger(Strength ~ ., data = train_set, importance = "permutation")
##
## Type:                             Regression
## Number of trees:                  500
## Sample size:                      700
## Number of independent variables:  8
## Mtry:                             2
## Target node size:                 5
## Variable importance mode:         permutation
## Splitrule:                        variance
## OOB prediction error (MSE):       35.68536
## R squared (OOB):                  0.8736056
```

Both methods have similar performance (impurity being slightly better)

c. Do a graphical representation of both Variable Importance measures.

```r
rf_imp_vip <- vip(model_rf_imp)
rf_perm_vip <- vip(model_rf_perm)
grid.arrange(rf_imp_vip, rf_perm_vip, ncol=2,
             top="Left: Reduction in impurity at splits. Right: Out-of-bag permutations")
```



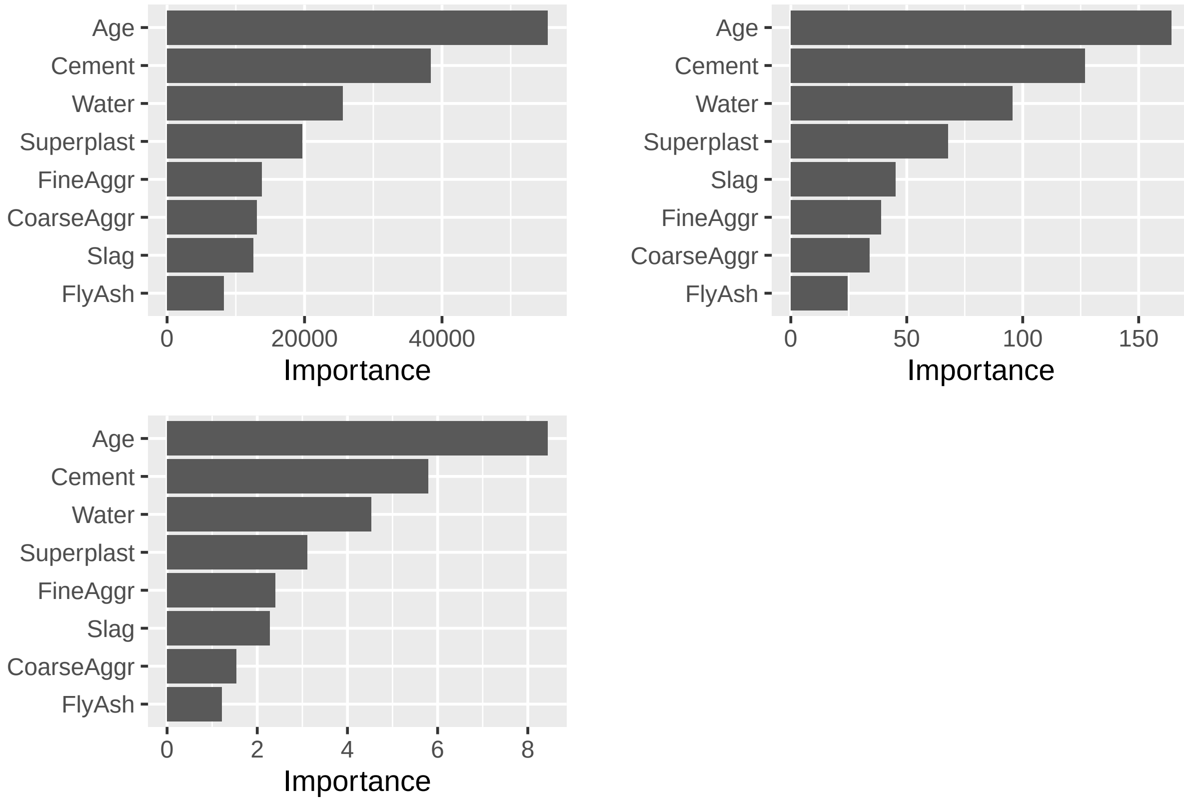
Both methods agree on nearly every parameter. Age, cement, water and superplast are unquestionably the four most significant variables.

d. Compute the Variable Importance of each variable by Shapley Values.

```
rf_shapley <- vip(model_rf_imp, method = "shap",
                  pred_wrapper = yhat, num_features = 8,
                  train = train_set,
                  newdata = test_set[,-9])

grid.arrange(rf_imp_vip, rf_perm_vip, rf_shapley,
             ncol=2, nrow=2,
             top="Top left: Impurity. Top right: oob permutations. Bottom left: Shapley values"
            )
```

Top left: Impurity. Top right: oob permutations. Bottom left: Shapley values



Shapley's results align with the same trend, reaffirming the most important variables.

2. Fit a linear model and a gam model.

a. Summarize, numerically and graphically, the fitted models.

```
lm_strength <- lm(Strength ~ ., data = train_set)
(summ_lm_strength <- summary(lm_strength))
```

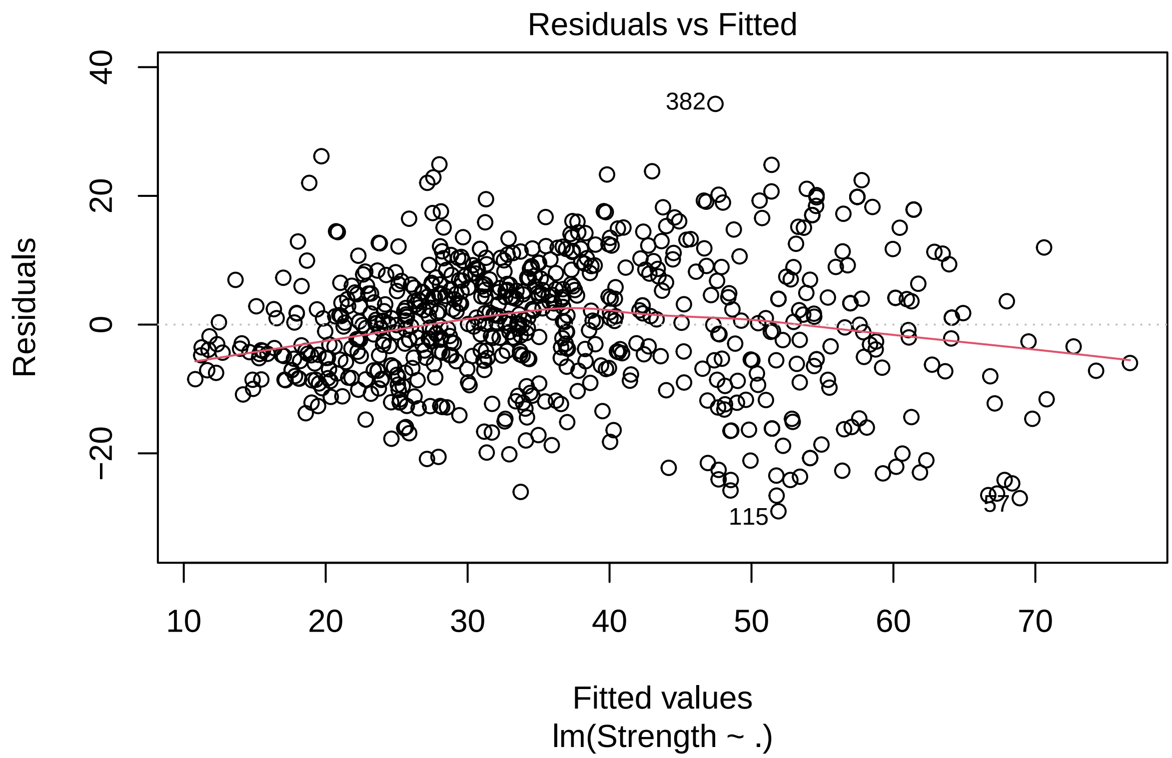
```
##
## Call:
## lm(formula = Strength ~ ., data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.003  -6.253   0.355   6.380  34.288
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -37.444031  32.441685  -1.154  0.24882
## Cement        0.122253   0.010483  11.661  < 2e-16 ***
## Slag          0.111016   0.012583   8.823  < 2e-16 ***
## FlyAsh        0.094141   0.015581   6.042 2.49e-09 ***
## Water        -0.130398   0.048175  -2.707  0.00696 **
```

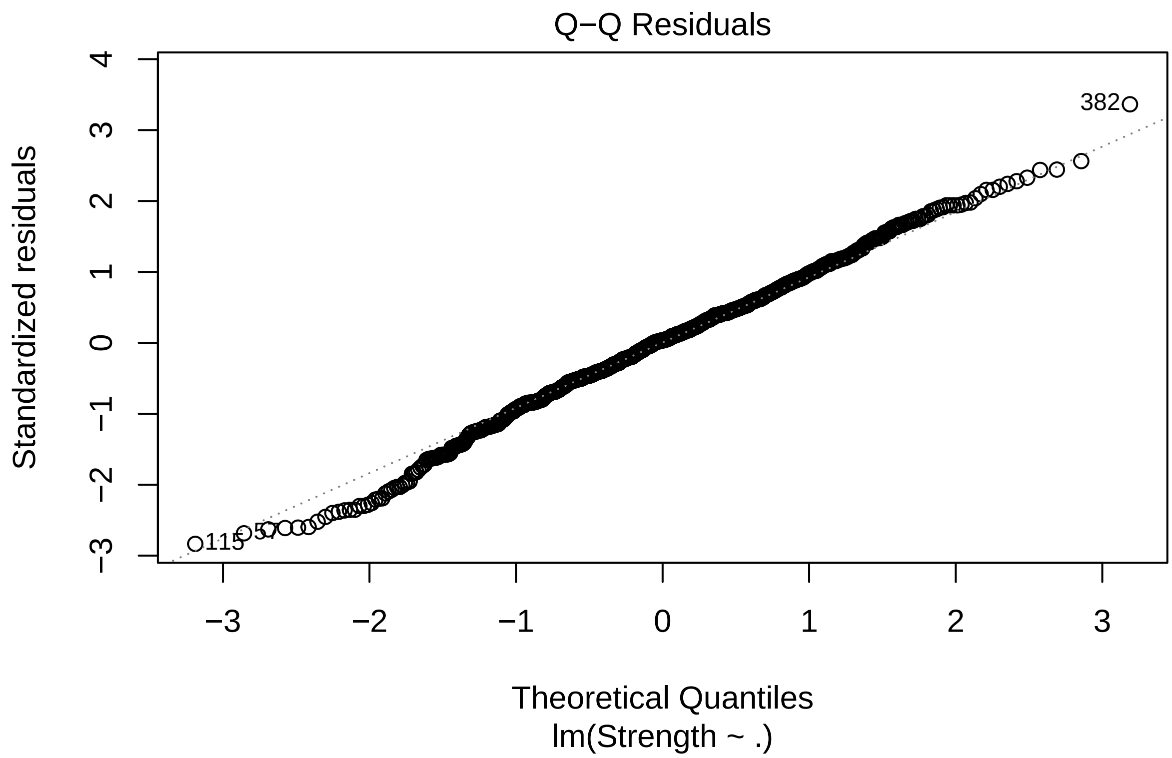
```
## Superplast    0.324301   0.110096   2.946  0.00333 **
## CoarseAggr    0.023198   0.011473   2.022  0.04356 *
## FineAggr      0.025225   0.013078   1.929  0.05418 .
## Age           0.113435   0.006538  17.349  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.27 on 691 degrees of freedom
## Multiple R-squared:  0.6308, Adjusted R-squared:  0.6265
## F-statistic: 147.6 on 8 and 691 DF,  p-value: < 2.2e-16
```

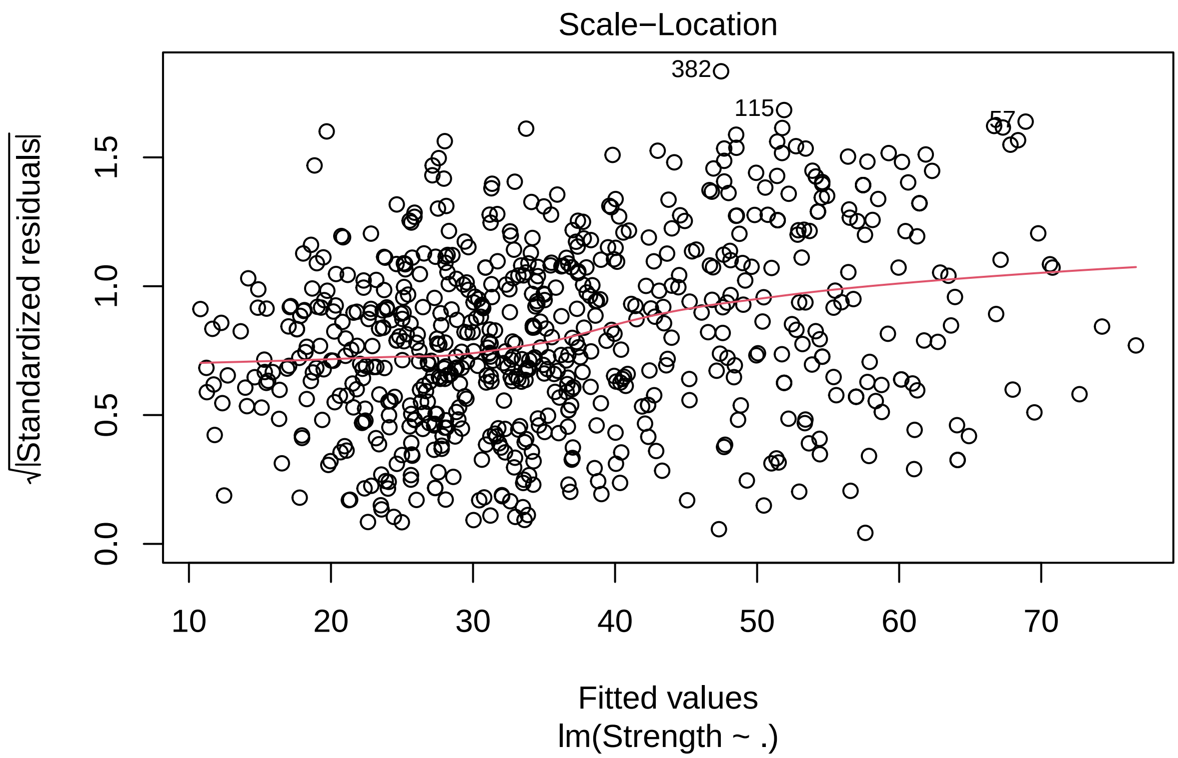
```
gam_strength <- gam(Strength ~ s(Cement, k=30) + s(Slag, k=45) + s(FlyAsh, k=30) +
                        s(Water, k=30) + s(Superplast, k=30) + s(CoarseAggr, k=30) +
                        s(FineAggr, k=30) + s(Age, k=10),
                  data = train_set)
(summ_gam_strength <- summary(gam_strength))
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Strength ~ s(Cement, k = 30) + s(Slag, k = 45) + s(FlyAsh, k = 30) +
##     s(Water, k = 30) + s(Superplast, k = 30) + s(CoarseAggr,
##     k = 30) + s(FineAggr, k = 30) + s(Age, k = 10)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.0285     0.1809   199.2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                  edf Ref.df       F p-value
## s(Cement)      5.301  6.557  22.917  <2e-16 ***
## s(Slag)       37.691 41.115   3.968  <2e-16 ***
## s(FlyAsh)      8.400  9.864   2.500  0.0063 **
## s(Water)      26.409 27.995   6.326  <2e-16 ***
## s(Superplast) 16.899 19.552   4.212  <2e-16 ***
## s(CoarseAggr) 21.732 24.862   1.811  0.0103 *
## s(FineAggr)   14.561 17.354   5.798  <2e-16 ***
## s(Age)         8.311  8.746 278.119  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.919   Deviance explained = 93.5%
## GCV = 28.647  Scale est. = 22.905    n = 700
```

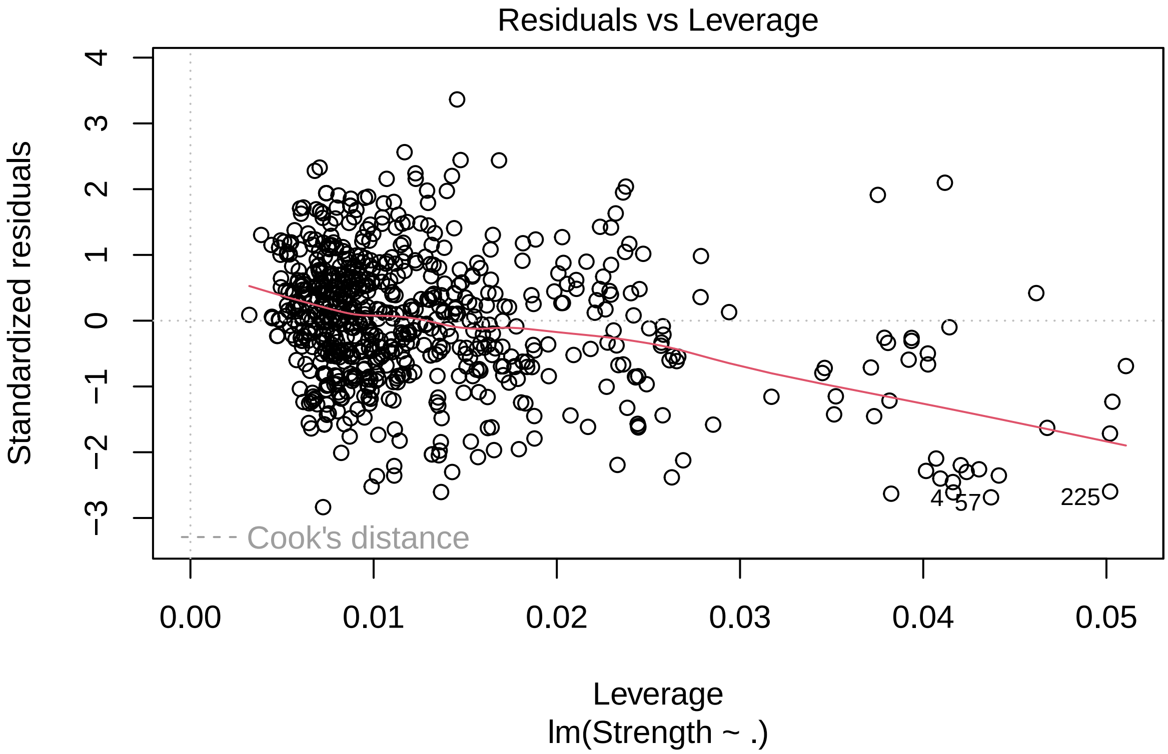
```
plot(lm_strength)
```


Residuals vs Fitted
Residuals
40
20
0
−20
382
115
57
10
20
30
40
50
60
70
Fitted values
lm(Strength ~ .)

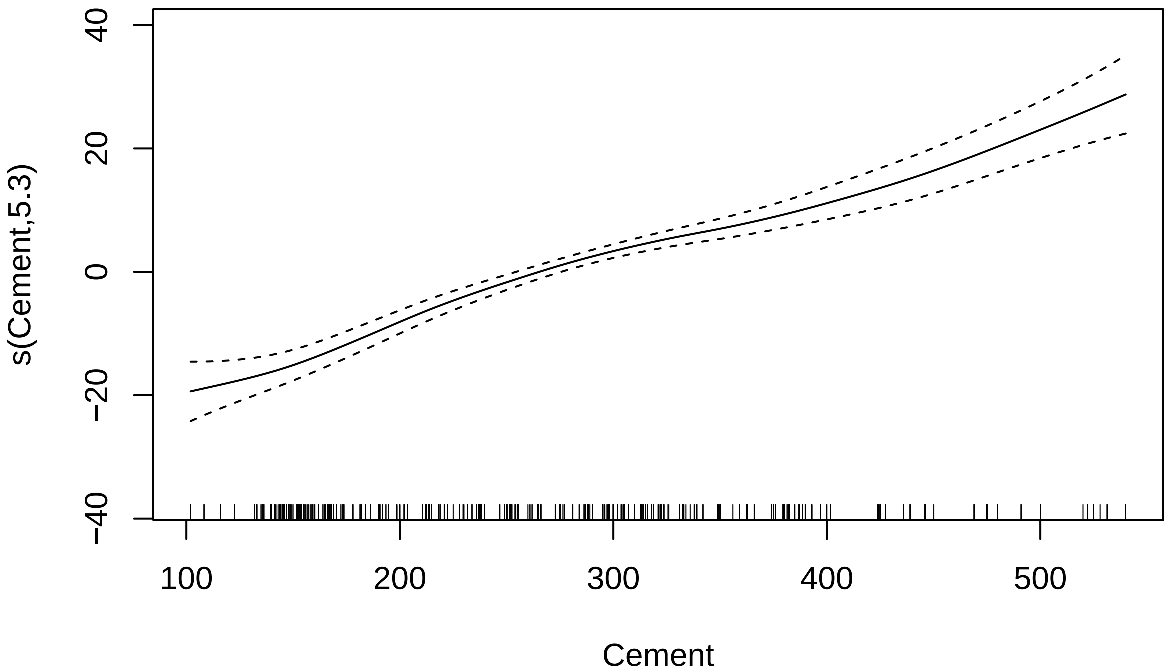

Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(Strength ~ .)
382
115
57
−3
−2
−1
0
1
2
3
4

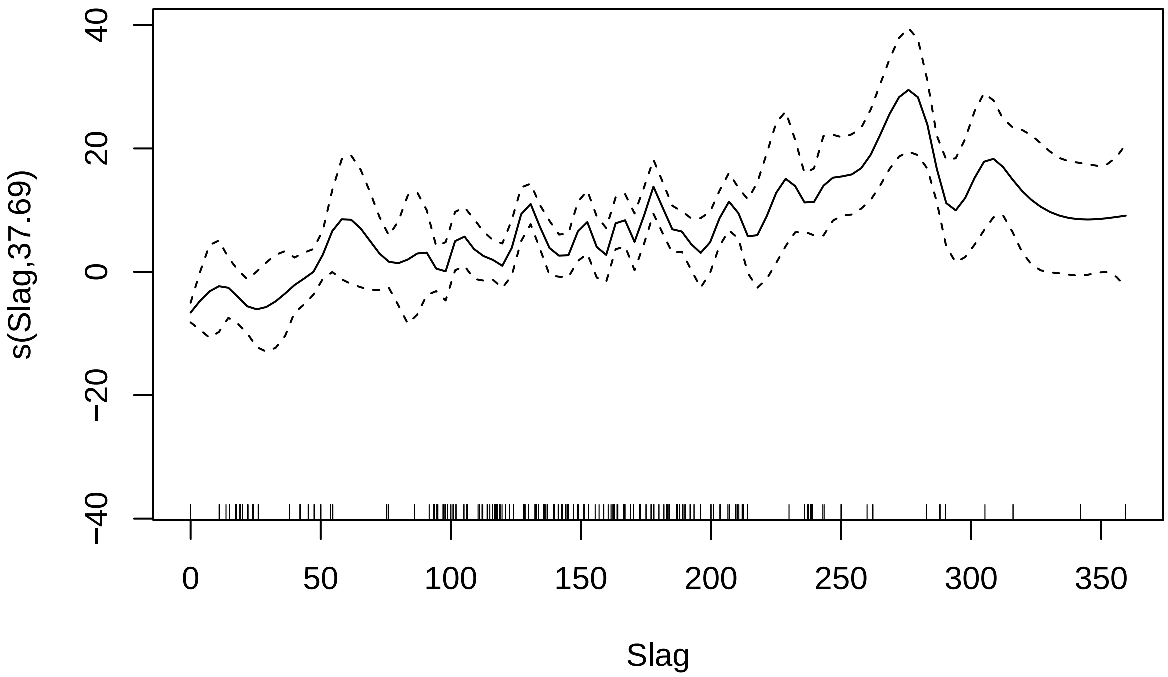


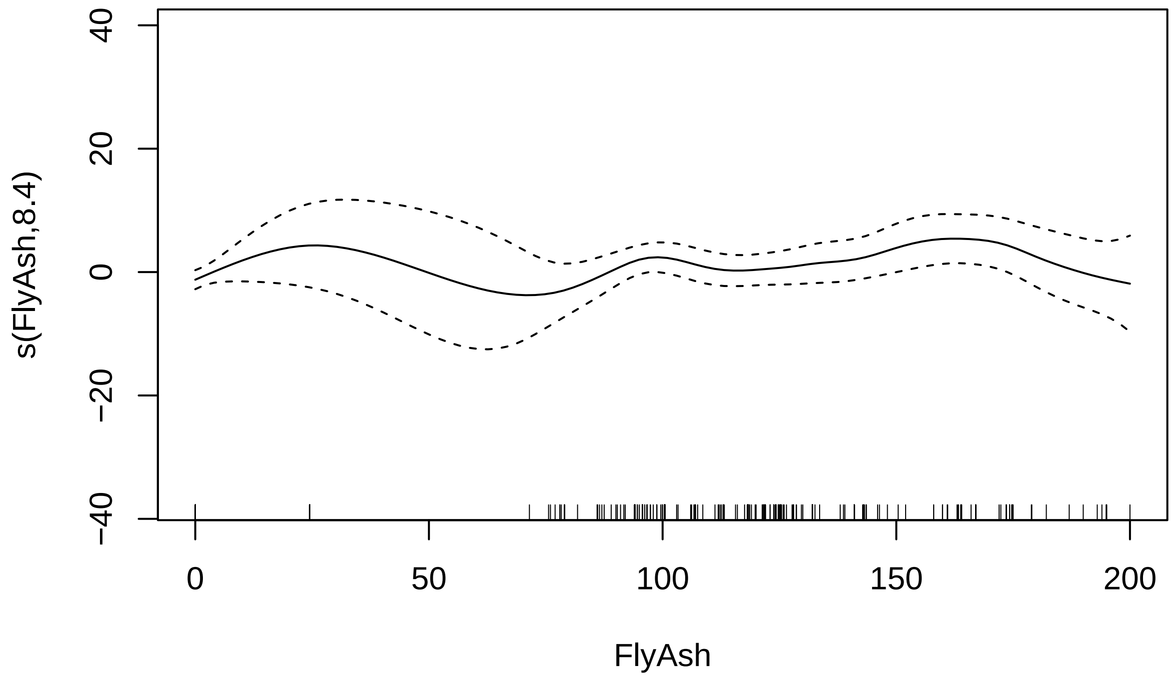
Scale−Location
382
115
57
1.5
1.0
0.5
0.0
√|Standardized residuals|
10
20
30
40
50
60
70
Fitted values
lm(Strength ~ .)



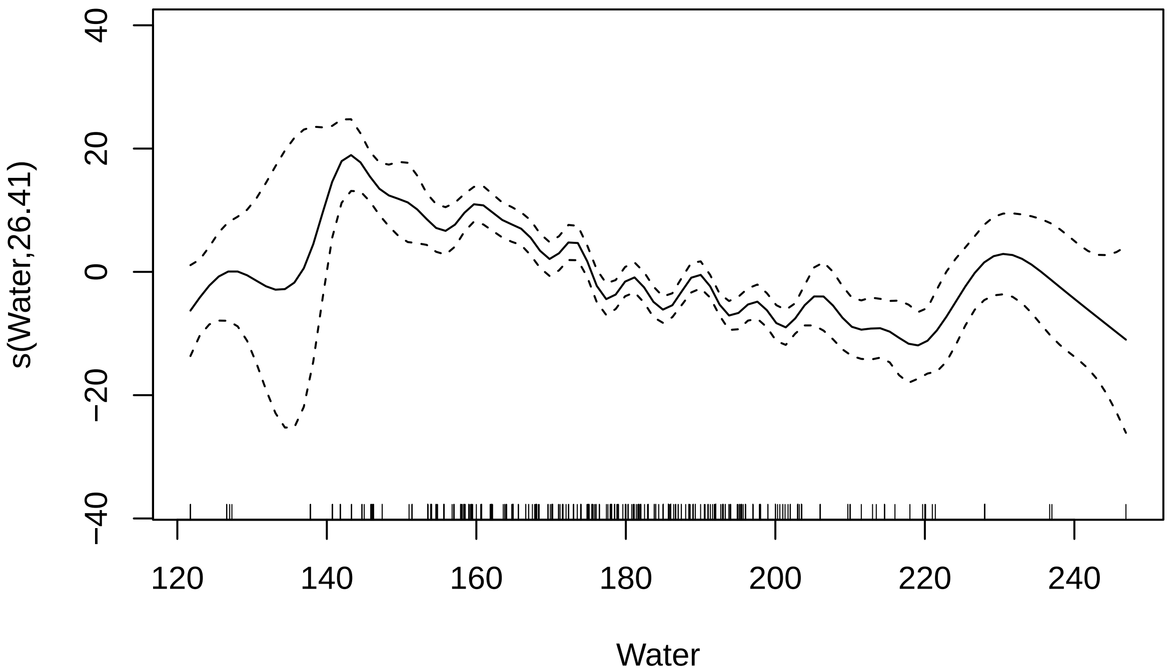

```
plot(gam_strength)
```


s(Cement,5.3)
40
20
0
−20
−40
100
200
300
400
500
Cement

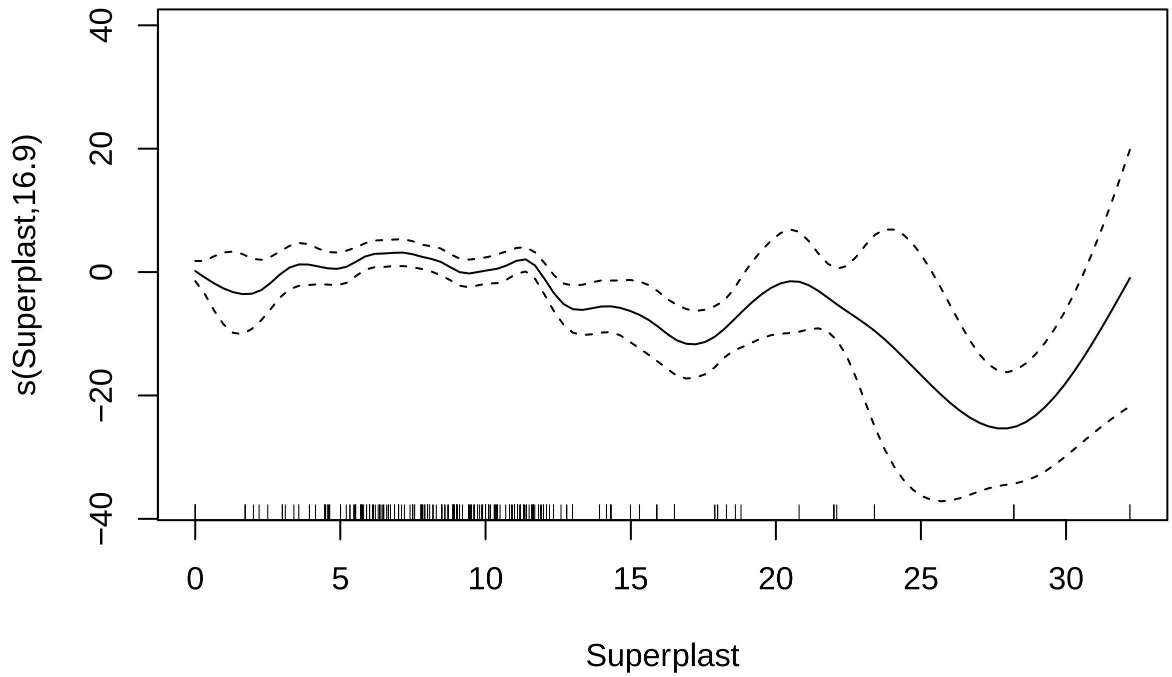

s(Slag,37.69)
40
20
0
−20
−40
0
50
100
150
200
250
300
350
Slag

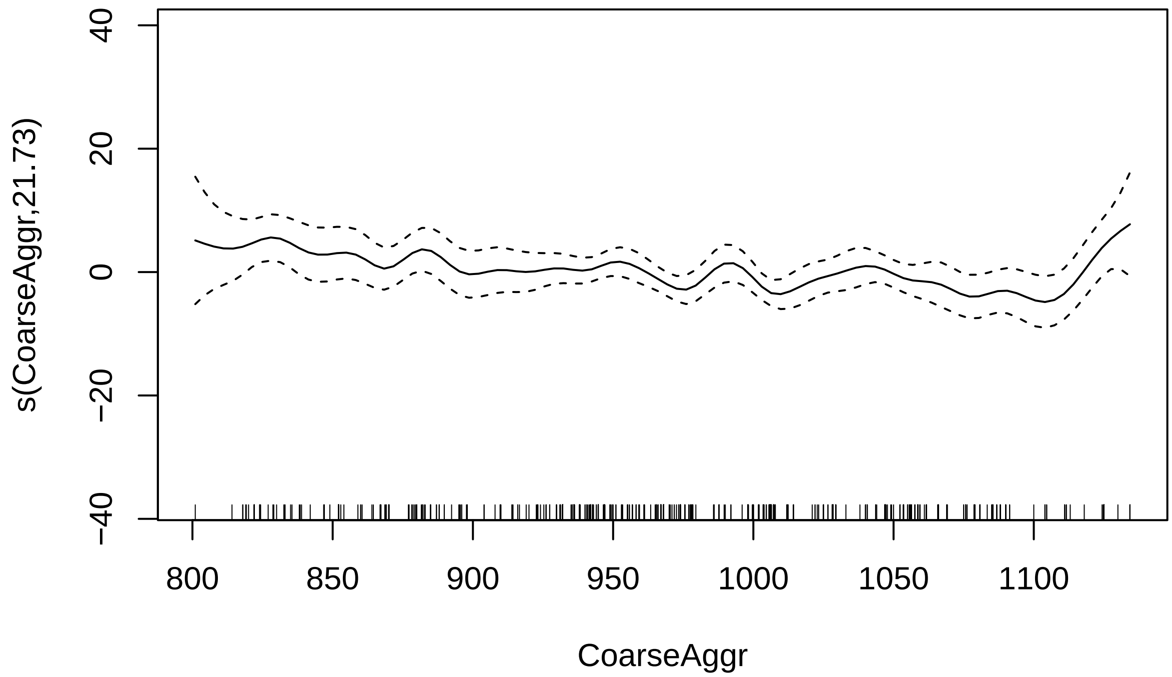


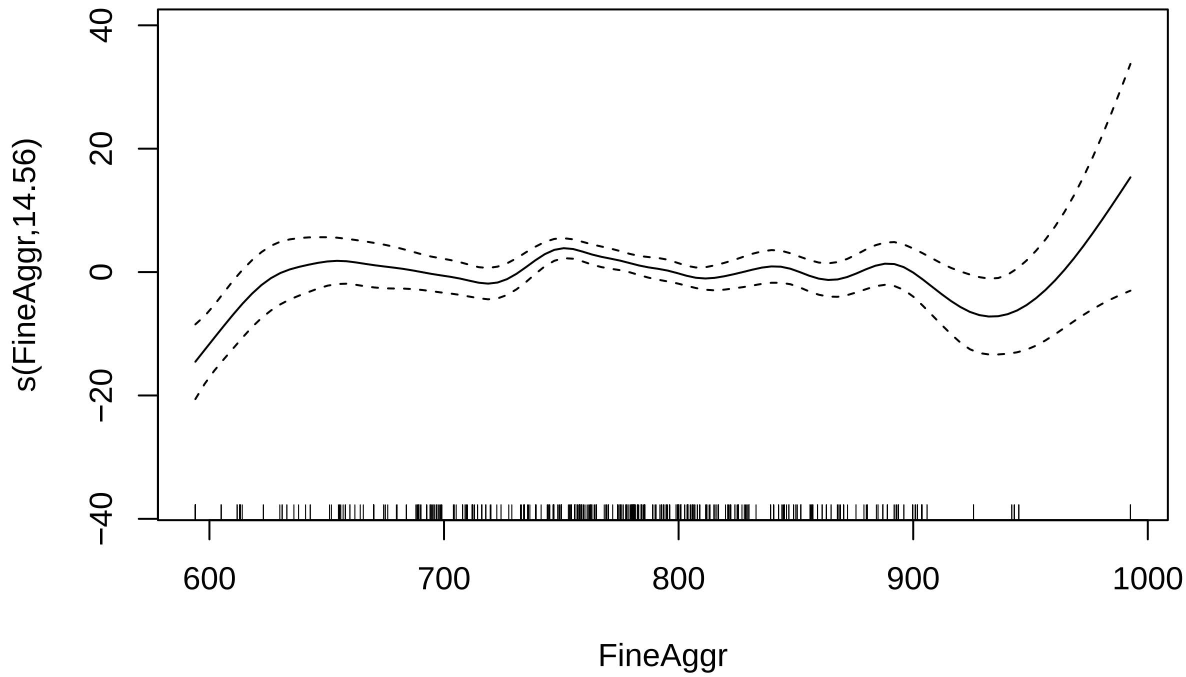
40
20
0
−20
−40
s(FlyAsh,8.4)
0
50
100
150
200
FlyAsh

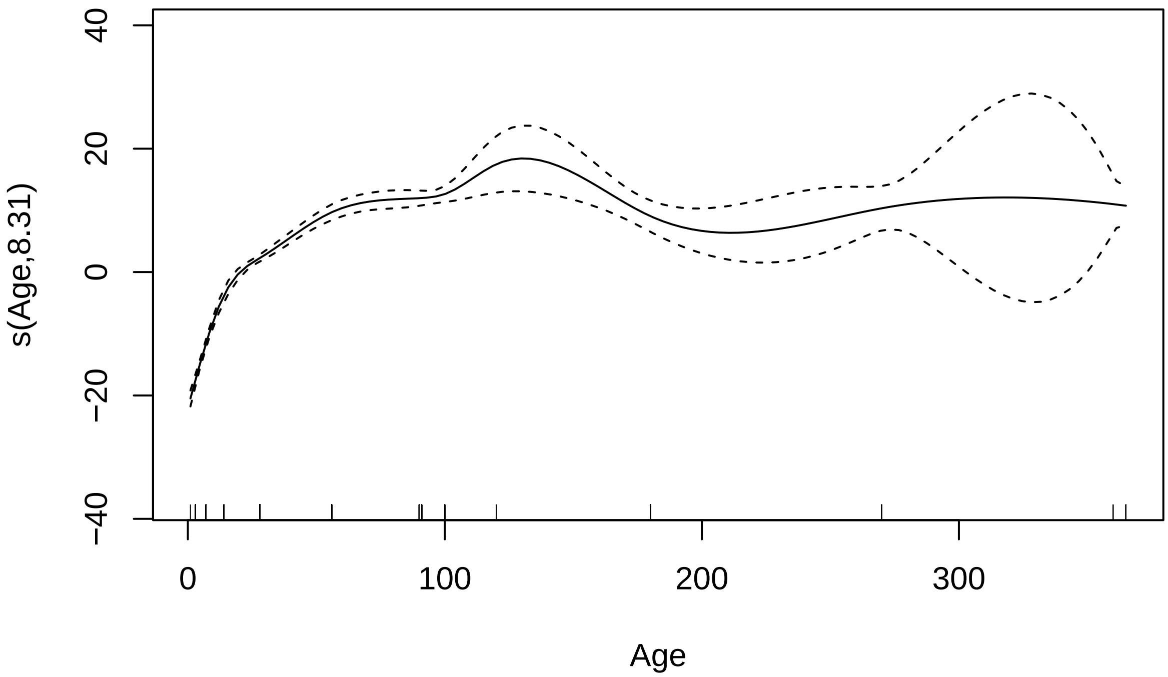


s(Water,26.41)
40
20
0
−20
−40
120
140
160
180
200
220
240
Water

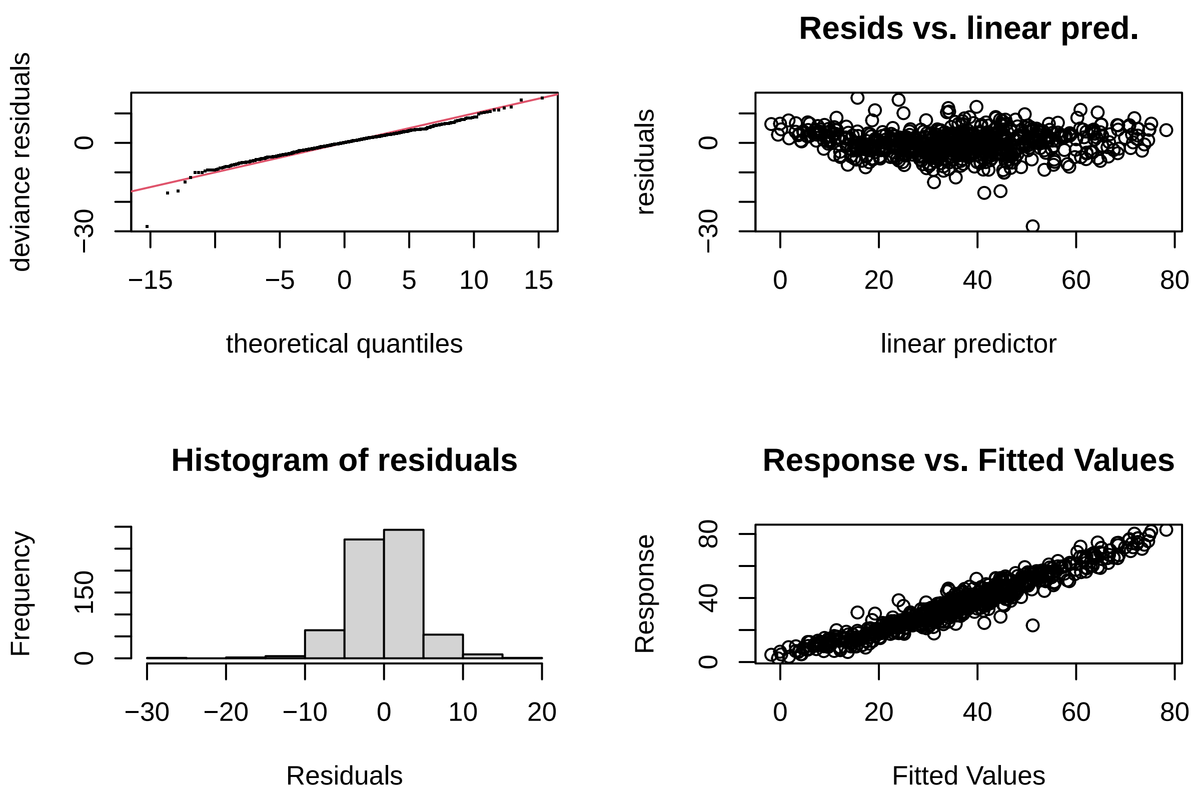

s(Superplast,16.9)
40
20
0
−20
−40
0
5
10
15
20
25
30
Superplast


40
20
0
−20
−40
s(CoarseAggr,21.73)
800
850
900
950
1000
1050
1100
CoarseAggr


s(FineAggr,14.56)
40
20
0
−20
−40
600
700
800
900
1000
FineAggr




```r
gam.check(gam_strength)
```




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 17 iterations.
## The RMS GCV score gradient at convergence was 4.505733e-06 .
## The Hessian was positive definite.
## Model rank =  228 / 228
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                   k'   edf k-index p-value
## s(Cement)      29.00  5.30    1.01   0.605
## s(Slag)        44.00 37.69    1.02   0.625
## s(FlyAsh)      29.00  8.40    0.97   0.240
## s(Water)       29.00 26.41    0.95   0.080 .
## s(Superplast) 29.00 16.90    1.01   0.595
## s(CoarseAggr) 29.00 21.73    0.95   0.075 .
## s(FineAggr)   29.00 14.56    1.00   0.510
## s(Age)          9.00  8.31    1.02   0.705
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

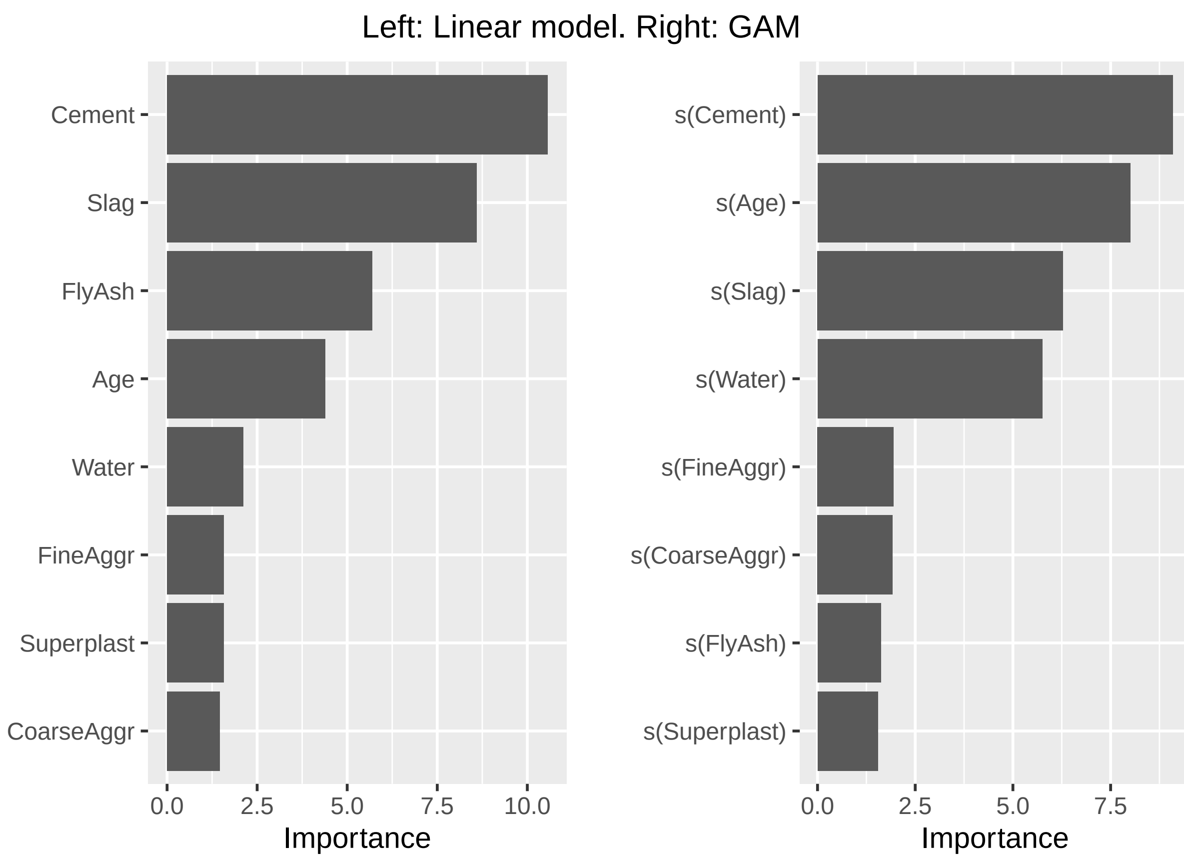
Without hesitation, the gam model looks better than the linear one. The error is minor, and the gam.check plots look very nice.

b. Compute the Variable Importance by Shappley values in the linear and gam fitted models. Compare your results with what you have learned before.

```
lm_strength_shapley <- vip(lm_strength, method="shap",
                  pred_wrapper=predict.lm,
                  train=train_set,
                  newdata=test_set[,-9],
                  num_features = 8,
                  exact=TRUE)

gam_strength_shapley <- vip(gam_strength, method="shap",
                  pred_wrapper=predict.gam,
                  train=train_set,
                  newdata=test_set[,-9],
                  num_features = 8,
                  exact=TRUE)

grid.arrange(lm_strength_shapley, gam_strength_shapley, ncol=2,
             top="Left: Linear model. Right: GAM")
```



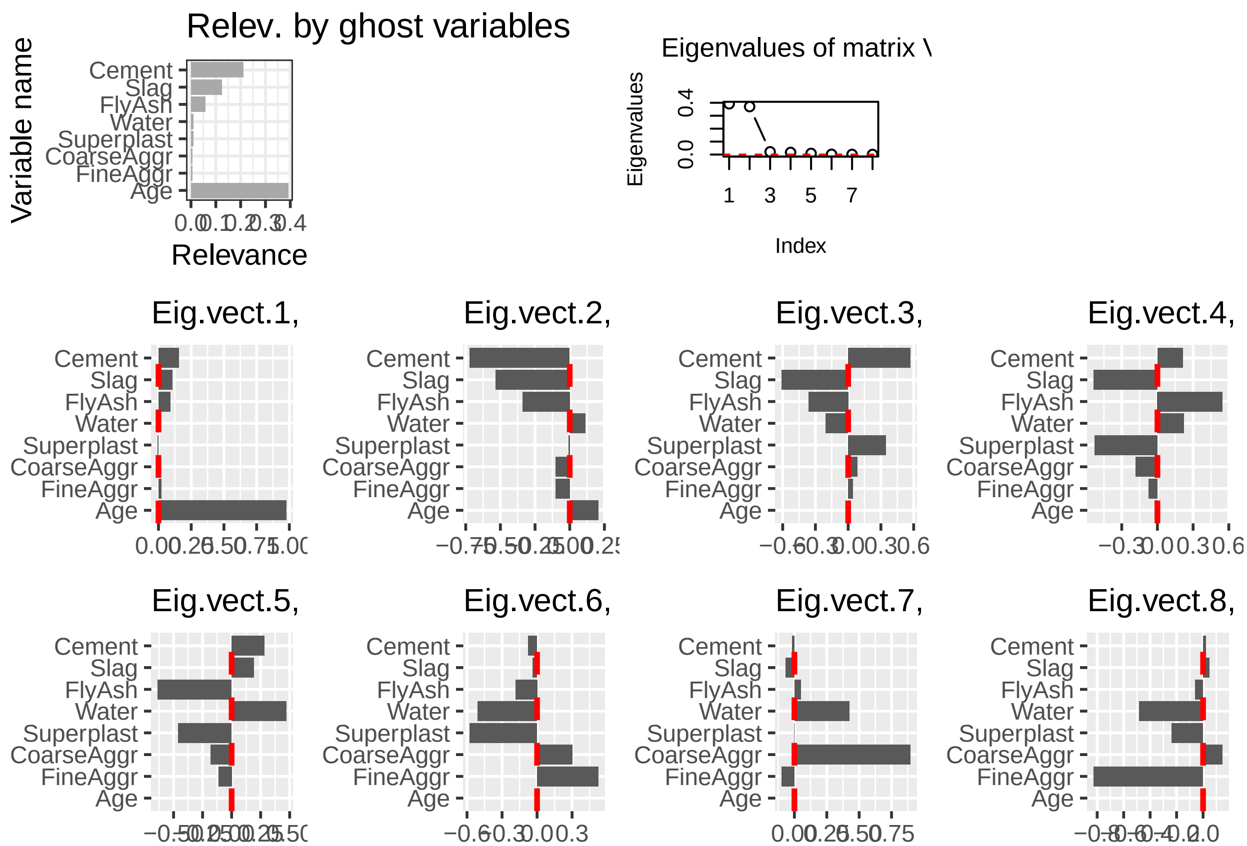
In this case Cement is the most important variable, but Age gains more relevance is the gam model.

3. Relevance by Ghost Variables

Compute the relevance by ghots variables in the three fitted models.

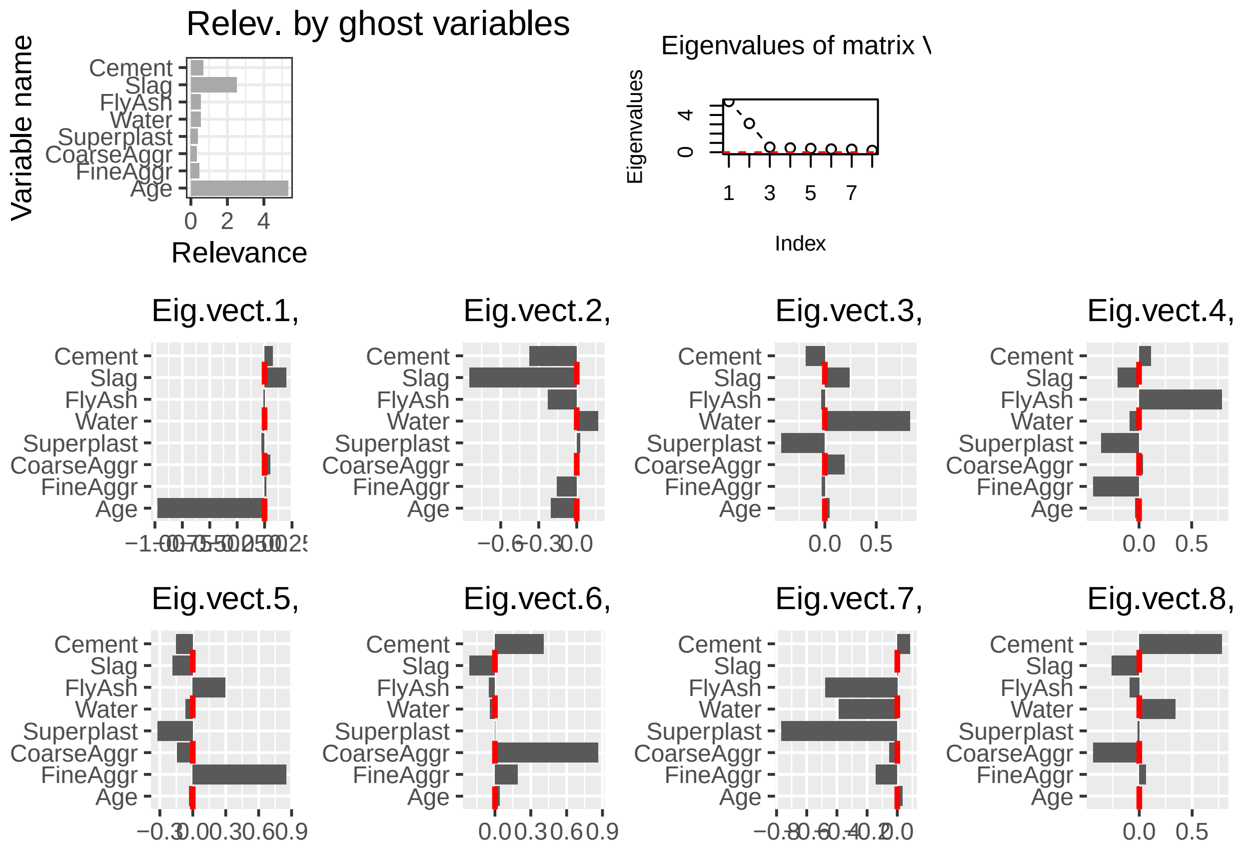
```r
source("relev.ghost.var.R")
Rel_Gh_Var <- relev.ghost.var(model=lm_strength,
                              newdata = test_set[, -9],
                              y.ts = test_set[, 9],
                              func.model.ghost.var = lm
)

plot.relev.ghost.var(Rel_Gh_Var,n1=500,ncols.plot = 4)
```




```r
Rel_Gh_Var <- relev.ghost.var(model=gam_strength,
                              newdata = test_set[, -9],
                              y.ts = test_set[, 9],
                              func.model.ghost.var = lm
)

plot.relev.ghost.var(Rel_Gh_Var,n1=500,ncols.plot = 4)
```



4. Global Importance Measures and Plots using the library DALEX

a. Compute Variable Importance by Random Permutations

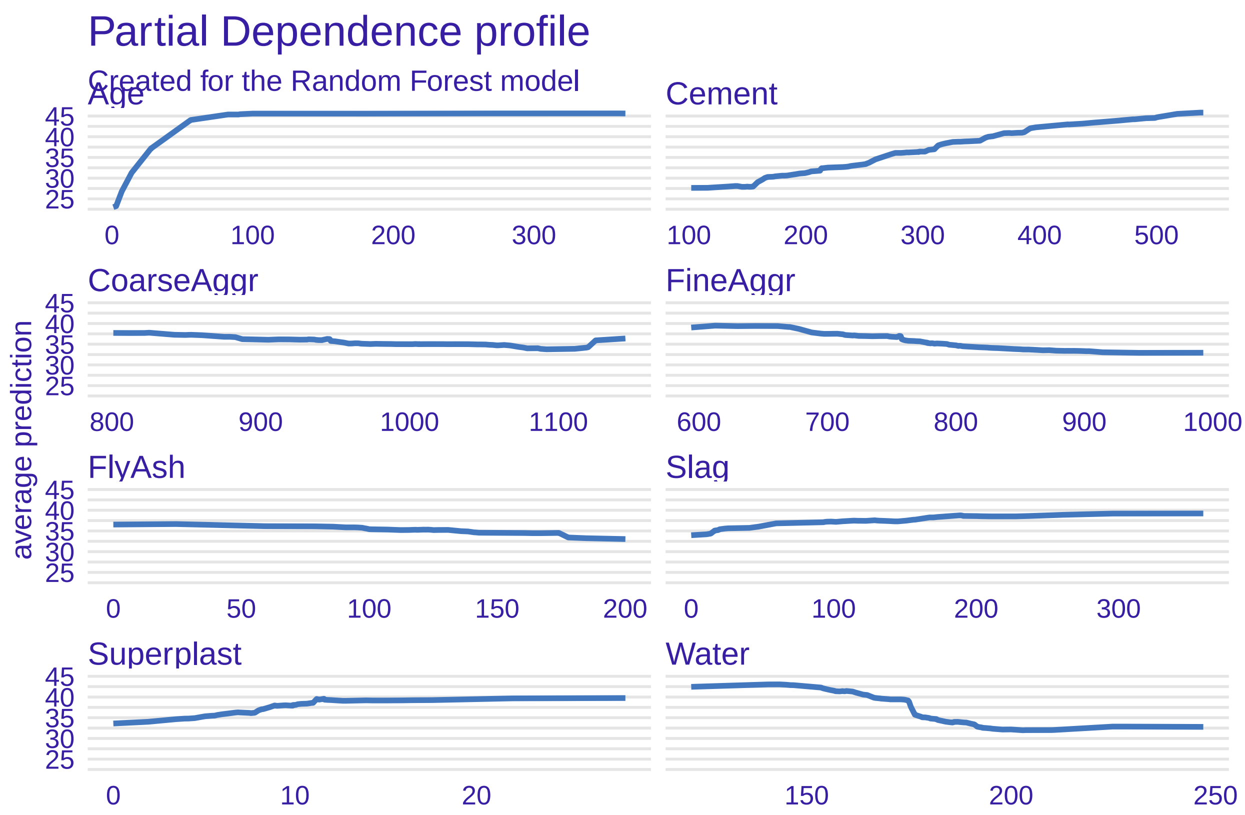
```
explainer_rf <- explain.default(model = model_rf_imp,
                               data = test_set[, -9],
                               y = test_set[, 9],
                               label = "Random Forest")
```

```
## Preparation of a new explainer is initiated
##   -> model label       :  Random Forest
##   -> data              :  330  rows  8  cols
##   -> target variable   :  330  values
##   -> predict function  :  yhat.ranger  will be used (  default  )
##   -> predicted values  :  No value for predict function target column. (  default  )
##   -> model_info        :  package ranger , ver. 0.16.0 , task regression (  default  )
##   -> predicted values  :  numerical, min =  8.99662 , mean =  35.46248 , max =  76.30324
##   -> residual function :  difference between y and yhat (  default  )
##   -> residuals         :  numerical, min =  -19.89475 , mean =  -0.09154271 , max =  24.07009
##   A new explainer has been created!
```

b. Do the Partial Dependence Plot for each explanatory variable.

```r
PDP_rf <- model_profile(
  explainer=explainer_rf,
  variables = NULL,  # All variables are used
  N = NULL, # All available data are used
  groups = NULL,
  k = NULL,
  center = TRUE,
  type = "partial" #  partial, conditional or accumulated
)

plot(PDP_rf, facet_ncol=2)
```



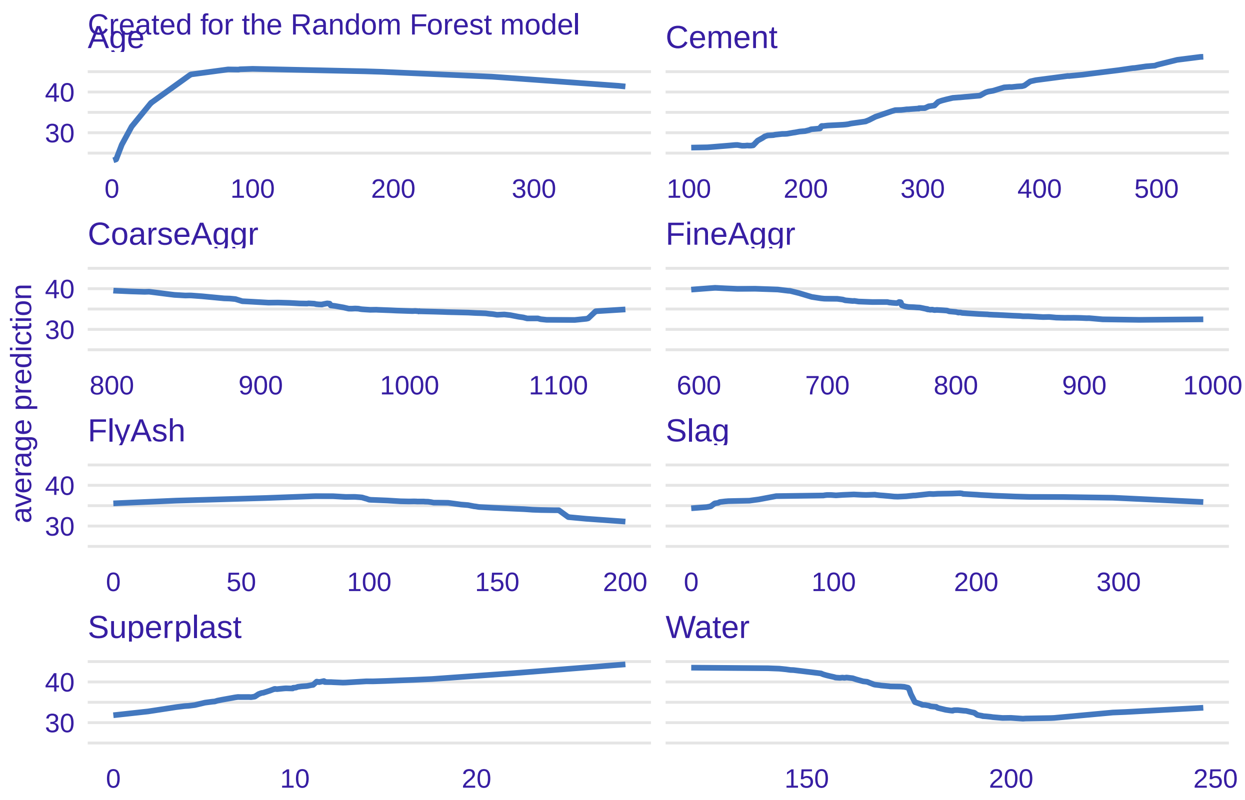
Cement and Slag show a consistent increase in predicted Strength with higher quantities, implying their positive impact on concrete Strength. FineAggr and FlyAsh on the other hand show a constant decrease. CoarseAggr has a little increase at 1150 so it has a minimum. Age increases strongly at the beginning and converges fast to a certain value. Superplast also increases at the beginning and converges to a certain value but the increase is lower. Water shows a decreasing S-curve with a higher slope at around 175. Age, Cement and Water seem to have the highest impact.

c. Do the Local (or Conditional) Dependence Plot for each explanatory variable.

```r
CDP_rf <- model_profile(
  explainer=explainer_rf,
  variables = NULL,  # All variables are used
  N = NULL, # All available data are used
  groups = NULL,
```

```
  k = NULL,
  center = TRUE,
  type = "conditional" #  partial, conditional or accumulated
)

plot(CDP_rf, facet_ncol=2)
```



In comparison to the previous plot age decreases a bit after reaching a maximum. Also Superplast continous increasing after 12. But in general the plots look very similar.

5. Local explainers with library DALEX

Choose two instances in the the test set, the prediction for which we want to explain:

- The data with the lowest value in Strength.
- The data with the largest value in Strength.

For these two instances, do the following tasks for the fitted random forest.

```
lowestStrength = concrete[which.min(concrete$Strength), ]
highestStrength = concrete[which.max(concrete$Strength), ]
```

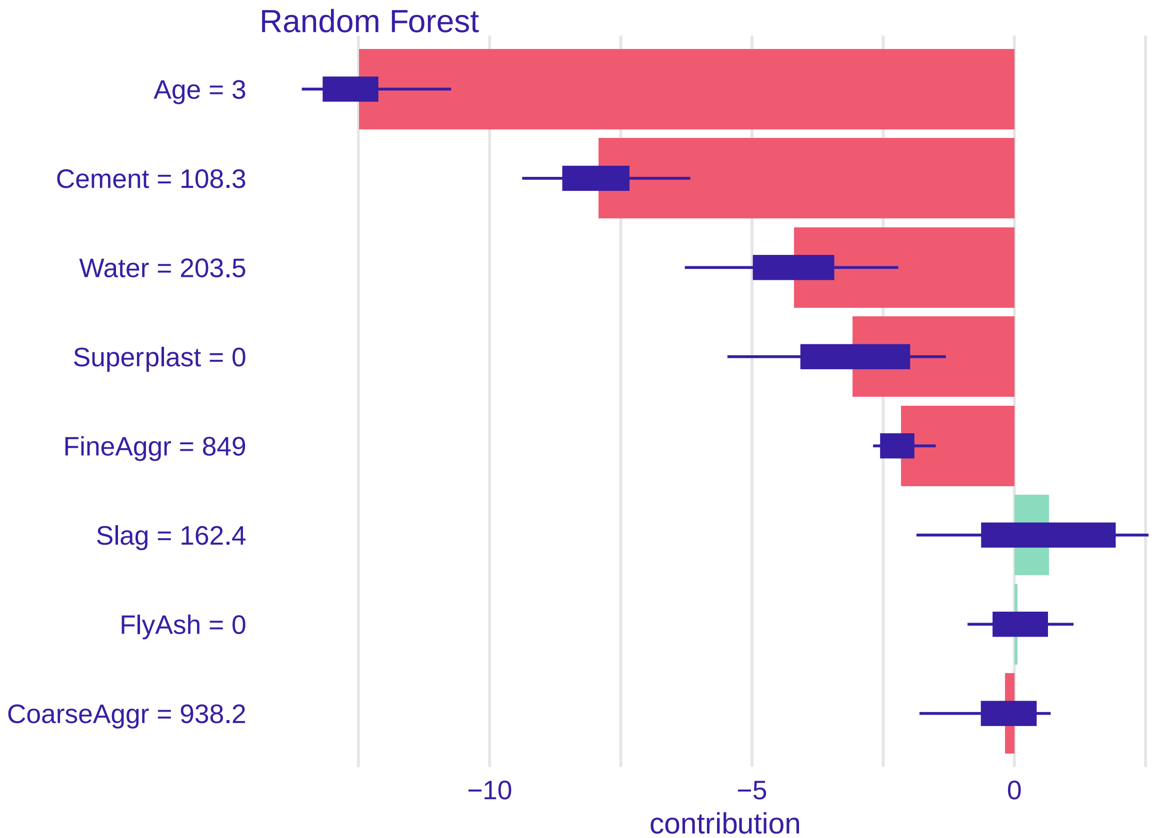
a. Explain the predictions using SHAP.

```r
bd_rf <- predict_parts(explainer = explainer_rf,
                new_observation = lowestStrength,
                           type = "shap")

bd_rf
```

```
##                                           min         q1       median
## Random Forest: Age = 3             -13.5790340 -13.1547108 -12.22659592
## Random Forest: Cement = 108.3       -9.3798927  -8.5883280  -7.98636151
## Random Forest: CoarseAggr = 938.2  -1.8091184  -0.6118426  -0.09574818
## Random Forest: FineAggr = 849       -2.6930467  -2.5316538  -2.17651207
## Random Forest: FlyAsh = 0           -0.8928709  -0.3885126   0.03285164
## Random Forest: Slag = 162.4         -1.8662139  -0.6068302   0.88977560
## Random Forest: Superplast = 0       -5.4679036  -4.0509231  -2.67999207
## Random Forest: Water = 203.5        -6.2797599  -4.9571307  -4.09345336
##                                           mean         q3         max
## Random Forest: Age = 3             -12.49151344 -12.1474218 -10.732151
## Random Forest: Cement = 108.3       -7.92286273  -7.3611992  -6.175855
## Random Forest: CoarseAggr = 938.2  -0.17384160   0.3981936   0.691545
## Random Forest: FineAggr = 849       -2.16068774  -1.9323358  -1.498059
## Random Forest: FlyAsh = 0            0.05526416   0.6137455   1.127896
## Random Forest: Slag = 162.4          0.65509943   1.9045264   2.557002
## Random Forest: Superplast = 0       -3.08369161  -2.0136204  -1.304557
## Random Forest: Water = 203.5        -4.19856148  -3.4590320  -2.213205
```

```r
plot(bd_rf)
```



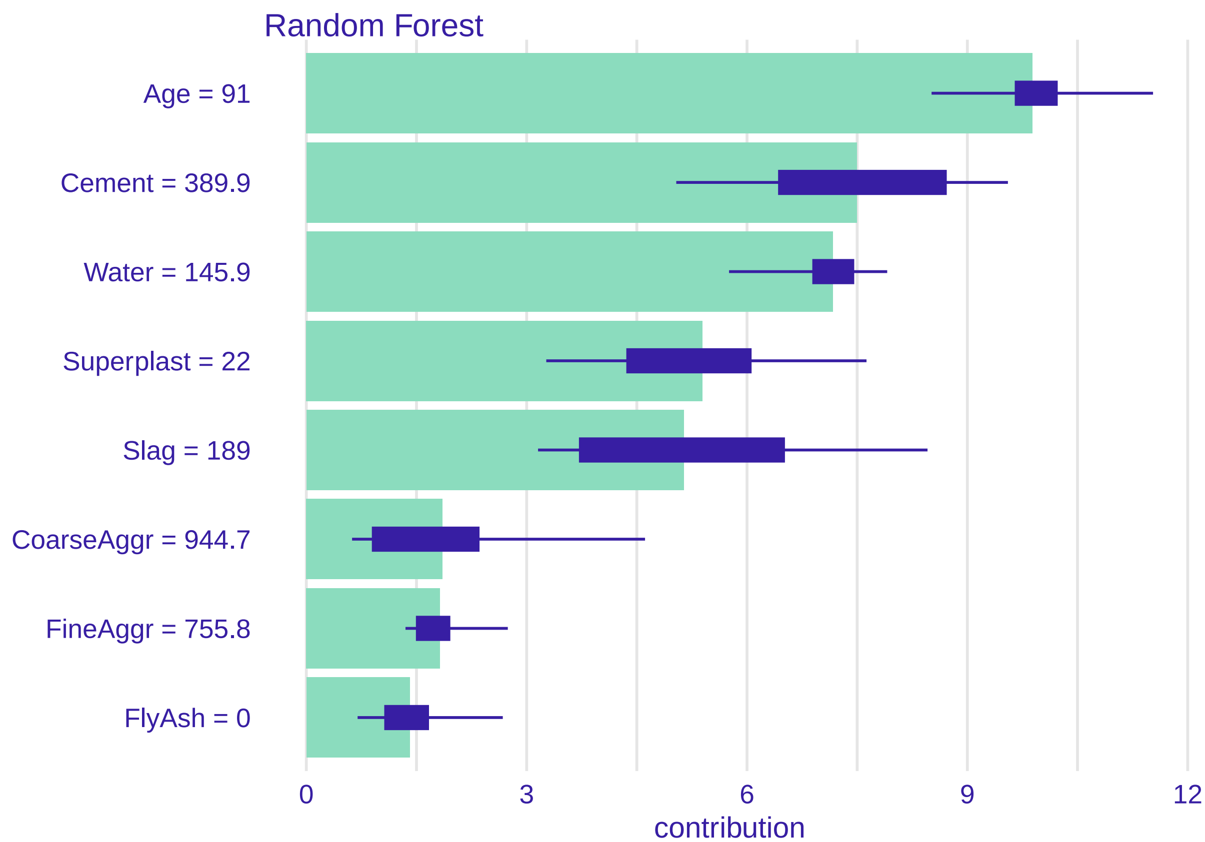
This plot shows that the features Age, Cement, Water, Superplast and FineAggr have the biggest impact (negatively).

```
bd_rf <- predict_parts(explainer = explainer_rf,
                 new_observation = highestStrength,
                            type = "shap")

bd_rf
```

```
##                                          min        q1   median     mean
## Random Forest: Age = 91            8.5128773 9.6649497 9.969725 9.886072
## Random Forest: Cement = 389.9      5.0367859 6.4426935 7.392447 7.493829
## Random Forest: CoarseAggr = 944.7 0.6232782 0.9117457 1.487802 1.854335
## Random Forest: FineAggr = 755.8    1.3503909 1.5116561 1.603108 1.820995
## Random Forest: FlyAsh = 0          0.6981845 1.0804958 1.382357 1.408065
## Random Forest: Slag = 189          3.1553095 3.7313676 4.193841 5.137639
## Random Forest: Superplast = 22     3.2671457 4.3762176 5.227560 5.394694
## Random Forest: Water = 145.9       5.7554232 6.9083336 7.374538 7.167695
##                                           q3       max
## Random Forest: Age = 91            10.210767 11.528239
## Random Forest: Cement = 389.9       8.700054  9.552234
## Random Forest: CoarseAggr = 944.7   2.339150  4.611715
## Random Forest: FineAggr = 755.8     1.941480  2.743423
## Random Forest: FlyAsh = 0           1.651160  2.675339
## Random Forest: Slag = 189           6.497144  8.457661
## Random Forest: Superplast = 22      6.042901  7.627039
## Random Forest: Water = 145.9        7.440504  7.908843
```

```r
plot(bd_rf)
```



This plot shows that all features have a good contribution towards Strength. Here again Age, Water and Cement have the highest impact but Water is over Cement even though they value es very similar. Also Slag is more significant here.

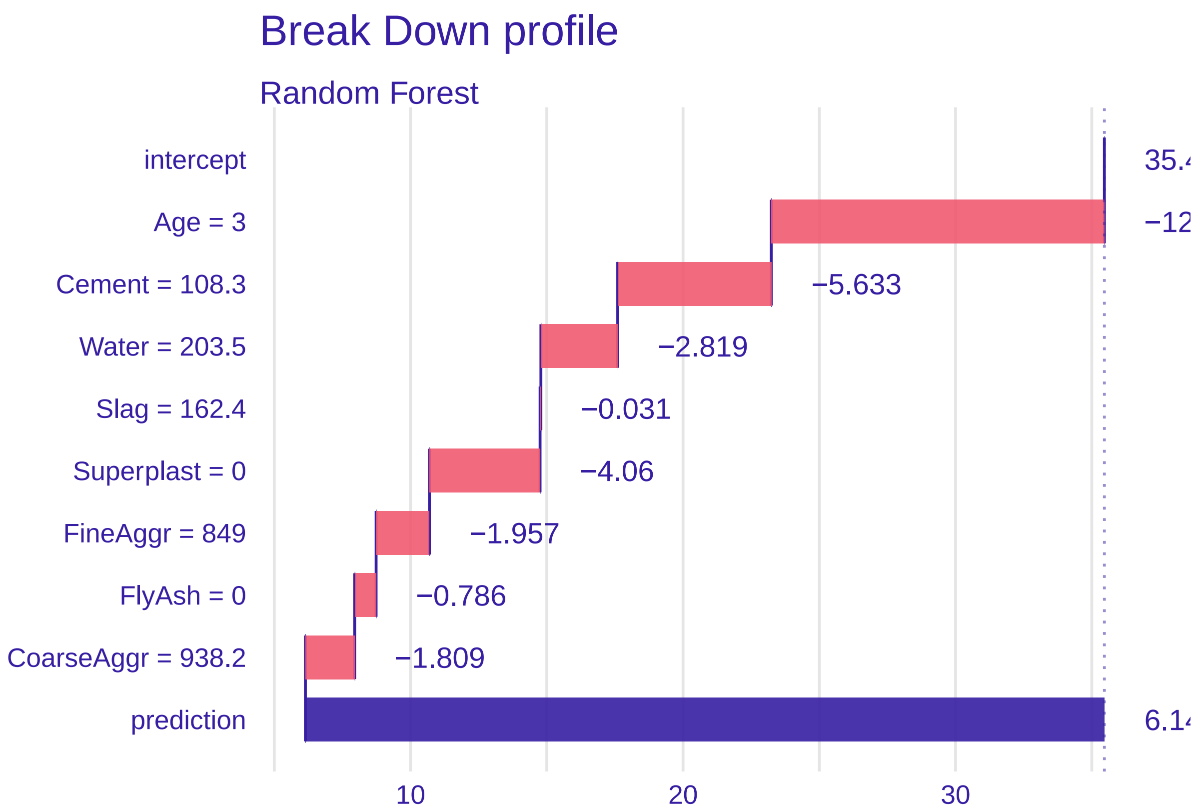
b. Explain the predictions using Break-down plots.

```r
bd_rf <- predict_parts(explainer = explainer_rf,
                 new_observation = lowestStrength,
                          type = "break_down")

bd_rf
```

```
##                                   contribution
## Random Forest: intercept                35.462
## Random Forest: Age = 3                 -12.227
## Random Forest: Cement = 108.3           -5.633
## Random Forest: Water = 203.5            -2.819
## Random Forest: Slag = 162.4             -0.031
## Random Forest: Superplast = 0           -4.060
## Random Forest: FineAggr = 849           -1.957
## Random Forest: FlyAsh = 0               -0.786
## Random Forest: CoarseAggr = 938.2       -1.809
## Random Forest: prediction                6.142
```

```r
plot(bd_rf)
```



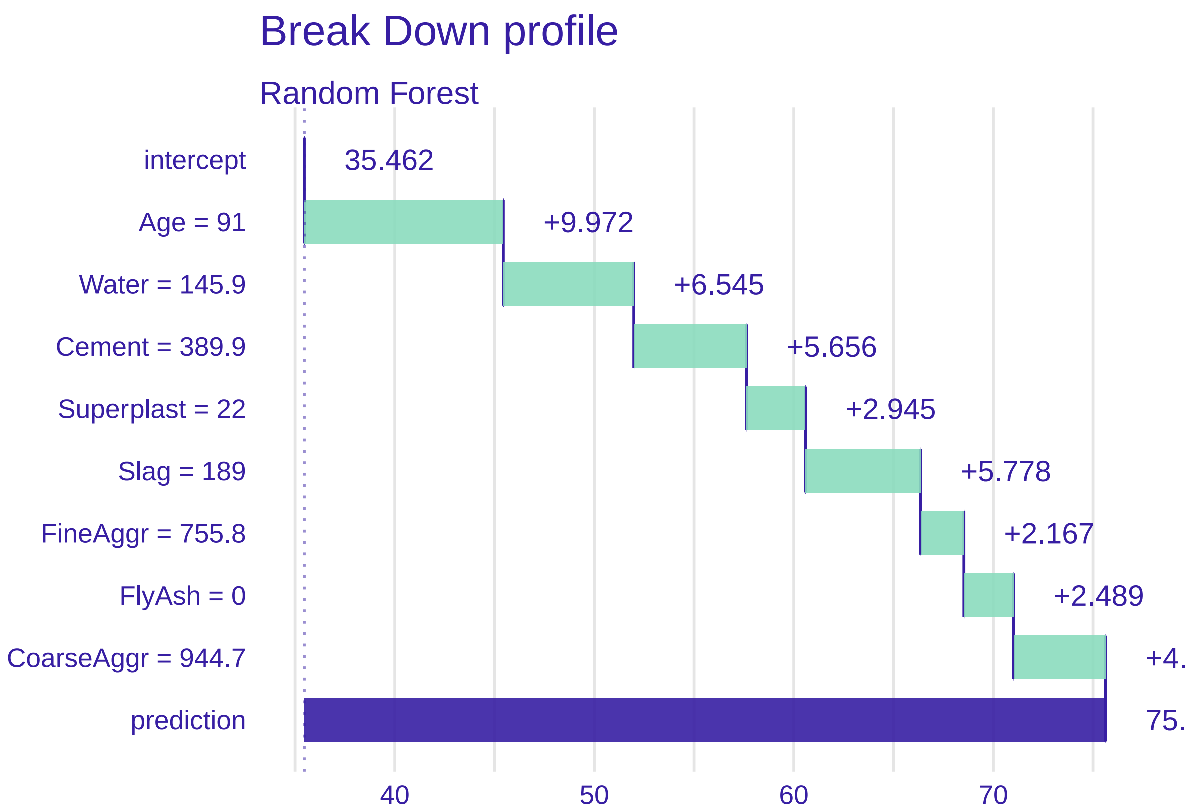
Here the plot shows that Age, Cement, Superplast and Water have a significant impact on the Strength. This means that we can focus on optimizing these input variables to achieve the desired Strength.

```r
bd_rf <- predict_parts(explainer = explainer_rf,
                       new_observation = highestStrength,
                                  type = "break_down")

bd_rf
```

```
##                                          contribution
## Random Forest: intercept                       35.462
## Random Forest: Age = 91                         9.972
## Random Forest: Water = 145.9                    6.545
## Random Forest: Cement = 389.9                   5.656
## Random Forest: Superplast = 22                  2.945
## Random Forest: Slag = 189                       5.778
## Random Forest: FineAggr = 755.8                 2.167
## Random Forest: FlyAsh = 0                       2.489
## Random Forest: CoarseAggr = 944.7               4.612
## Random Forest: prediction                      75.626
```

```r
plot(bd_rf)
```



Here we can see again that age, water and cement are very important.

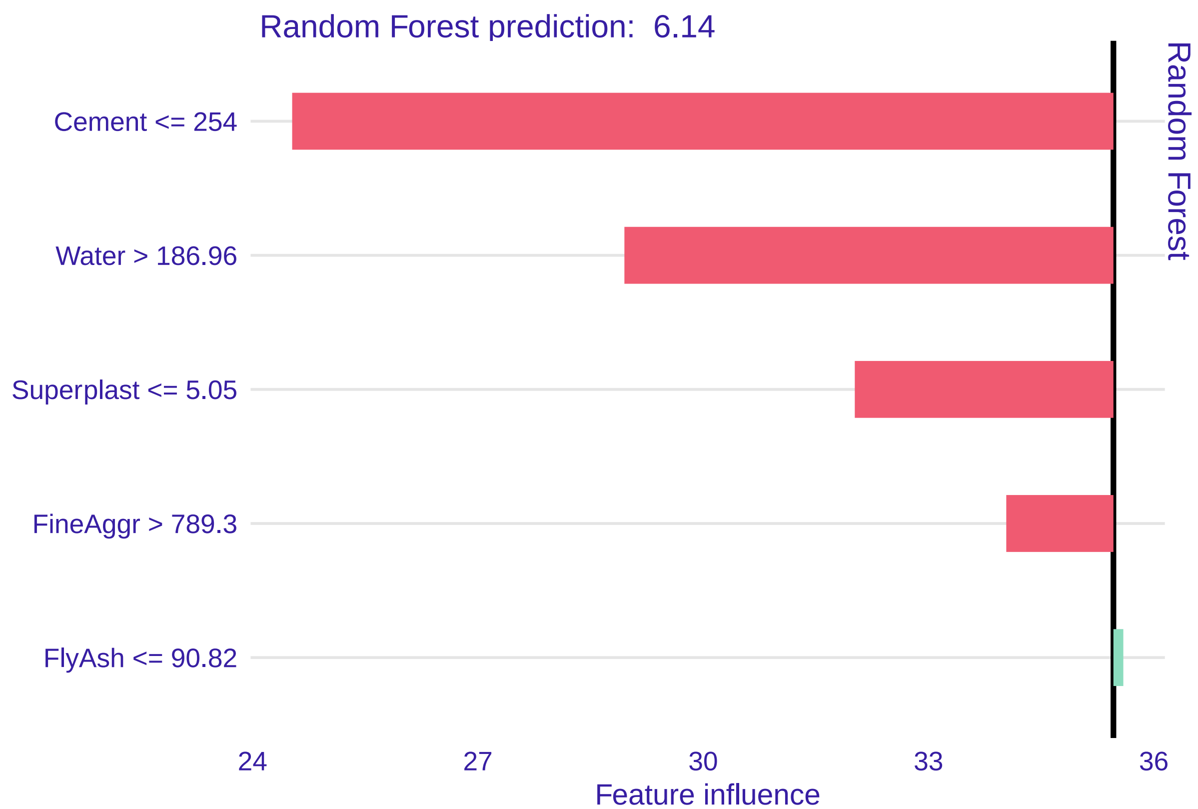
c. Explain the predictions using LIME.

```
bd_rf <- predict_surrogate(explainer = explainer_rf,
                new_observation = lowestStrength,
                           type = "localModel")

bd_rf
```

```
##     estimated           variable original_variable dev_ratio response
## 1  35.4624814       (Model mean)                   0.4461587
## 2  44.1606935        (Intercept)                   0.4461587
## 3 -10.9347468      Cement <= 254            Cement 0.4461587
## 4   0.1316822     FlyAsh <= 90.82           FlyAsh 0.4461587
## 5  -6.5112044      Water > 186.96            Water 0.4461587
## 6  -3.4436503 Superplast <= 5.05        Superplast 0.4461587
##   predicted_value         model
## 1        6.141686 Random Forest
## 2        6.141686 Random Forest
## 3        6.141686 Random Forest
## 4        6.141686 Random Forest
## 5        6.141686 Random Forest
## 6        6.141686 Random Forest
```

```r
plot(bd_rf)
```



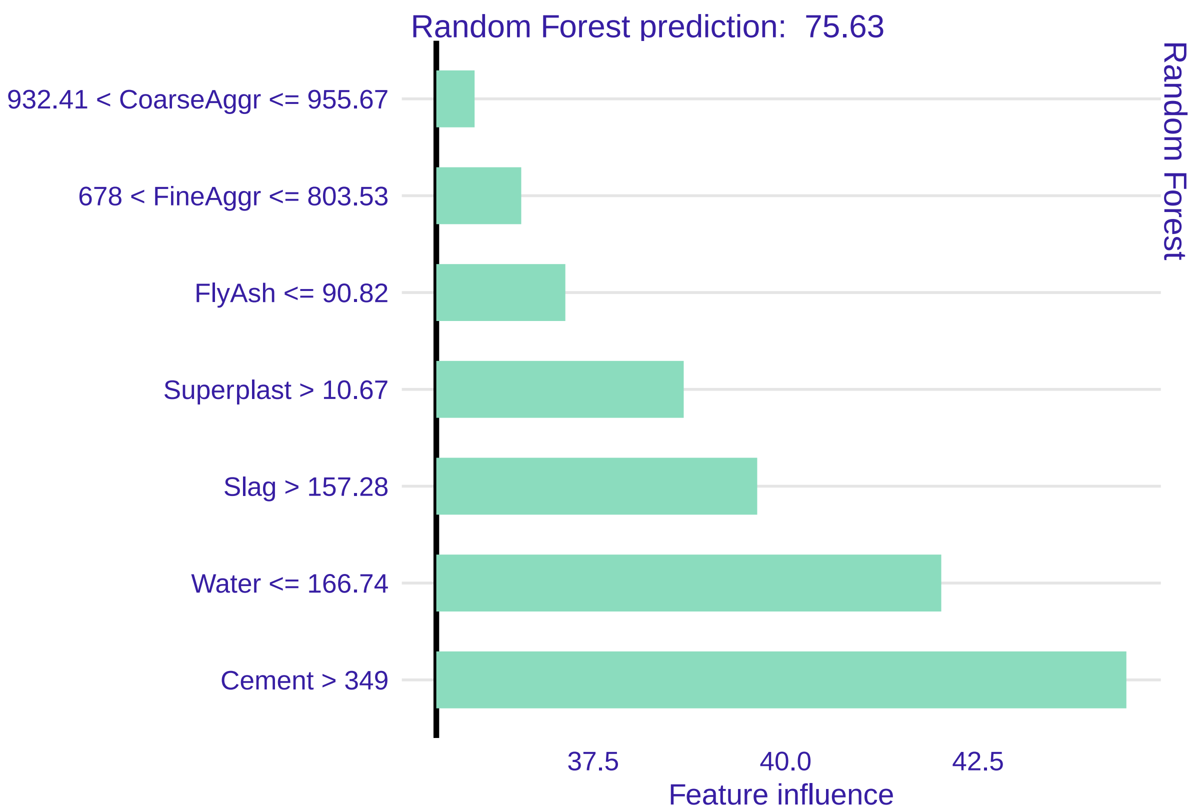
This plot that Cement, Water, Superplast and FineAggr have the biggest negative impact while Slag the biggest positive impact.

```r
bd_rf <- predict_surrogate(explainer = explainer_rf,
                  new_observation = highestStrength,
                           type = "localModel")

bd_rf
```

```
##   estimated        variable original_variable dev_ratio response
## 1 35.462481    (Model mean)                   0.4164575
## 2 31.499977     (Intercept)                   0.4164575
## 3  8.964384    Cement > 349            Cement 0.4164575
## 4  4.168543     Slag > 157.28            Slag 0.4164575
## 5  1.676273 FlyAsh <= 90.82            FlyAsh 0.4164575
## 6  6.559628 Water <= 166.74             Water 0.4164575
##   predicted_value         model
## 1        75.62581 Random Forest
## 2        75.62581 Random Forest
## 3        75.62581 Random Forest
## 4        75.62581 Random Forest
## 5        75.62581 Random Forest
## 6        75.62581 Random Forest
```

```r
plot(bd_rf)
```



The plot shows all having a positive impact, but Cement and Water and Superplast being the top ones.

d. Do the Individual conditional expectation (ICE) plot, or ceteris paribus plot

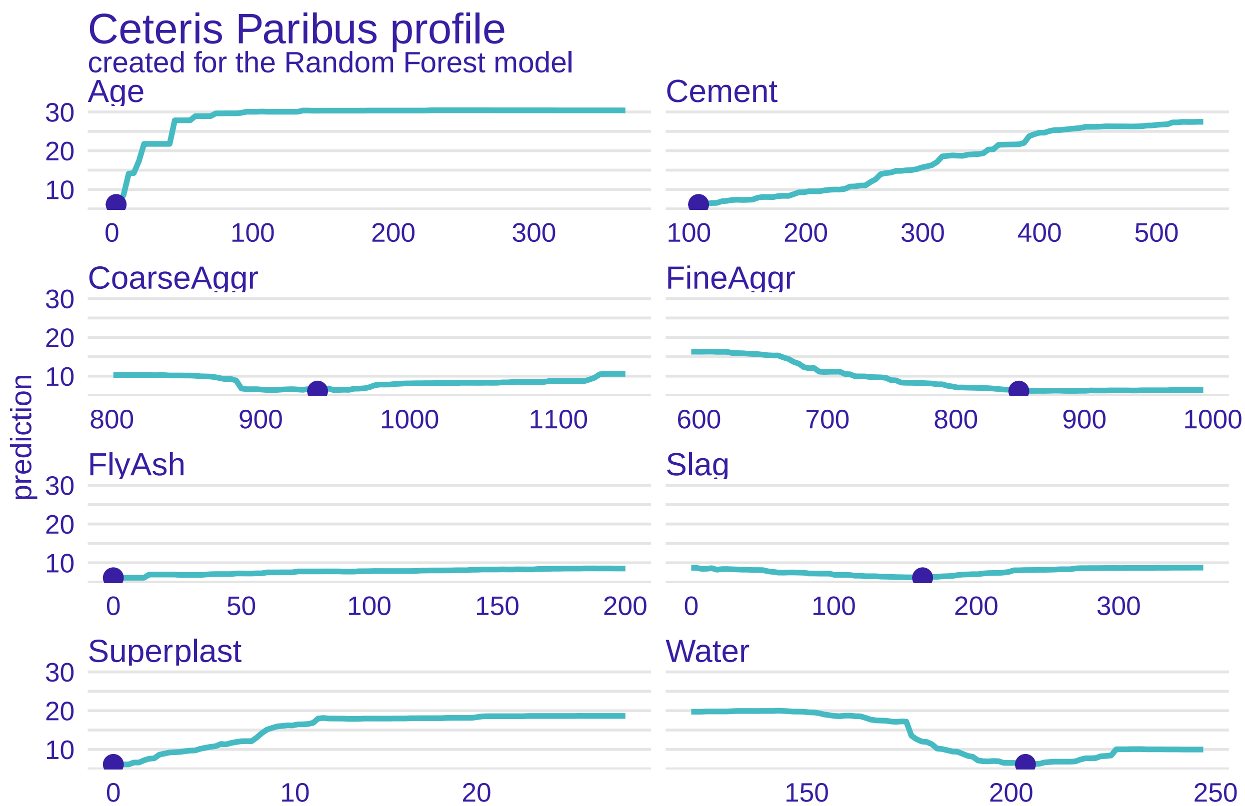
```r
cp_rf <- predict_profile(explainer = explainer_rf,
                new_observation = lowestStrength)

cp_rf
```

```
## Top profiles    :
##        Cement  Slag FlyAsh Water Superplast CoarseAggr FineAggr Age   _yhat_
## 689    102.00 162.4      0 203.5          0      938.2      849   3 6.209473
## 689.1  106.38 162.4      0 203.5          0      938.2      849   3 6.141686
## 689.2  110.76 162.4      0 203.5          0      938.2      849   3 6.145607
## 689.3  115.14 162.4      0 203.5          0      938.2      849   3 6.338861
## 689.4  119.52 162.4      0 203.5          0      938.2      849   3 6.516894
## 689.5  123.90 162.4      0 203.5          0      938.2      849   3 6.547655
##        _vname_ _ids_       _label_
## 689     Cement   689 Random Forest
## 689.1   Cement   689 Random Forest
## 689.2   Cement   689 Random Forest
## 689.3   Cement   689 Random Forest
## 689.4   Cement   689 Random Forest
```

```
## 689.5   Cement    689 Random Forest
##
##
## Top observations:
##     Cement  Slag FlyAsh Water Superplast CoarseAggr FineAggr Age    _yhat_
## 689  108.3 162.4      0 203.5          0      938.2      849   3 6.141686
##            _label_ _ids_
## 689 Random Forest     1
```

```
plot(cp_rf,facet_ncol=2)
```



The plots show that the predicted concrete strength is low when the content of its ingredients is very low as the points of the prediction seem to be the minimum of each curve..

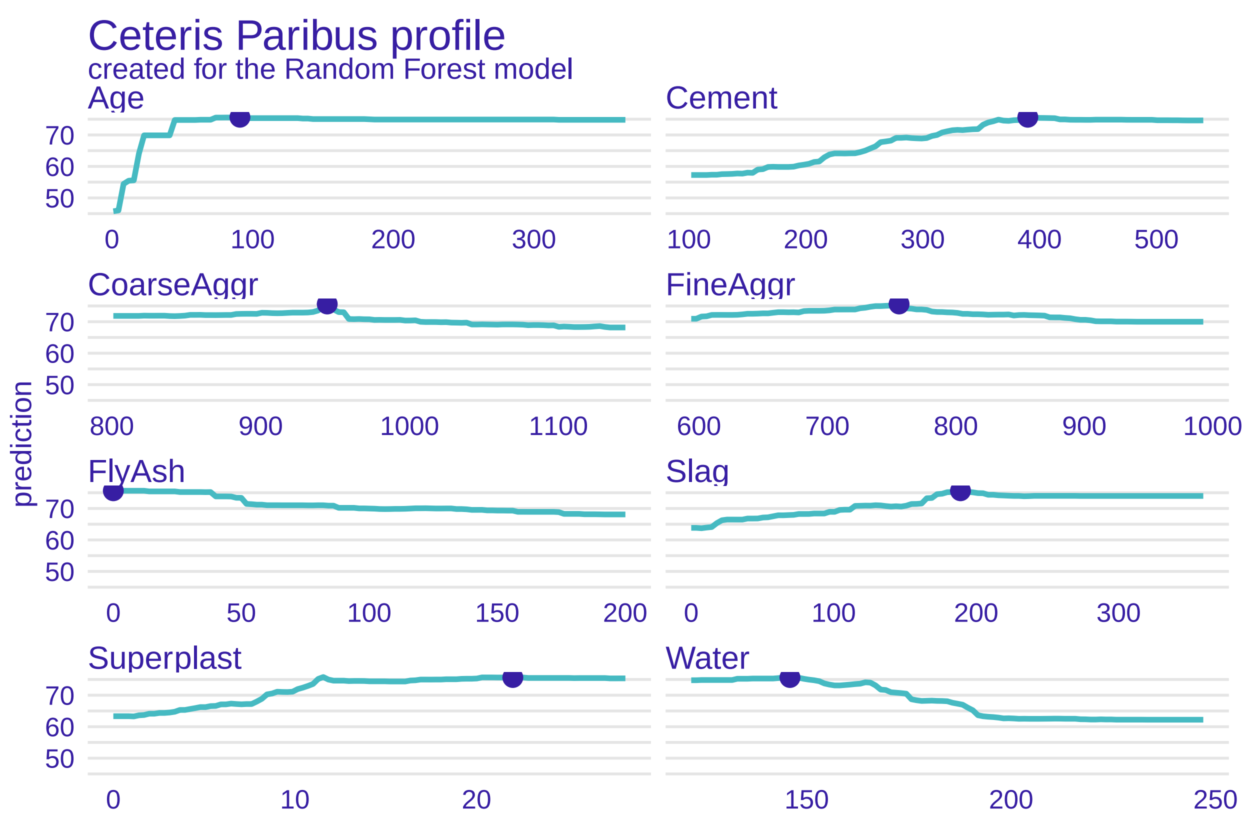
```
cp_rf <- predict_profile(explainer = explainer_rf,
                new_observation = highestStrength)

cp_rf
```

```
## Top profiles    :
##        Cement Slag FlyAsh Water Superplast CoarseAggr FineAggr Age    _yhat_
## 182    102.00  189      0 145.9         22      944.7    755.8  91 57.27172
## 182.1 106.38  189      0 145.9         22      944.7    755.8  91 57.27172
## 182.2 110.76  189      0 145.9         22      944.7    755.8  91 57.27172
## 182.3 115.14  189      0 145.9         22      944.7    755.8  91 57.27172
## 182.4 119.52  189      0 145.9         22      944.7    755.8  91 57.35989
```

```
## 182.5 123.90   189       0 145.9         22       944.7     755.8   91 57.35989
##       _vname_ _ids_        _label_
## 182    Cement   182 Random Forest
## 182.1  Cement   182 Random Forest
## 182.2  Cement   182 Random Forest
## 182.3  Cement   182 Random Forest
## 182.4  Cement   182 Random Forest
## 182.5  Cement   182 Random Forest
##
##
## Top observations:
##     Cement Slag FlyAsh Water Superplast CoarseAggr FineAggr Age    _yhat_
## 182  389.9  189      0 145.9         22      944.7    755.8  91 75.62581
##           _label_ _ids_
## 182 Random Forest     1
```

```
plot(cp_rf,facet_ncol=2)
```



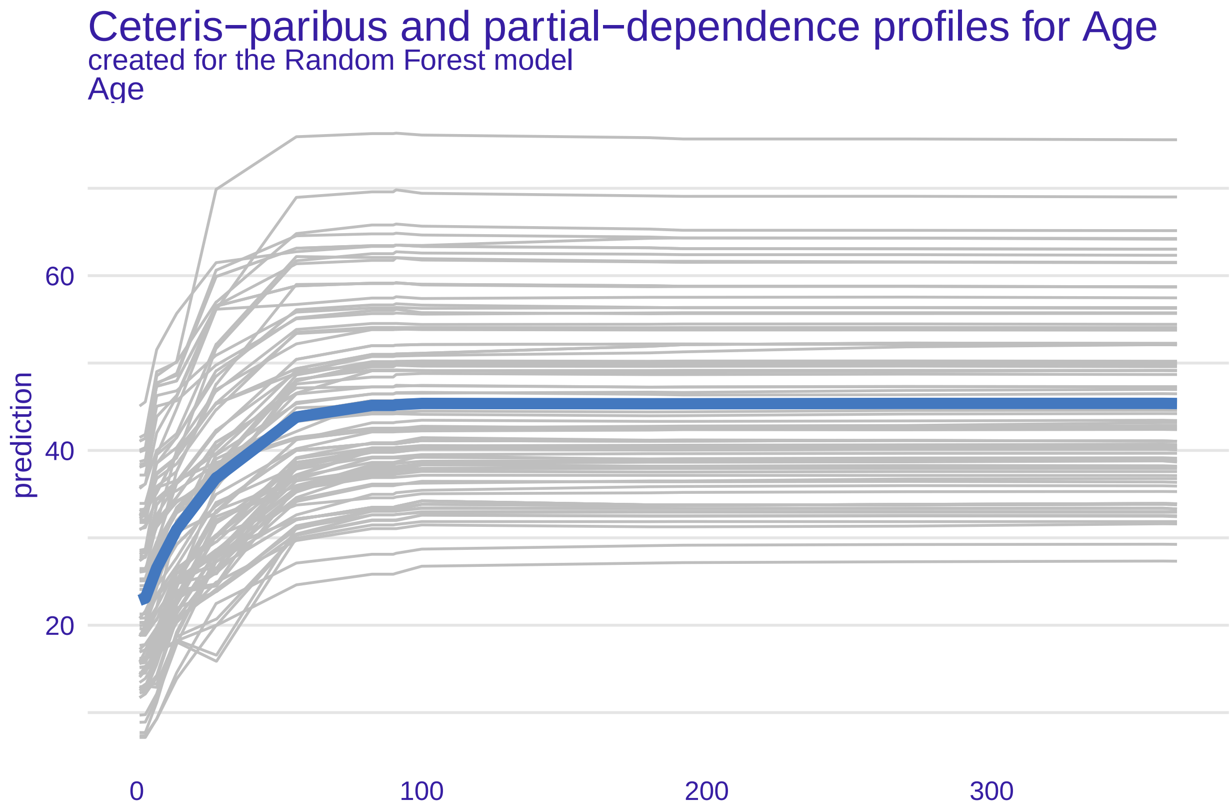
The highest strength however is reached by the maximum of each curve.

e. Plot in one graphic the Individual conditional expectation (ICE) plot for variable Age for eachcase in the test sample. Add the global Partial Depedence Plot.

```
mp_rf <- model_profile(explainer = explainer_rf,
  variables = "Age",
  N = 100,
```

```
  type = "partial"
)

plot(mp_rf, geom = "profiles") +
  ggtitle("Ceteris-paribus and partial-dependence profiles for Age")
```



The plot shows that the predicted Strength of concrete generally increases with increasing Age, but the relationship is complex and non-linear. The average effect of Age on Strength is positive, but the effect diminishes at higher ages.