Interpretability and Explainability in Machine Learning

Biel Caballero Vergés, Svenja Menzenbach and Kleber Enrique Reyes Illescas

2023-12-29

Data preperation

```r
set.seed(42)

concrete_copy <- concrete

sample <- sample(nrow(concrete), 700)
train_set <- concrete_copy[sample,]
test_set <- concrete_copy[-sample,]

head(train_set)
```

```
##     Cement   Slag FlyAsh  Water Superplast CoarseAggr FineAggr Age Strength
## 561 220.80 147.20   0.00 185.70       0.00    1055.00   744.30  28 25.74503
## 321 249.10   0.00  98.75 158.11      12.80     987.76   889.01  14 28.68220
## 634 275.00   0.00   0.00 183.00       0.00    1088.00   808.00   7 14.20321
## 49  237.50 237.50   0.00 228.00       0.00     932.00   594.00   7 26.25800
## 24  139.60 209.40   0.00 192.00       0.00    1047.00   806.90 180 44.20782
## 356 277.19  97.82  24.46 160.70      11.19    1061.70   782.46  14 47.71174
```

```r
head(test_set)
```

```
##    Cement  Slag FlyAsh Water Superplast CoarseAggr FineAggr Age Strength
## 1   540.0   0.0      0   162        2.5     1040.0    676.0  28 79.98611
## 5   198.6 132.4      0   192        0.0      978.4    825.5 360 44.29608
## 6   266.0 114.0      0   228        0.0      932.0    670.0  90 47.02985
## 15  304.0  76.0      0   228        0.0      932.0    670.0  28 47.81378
## 17  139.6 209.4      0   192        0.0     1047.0    806.9  90 39.35805
## 29  427.5  47.5      0   228        0.0      932.0    594.0  28 37.42752
```

1. Fit a Random Forest

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

```r
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
```

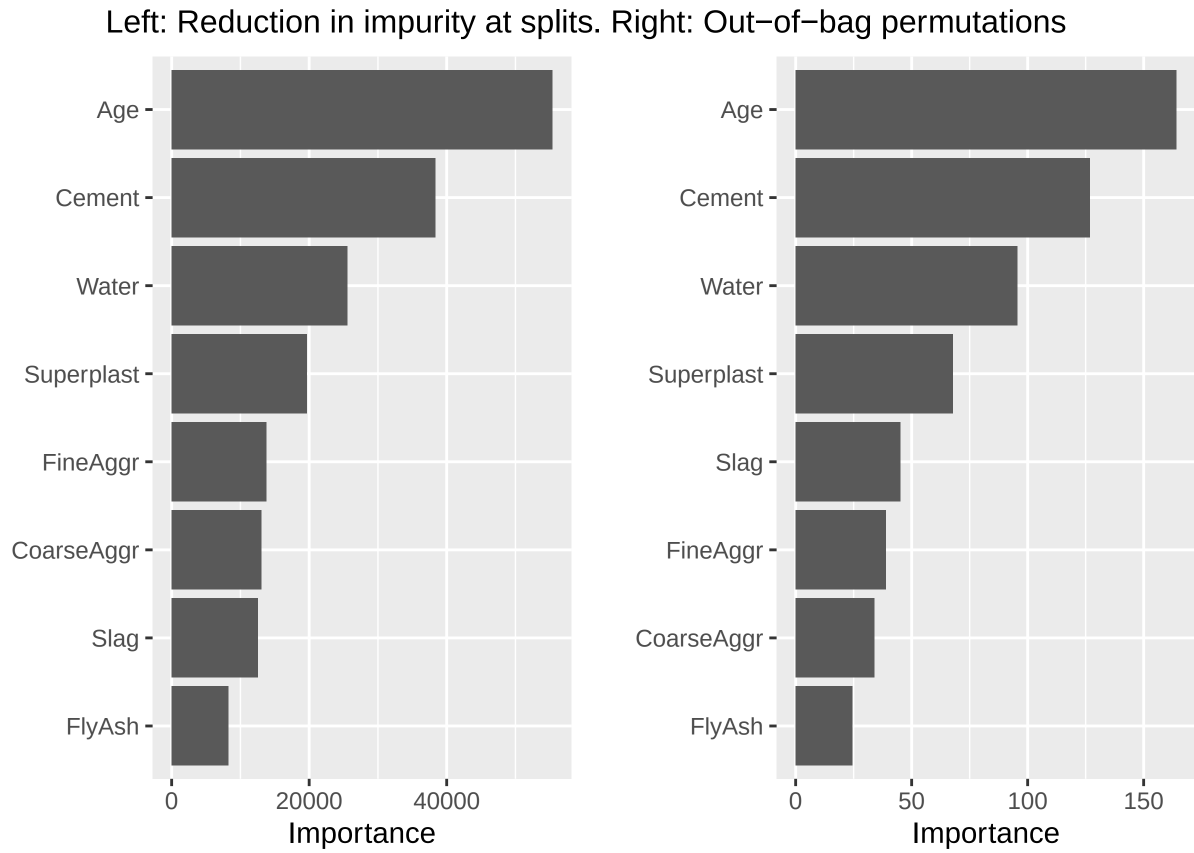
b. Compute the Variable Importance by out-of-bag random permutations.

```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
```

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")
```

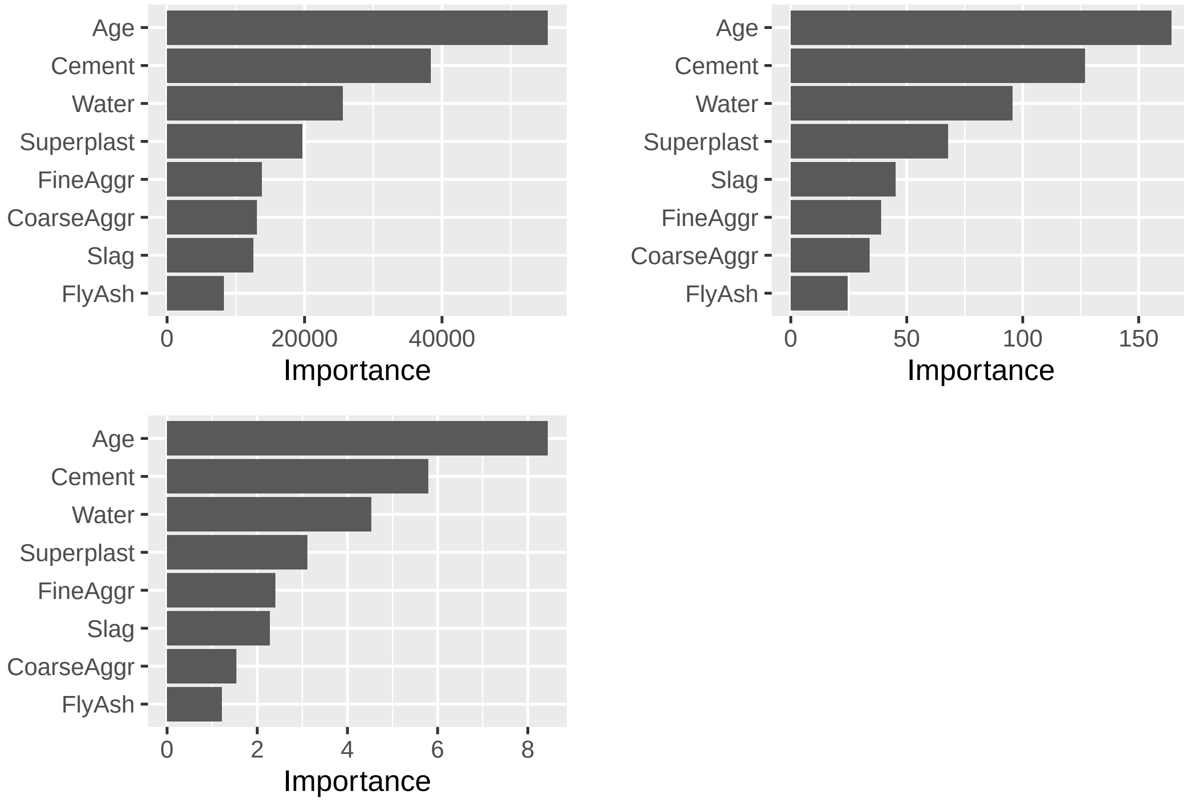


Both methods coincide in almost every parameter. Age, cement, water and superplast are the 4 most important variables without any doubt.

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

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



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))
```

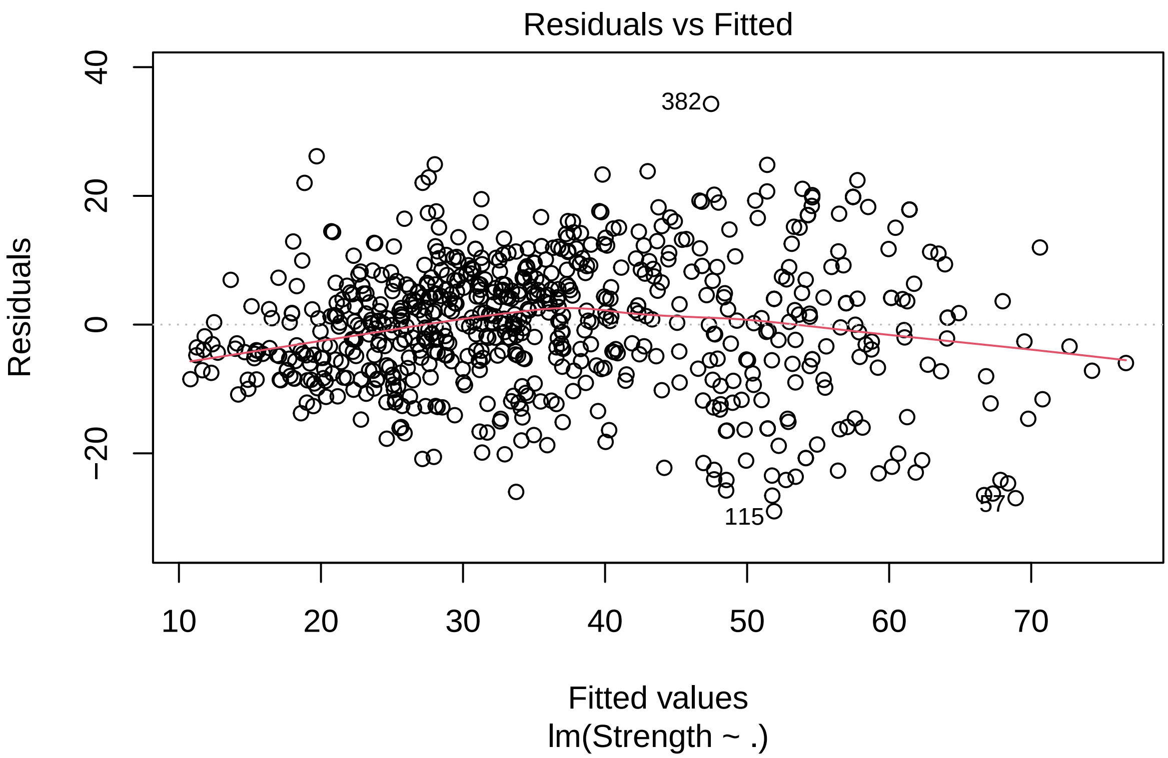
```
##
## Call:
## lm(formula = Strength ~ ., data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.003  -6.253   0.355   6.380  34.288
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) -37.444031  32.441685  -1.154  0.24882
## Cement        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 *
```

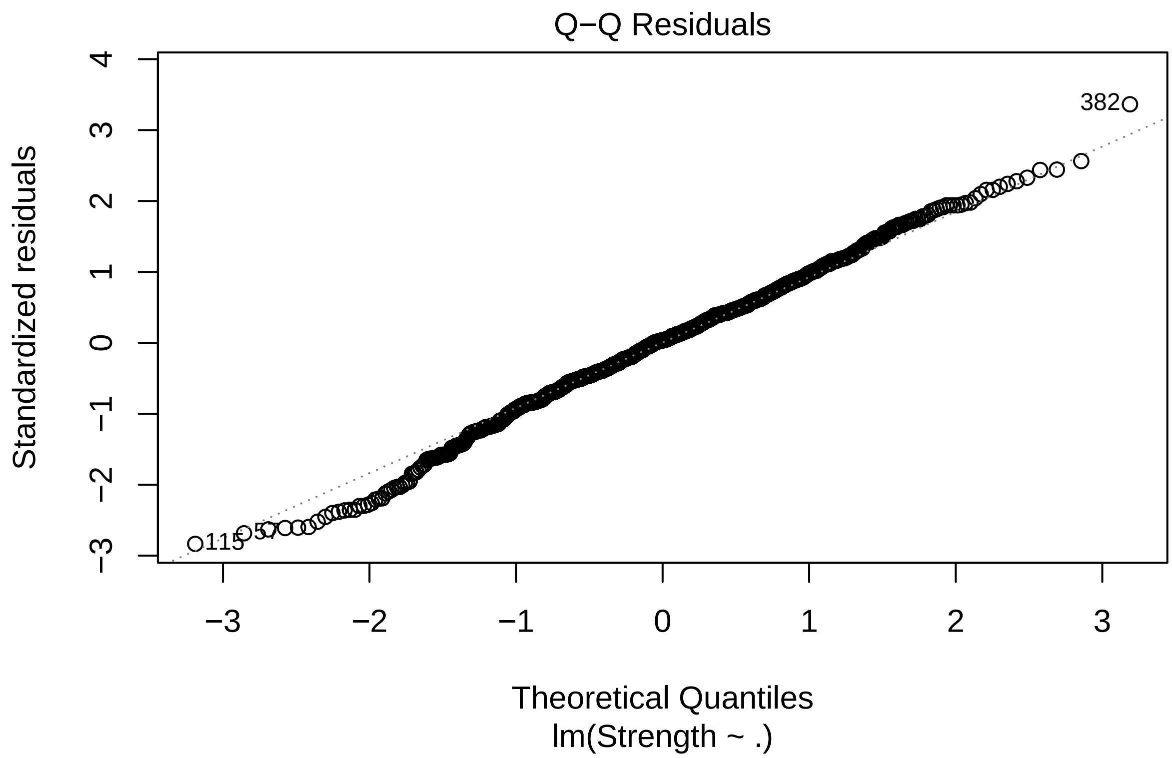
```
## FineAggr      0.025225   0.013078   1.929  0.05418 .
## Age           0.113435   0.006538  17.349  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.27 on 691 degrees of freedom
## Multiple R-squared:  0.6308, Adjusted R-squared: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

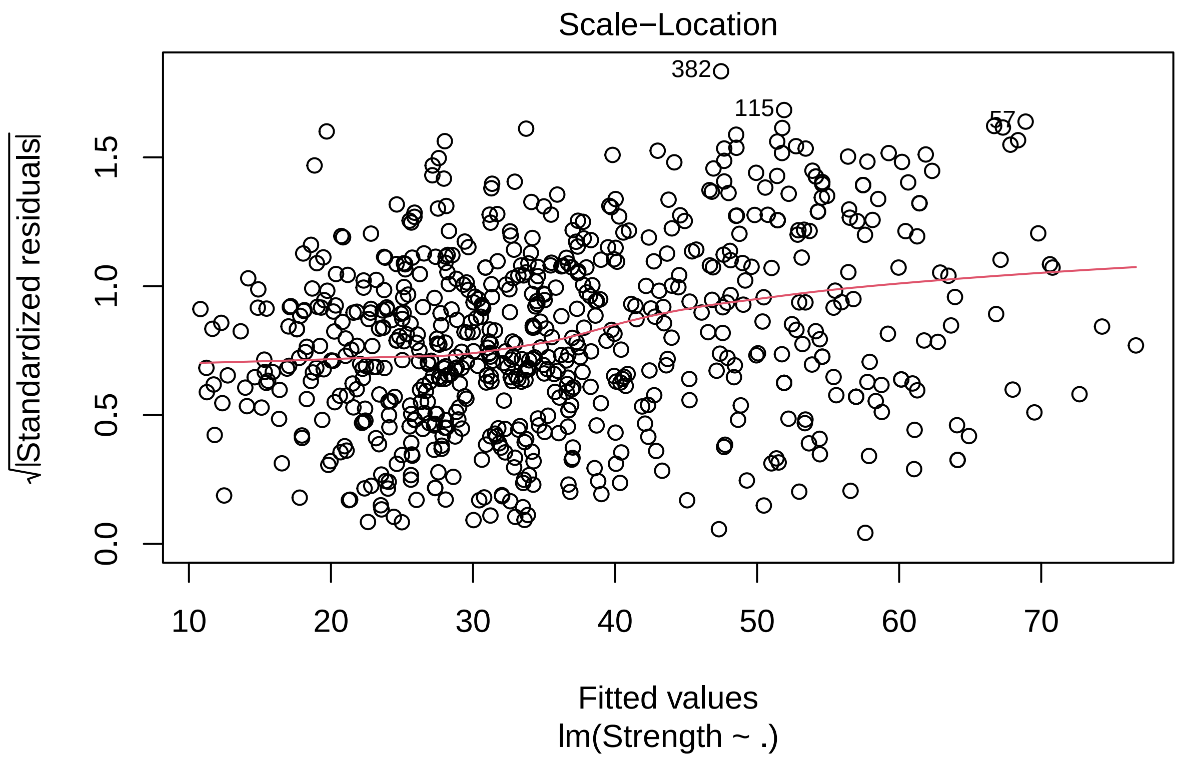
```
gam_strength <- gam(Strength ~ s(Cement) + s(Slag) + s(FlyAsh) + s(Water) + s(Superplast) + s(CoarseAggr
                 data = train_set)
(summ_gam_strength <- summary(gam_strength))
```

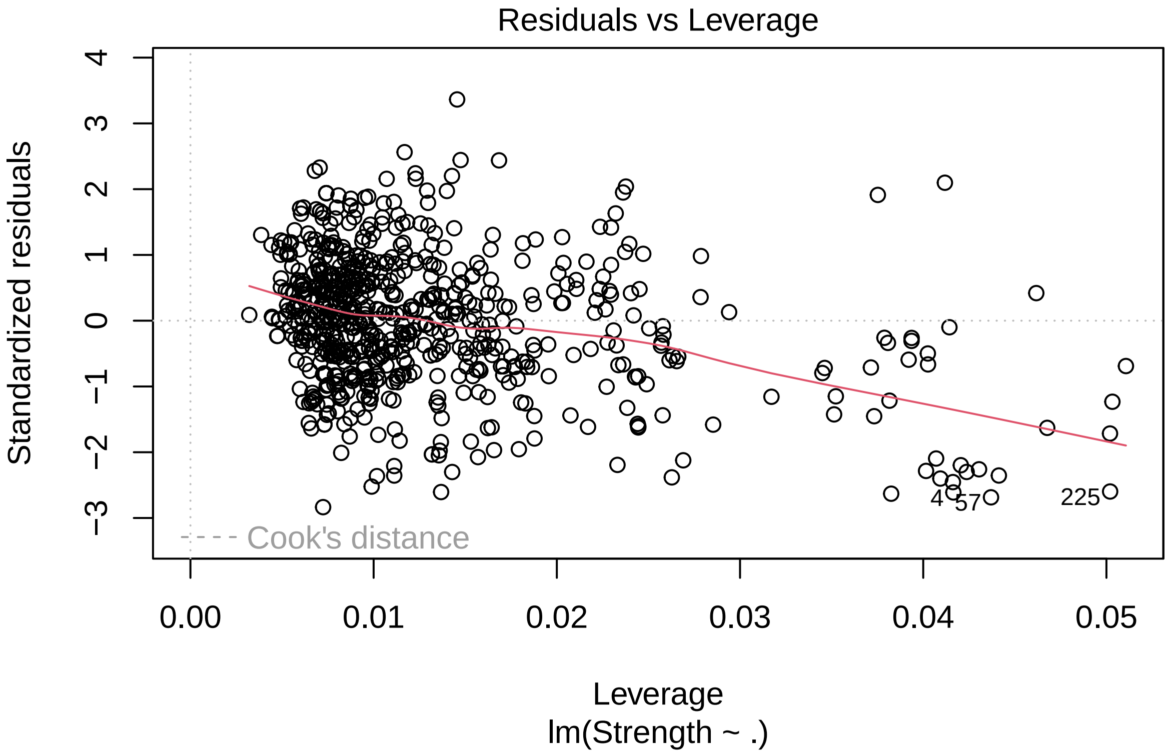
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Strength ~ s(Cement) + s(Slag) + s(FlyAsh) + s(Water) + s(Superplast) +
##     s(CoarseAggr) + s(FineAggr) + s(Age)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.0285     0.2035     177   <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)     7.798  8.615  28.737 < 2e-16 ***
## s(Slag)       8.240  8.810  14.212 < 2e-16 ***
## s(FlyAsh)     8.085  8.732   5.080 2.6e-06 ***
## s(Water)      8.506  8.916  18.461 < 2e-16 ***
## s(Superplast) 8.126  8.782   7.862 < 2e-16 ***
## s(CoarseAggr) 7.187  8.175   1.737  0.0789 .
## s(FineAggr)   8.556  8.932  11.849 < 2e-16 ***
## s(Age)        8.266  8.725 237.356 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.897   Deviance explained = 90.7%
## GCV = 32.004  Scale est. = 28.998    n = 700
```

```
plot(lm_strength)
```

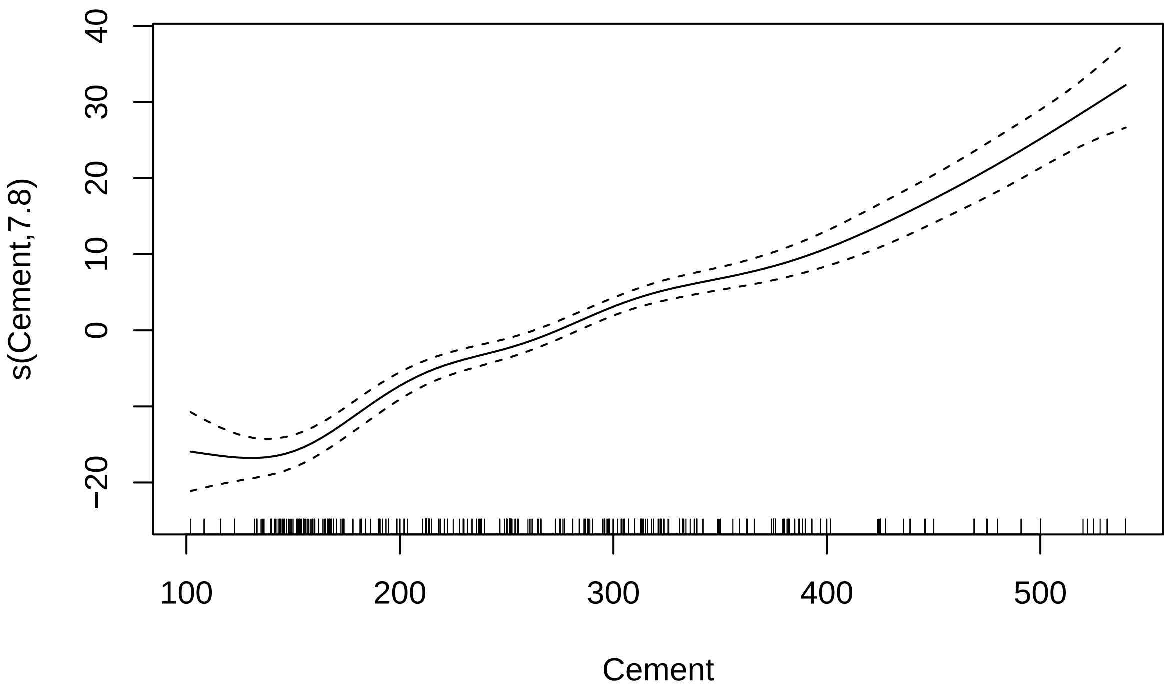

Residuals vs Fitted
Residuals
40
20
0
−20
382
115
57
10
20
30
40
50
60
70
Fitted values
lm(Strength ~ .)

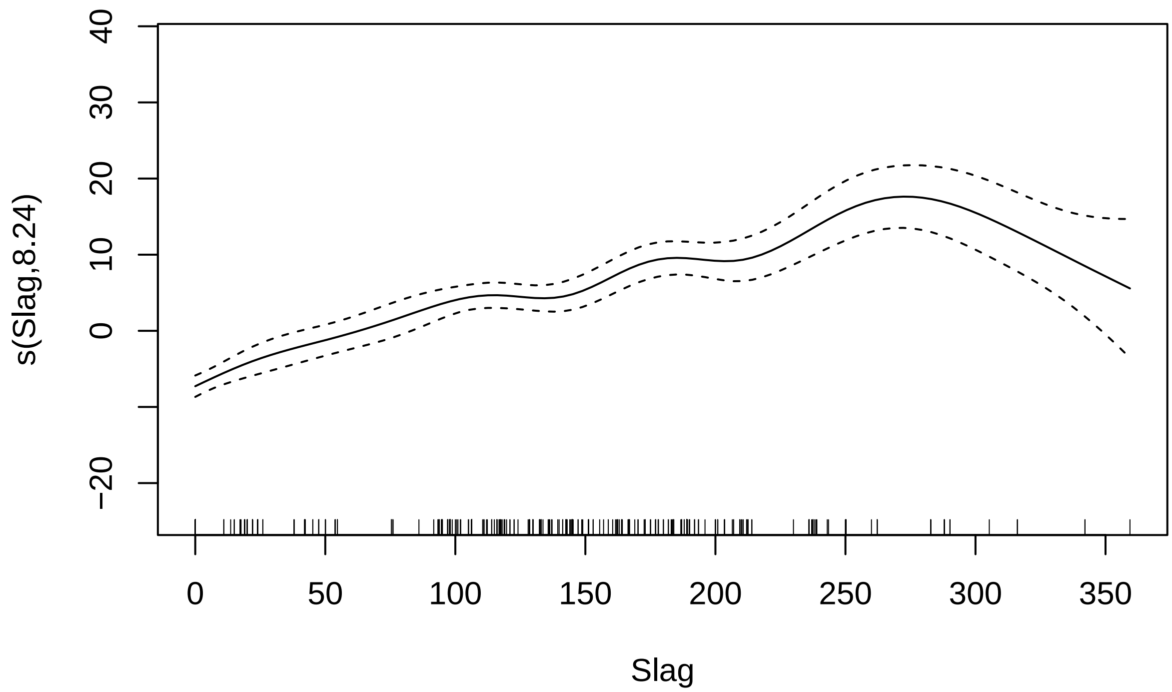

Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(Strength ~ .)
382
115
57
−3
−2
−1
0
1
2
3
4

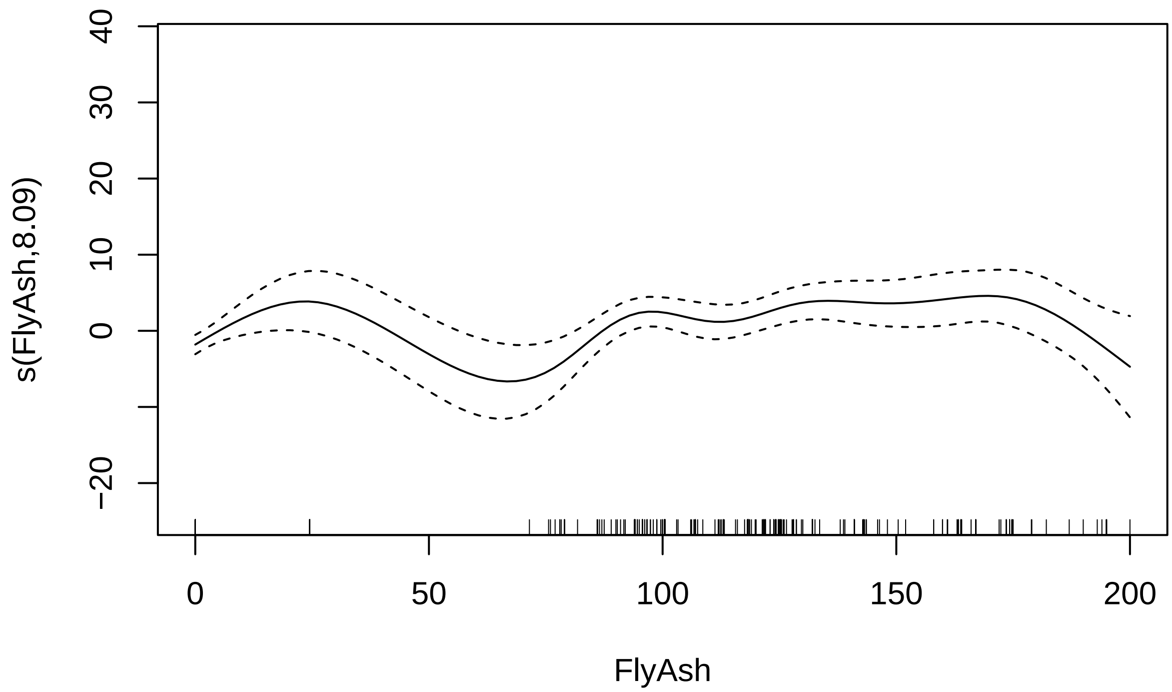

Scale−Location
382
115
57
√|Standardized residuals|
0.0
0.5
1.0
1.5
10
20
30
40
50
60
70
Fitted values
lm(Strength ~ .)



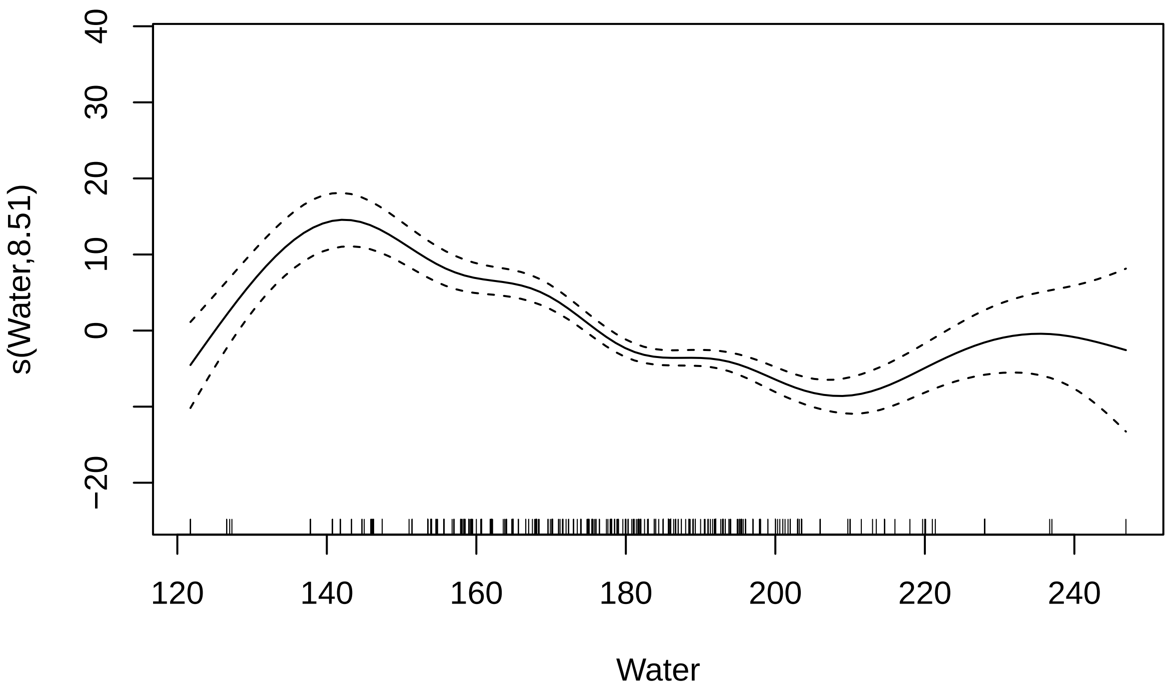

```
plot(gam_strength)
```


s(Cement,7.8)
40
30
20
10
0
−20
100
200
300
400
500
Cement

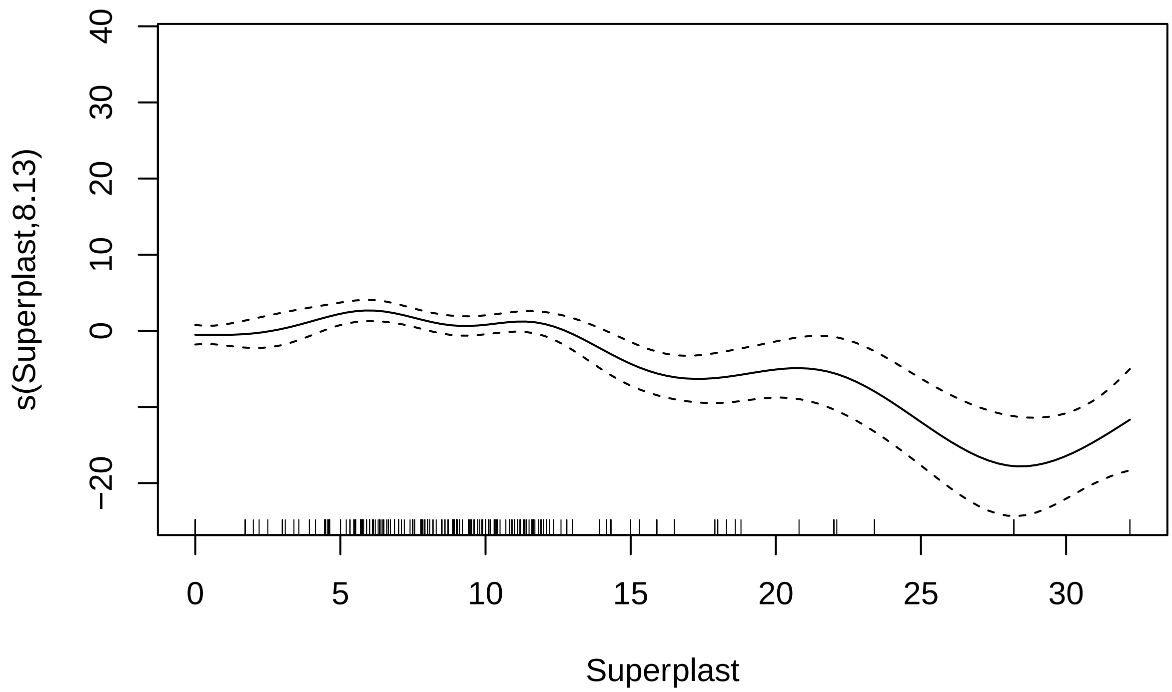

s(Slag,8.24)
40
30
20
10
0
−20
0
50
100
150
200
250
300
350
Slag



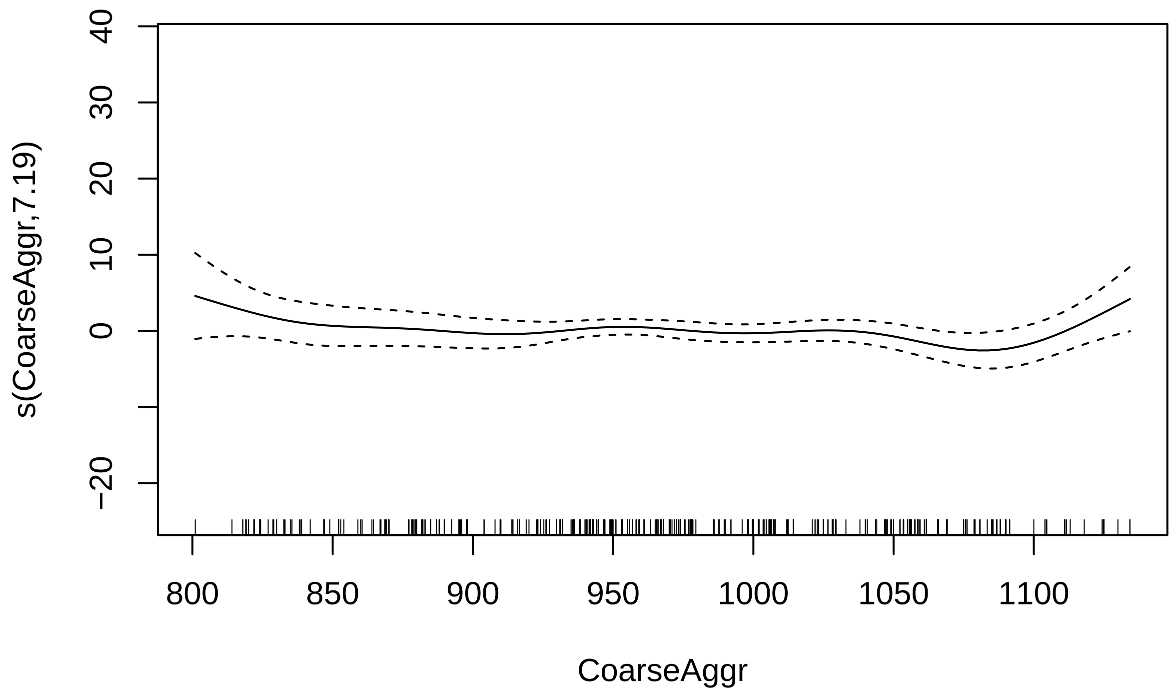
s(FlyAsh,8.09)
−20
0
10
20
30
40
0
50
100
150
200
FlyAsh



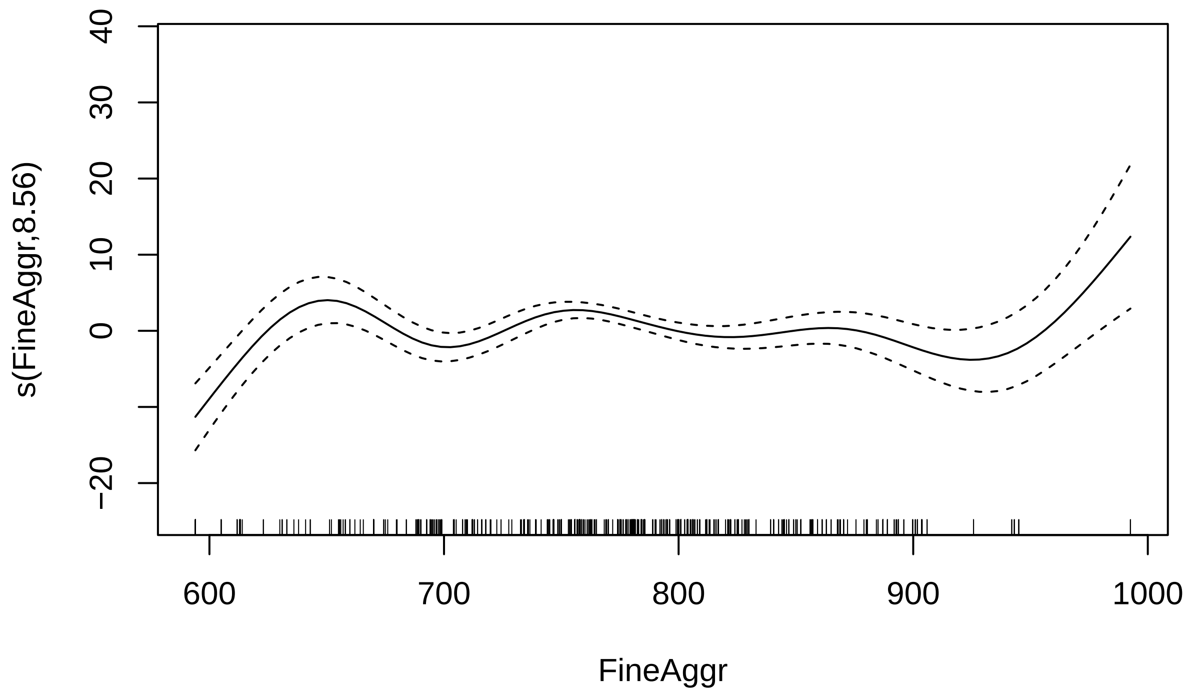
s(Water,8.51)
40
30
20
10
0
−20
120
140
160
180
200
220
240
Water

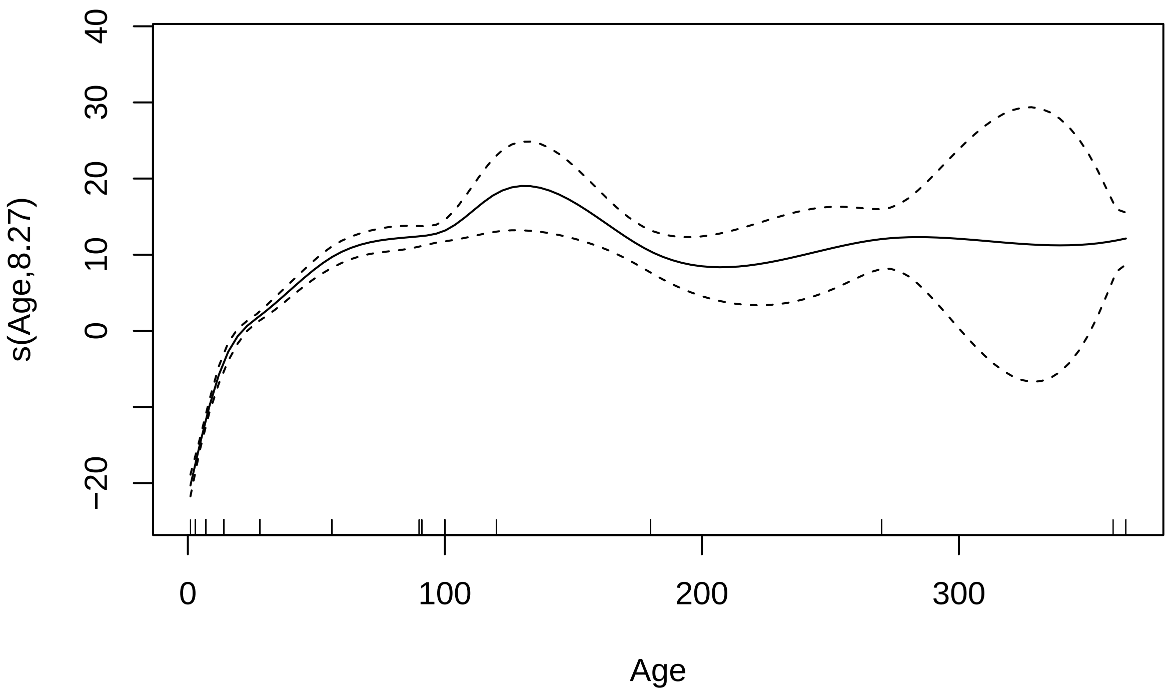


s(Superplast,8.13)
40
30
20
10
0
−20
0
5
10
15
20
25
30
Superplast



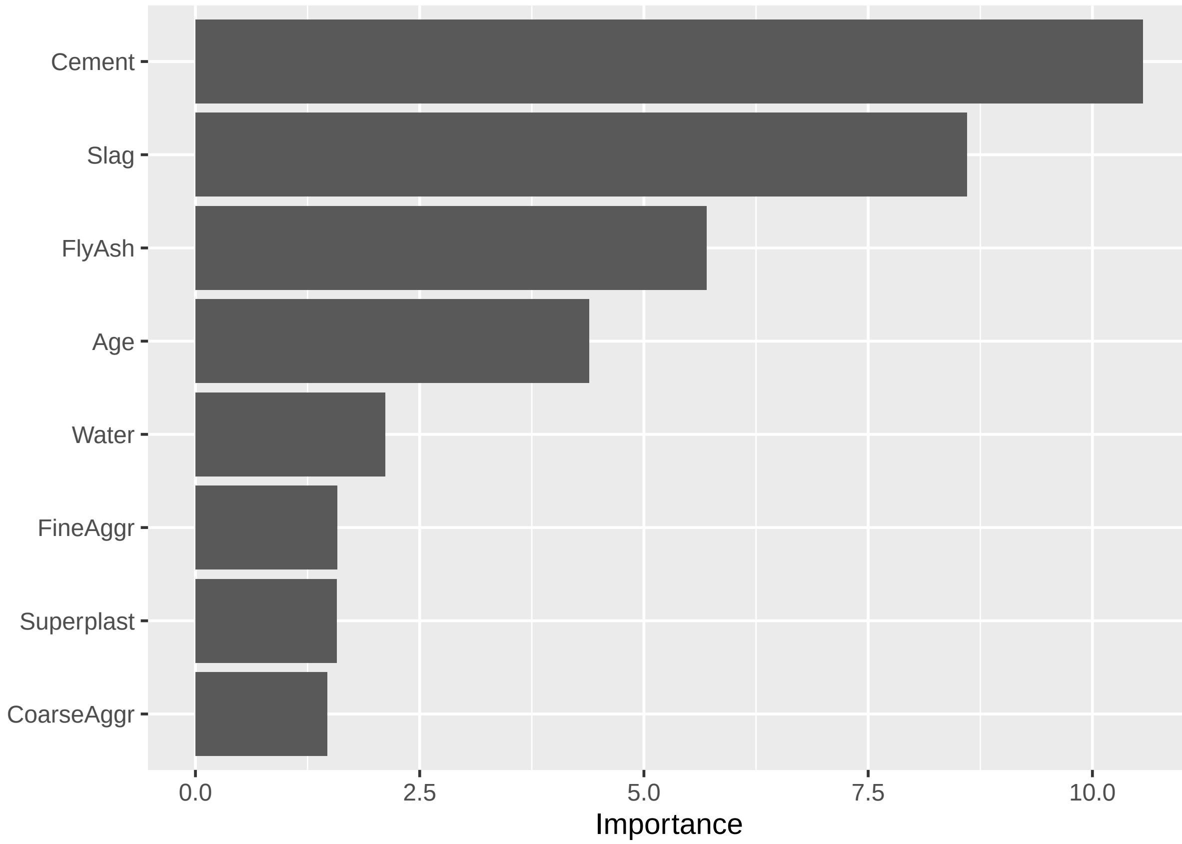
s(CoarseAggr,7.19)
40
30
20
10
0
−20
800
850
900
950
1000
1050
1100
CoarseAggr


s(FineAggr,8.56)
40
30
20
10
0
−20
600
700
800
900
1000
FineAggr

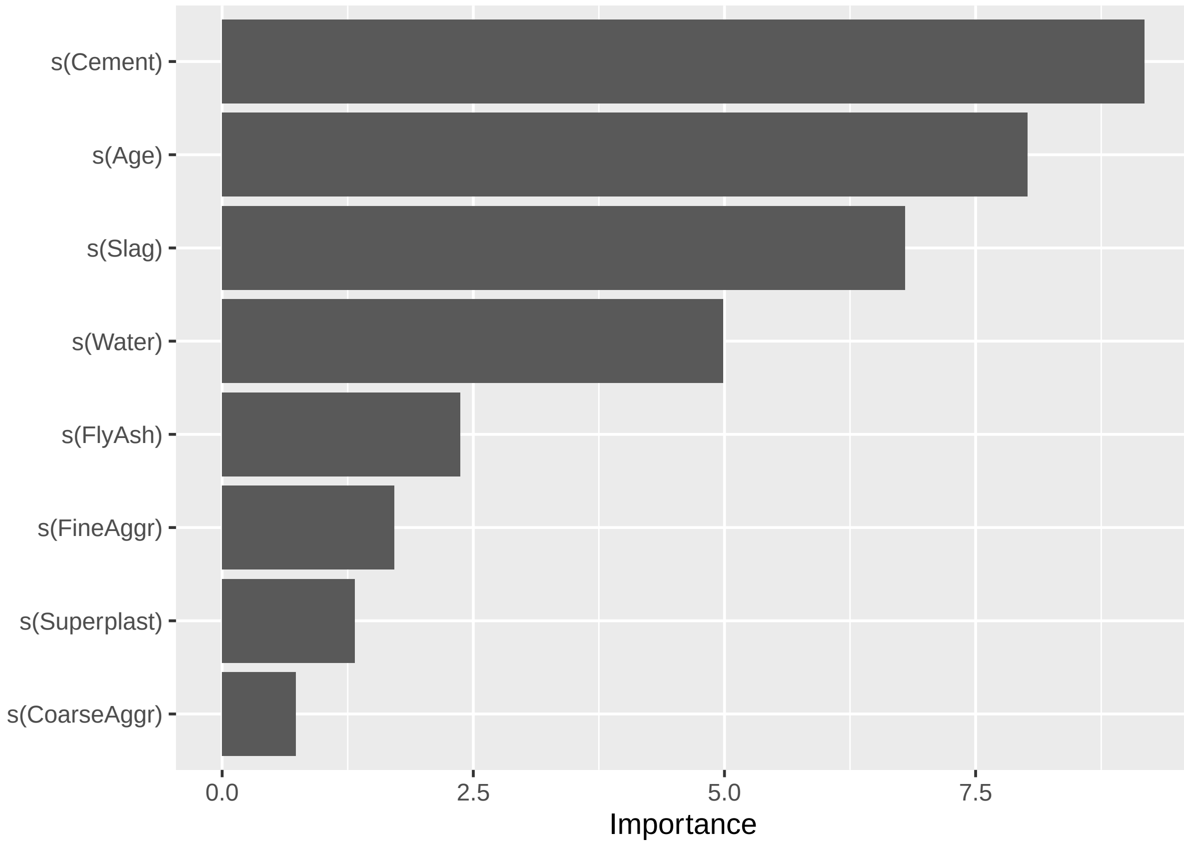


b. Compute the Variable Importance by Shappley values in the linear and gam fitted models. Compare your results with what you have learned before.

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

plot(lm_strength_shapley)
```



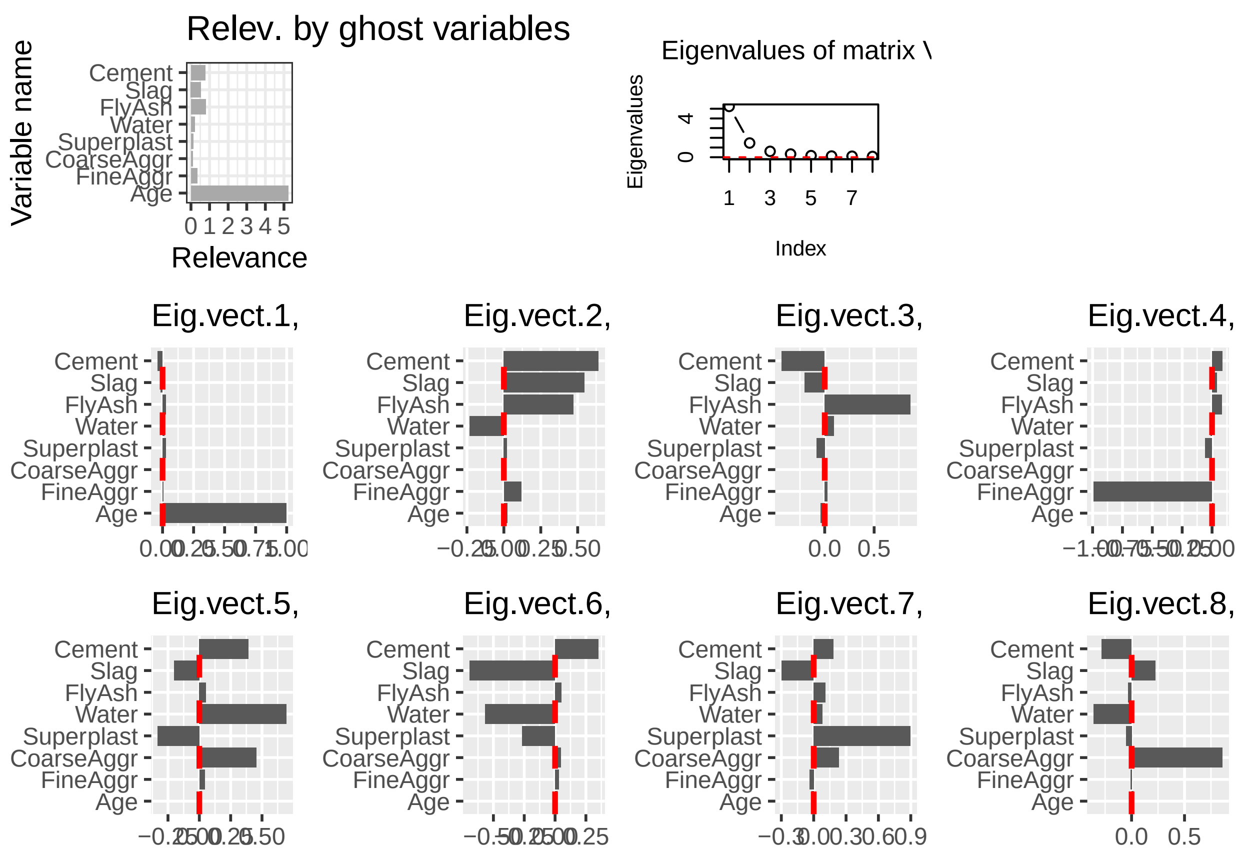

```
gam_strength_shapley <- vip(gam_strength, method="shap",
                  pred_wrapper=predict.gam,
                  train=train_set, # train set must be specified
                  newdata=test_set[,-9],
                  num_features = 8,
                  exact=TRUE)

plot(gam_strength_shapley)
```



3. Relevance by Ghost Variables

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

```
source("relev.ghost.var.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

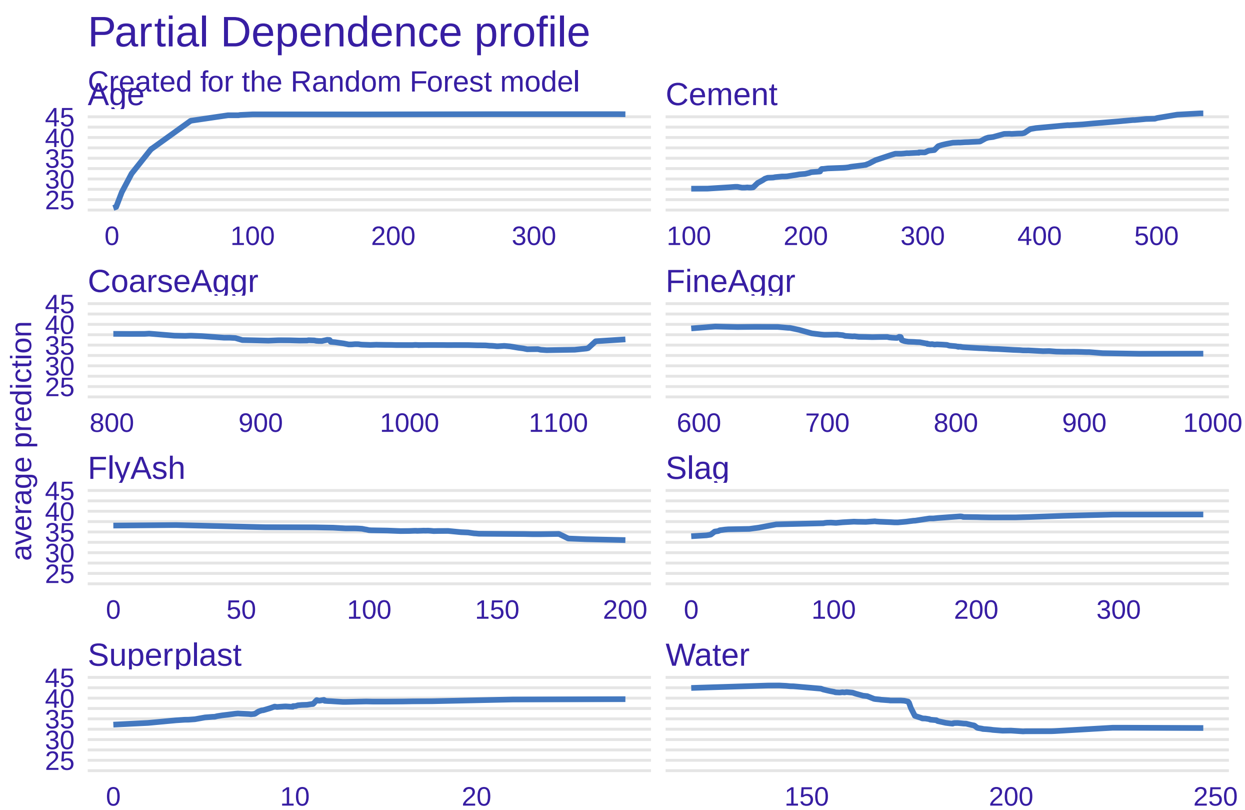
a. Compute Variable Importance by Random Permutations

```
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.

```
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)
```

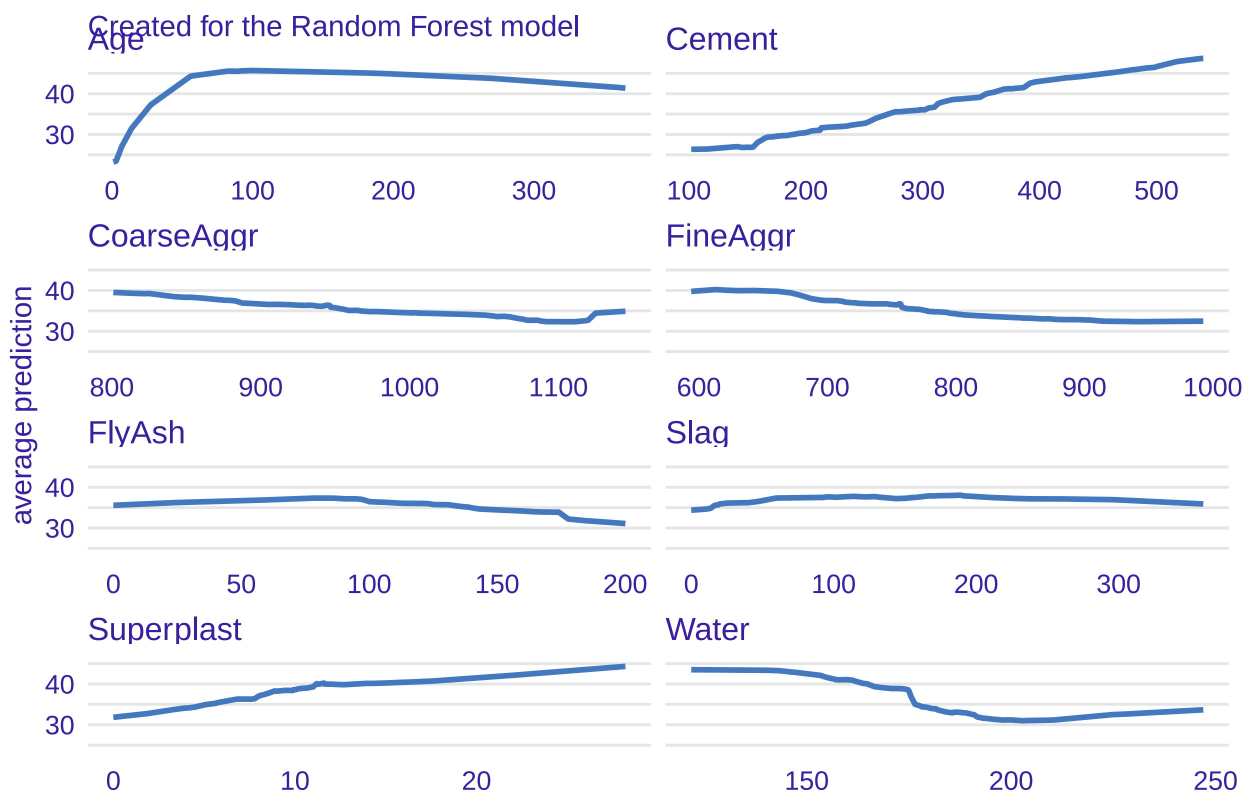


For Cement and CoarseAggr, the predicted Strength initially rises with an increase in these materials but eventually decreases after reaching an optimal point. This suggests an optimal quantity for both components, as excessive use could diminish Strength. On the other hand, FlyAsh and Slag show a consistent increase in predicted Strength with higher quantities, implying their positive impact on concrete Strength.

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

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

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

plot(CDP_rf, facet_ncol=2)
```



While cement and coarse aggregate initially boost concrete strength, their impact diminishes at higher proportions. Optimal dosages exist for these materials, as exceeding them may impair strength. In contrast, fly ash and slag consistently enhance concrete strength with increasing amounts.

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.

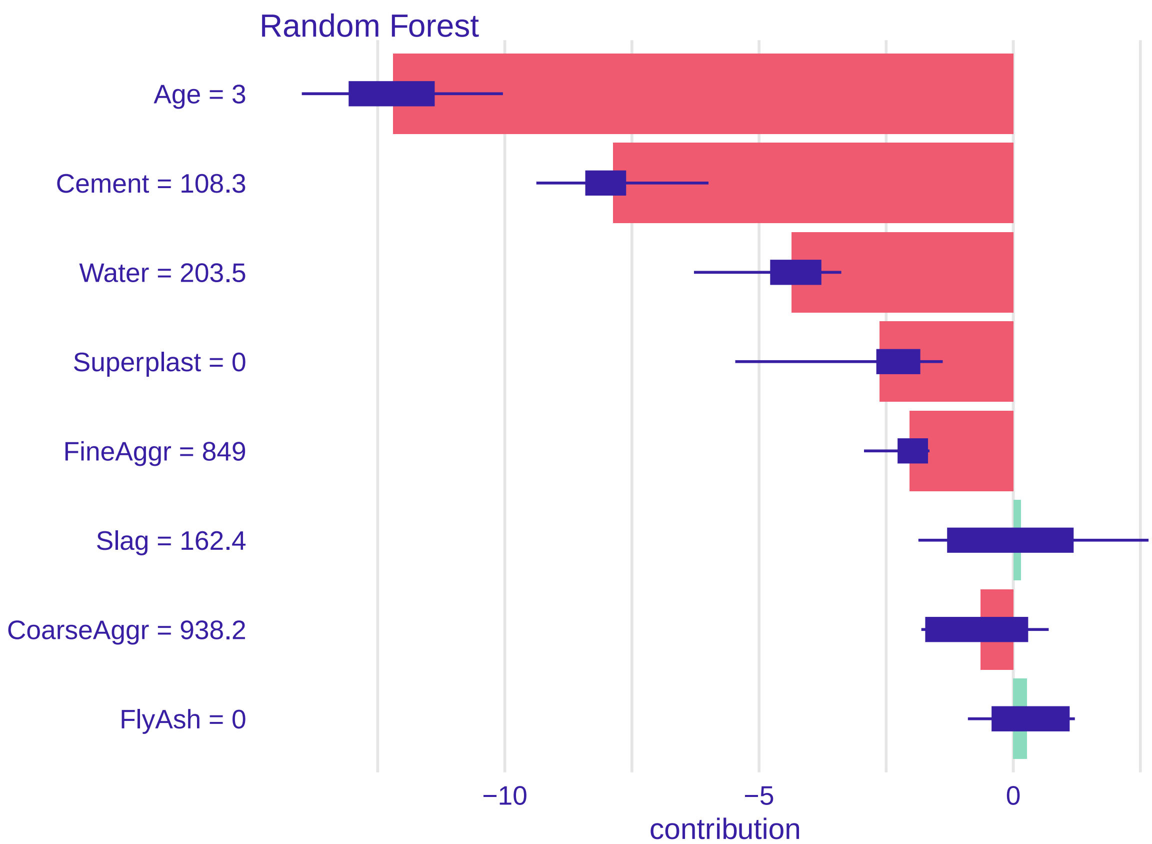
```r
lowestStrength = concrete[which.min(concrete$Strength), ]
highestStrength = concrete[which.max(concrete$Strength), ]
```

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            -13.9924970 -13.0435497 -12.2405352
## Random Forest: Cement = 108.3      -9.3798927  -8.3895787  -7.9576719
## Random Forest: CoarseAggr = 938.2  -1.8091184  -1.7040507  -0.6162180
## Random Forest: FineAggr = 849      -2.9360901  -2.2477249  -1.9731977
## Random Forest: FlyAsh = 0          -0.8928709  -0.3993512   0.2560205
## Random Forest: Slag = 162.4        -1.8662139  -1.2733588   0.2510276
## Random Forest: Superplast = 0      -5.4679036  -2.6657981  -2.1543975
## Random Forest: Water = 203.5       -6.2797599  -4.7535163  -4.0784268
##                                          mean          q3         max
## Random Forest: Age = 3            -12.1985213 -11.4068046 -10.0374747
## Random Forest: Cement = 108.3      -7.8722189  -7.6424840  -5.9945213
## Random Forest: CoarseAggr = 938.2  -0.6467063   0.2645544   0.6963735
## Random Forest: FineAggr = 849      -2.0383124  -1.7054286  -1.6484831
## Random Forest: FlyAsh = 0           0.2722941   1.0804958   1.2104952
## Random Forest: Slag = 162.4         0.1474898   1.1590417   2.6599450
## Random Forest: Superplast = 0      -2.6266321  -1.8566915  -1.3877501
## Random Forest: Water = 203.5       -4.3581878  -3.8026039  -3.3825910
```

```r
plot(bd_rf)
```

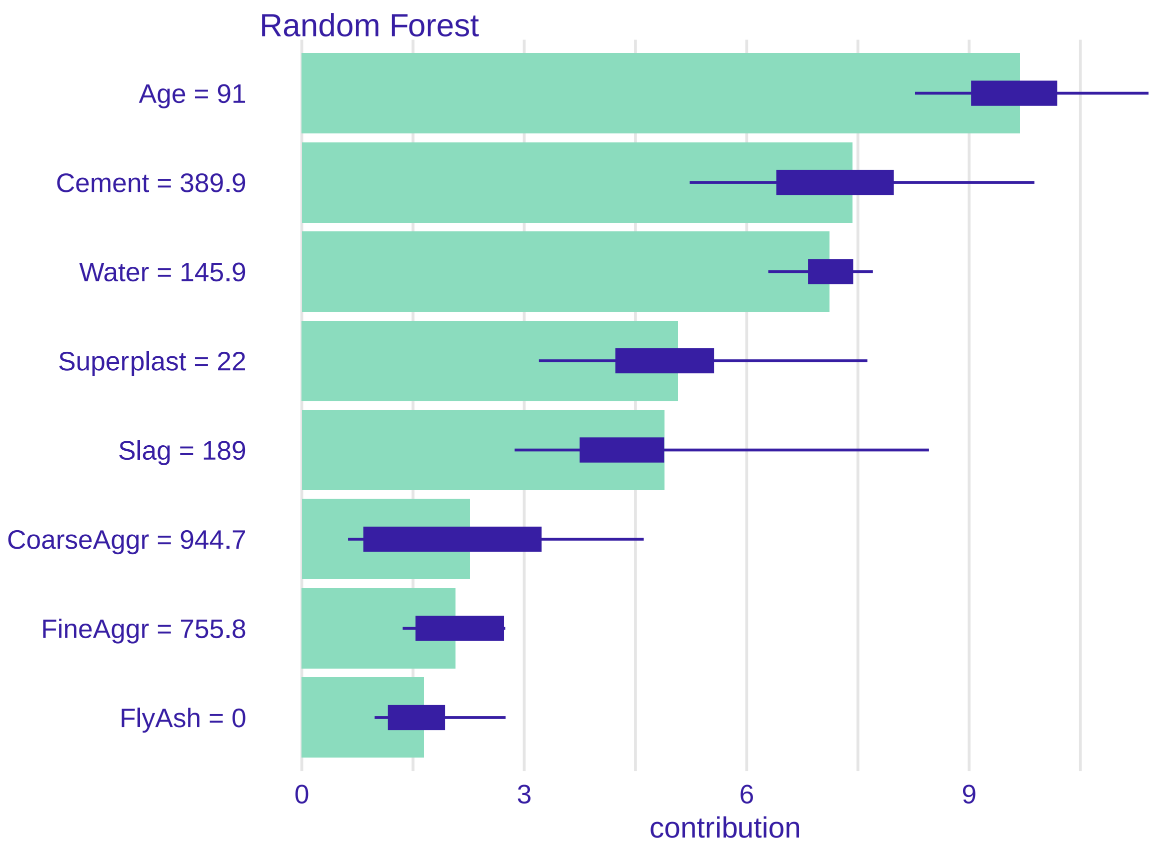


This plot shows that the features FineAggr, Cement, Superplast and Slag have the biggest impact (positively) and CoarseAggr negatively.

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

bd_rf
```

```
##                                          min        q1   median     mean
## Random Forest: Age = 91            8.2694009 9.0451195 9.629864 9.684048
## Random Forest: Cement = 389.9      5.2317888 6.4180259 7.341448 7.422482
## Random Forest: CoarseAggr = 944.7 0.6232782 0.8485973 1.978370 2.264006
## Random Forest: FineAggr = 755.8    1.3605962 1.5524153 1.924776 2.072876
## Random Forest: FlyAsh = 0          0.9824232 1.1799988 1.574334 1.646975
## Random Forest: Slag = 189          2.8703677 3.7662130 4.215196 4.887155
## Random Forest: Superplast = 22     3.1977067 4.2473083 4.890354 5.073260
## Random Forest: Water = 145.9       6.2907478 6.8462812 7.208771 7.112522
##                                           q3       max
## Random Forest: Age = 91            10.167768 11.419239
## Random Forest: Cement = 389.9       7.964974  9.879701
## Random Forest: CoarseAggr = 944.7   3.214871  4.611715
## Random Forest: FineAggr = 755.8     2.707798  2.743423
## Random Forest: FlyAsh = 0           1.912758  2.748777
## Random Forest: Slag = 189           4.868279  8.457661
## Random Forest: Superplast = 22      5.540542  7.627039
## Random Forest: Water = 145.9        7.416782  7.701468
```

```r
plot(bd_rf)
```



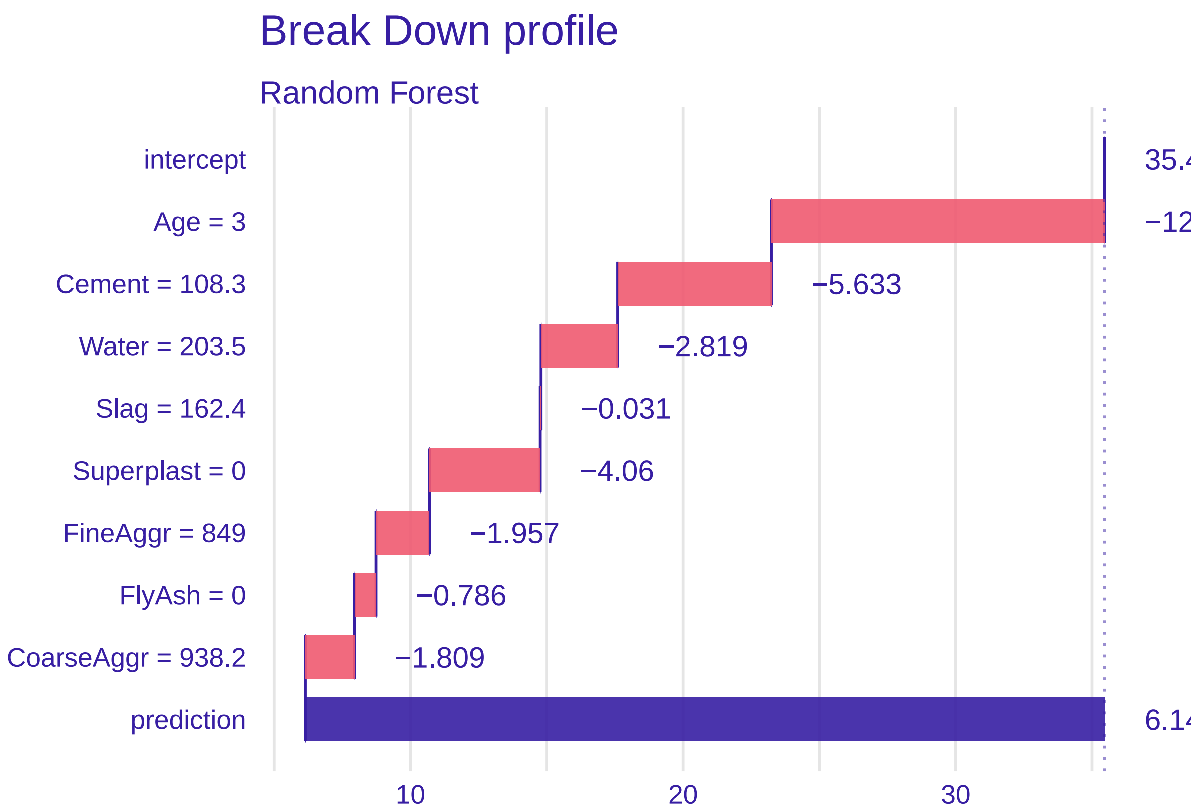
This plot shows that all features have a good contribution towards Strength.

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)
```

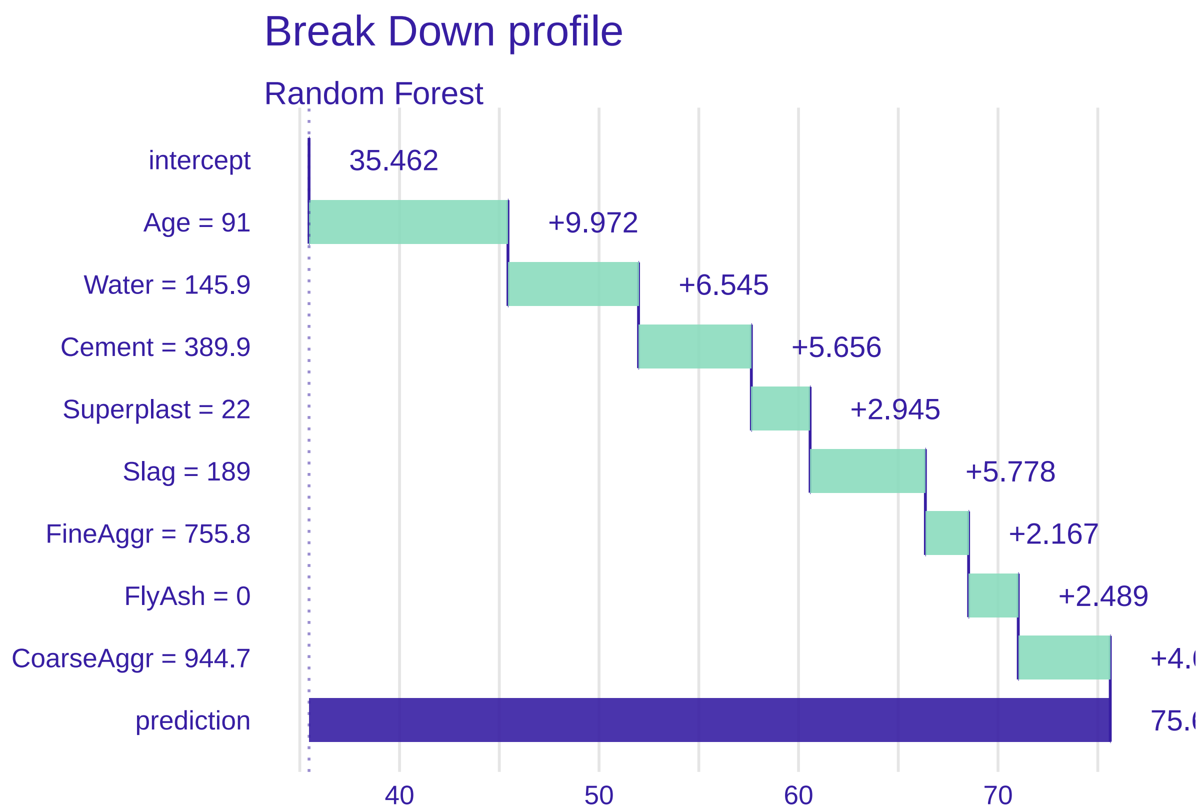


Here the plot shows that Cement and Superplast have a significant impact on the Strength. This means that we can focus on optimizing these two input variables to achieve the desired Strength.

```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)
```



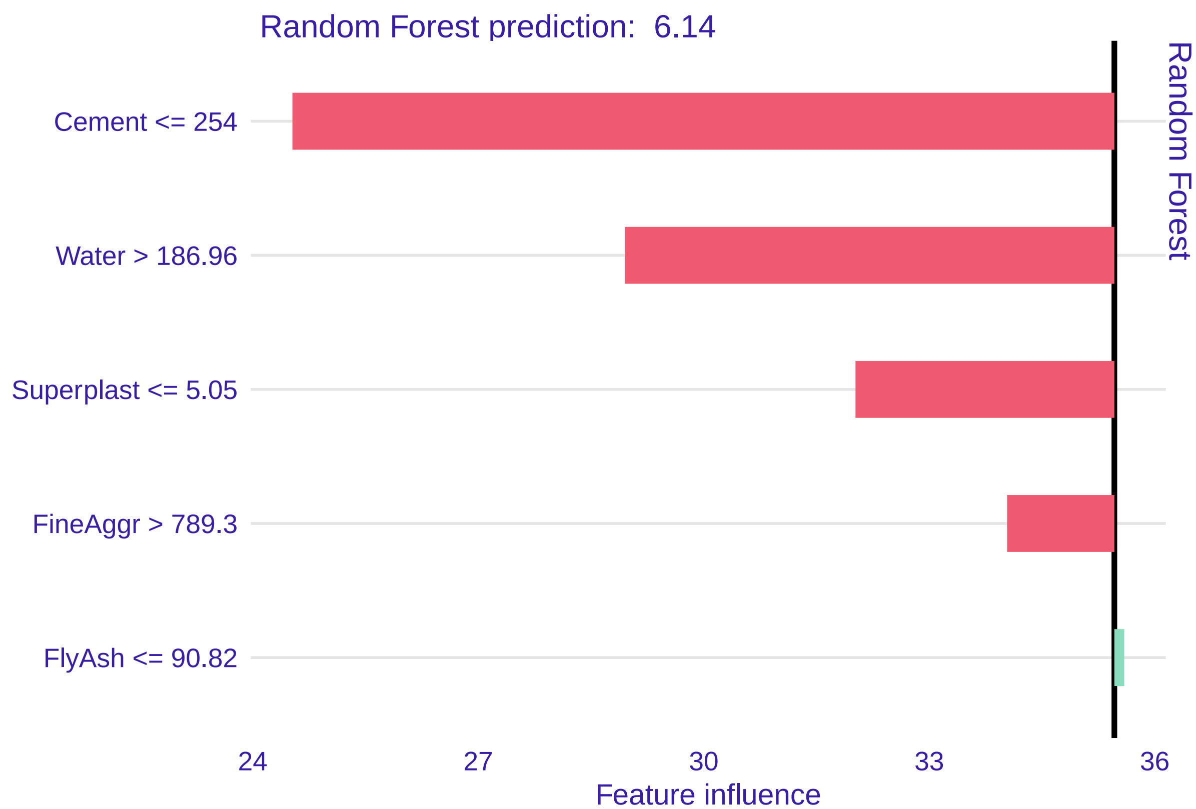
This plot shows that the predicted concrete strength increases with increasing CoarseAggr proportions up to a point of around 50%.

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)
```

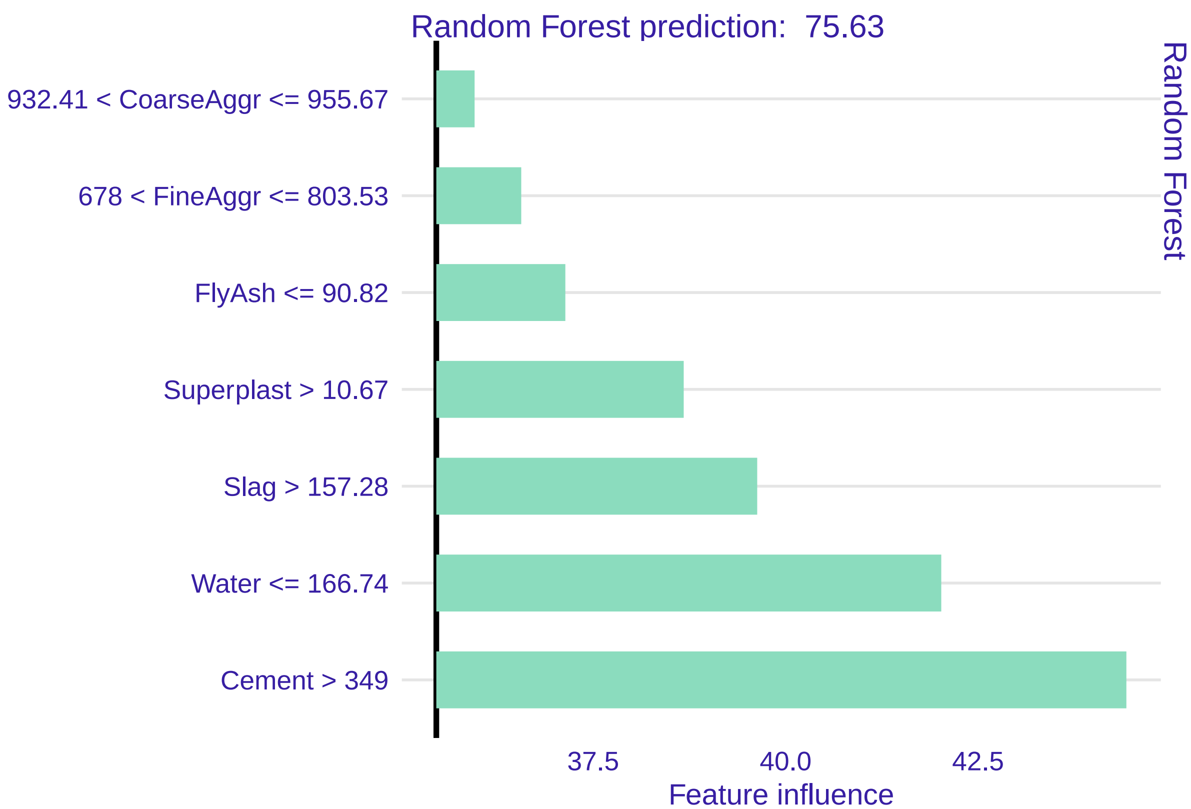


This plot that Cement, Water, Superplast and FineAggr have the biggest positive impact while Slag the biggest negative impact.

```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 big impact, but Cement and Water being the top ones.

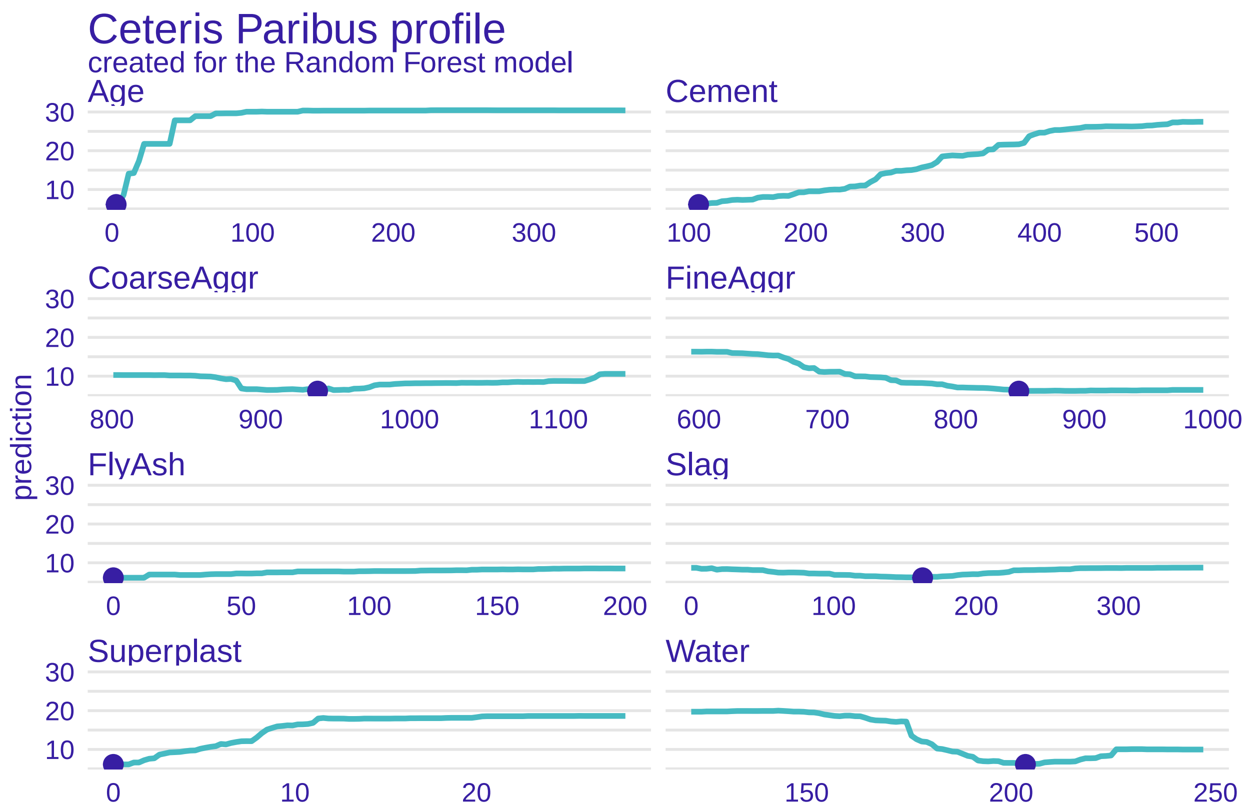
d. Do the Individual conditional expectation (ICE) plot, or ceteris paribus plot

```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)
```



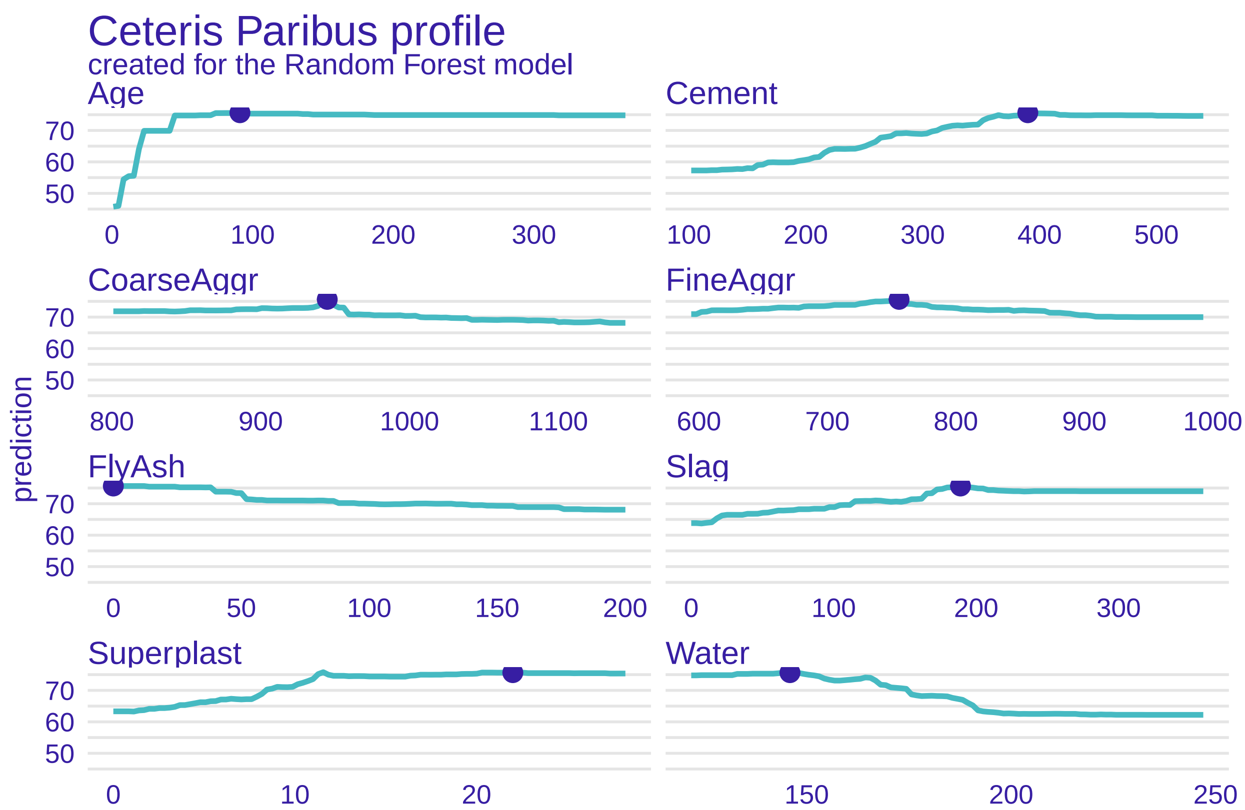
The plots show that the predicted concrete strength increases with increasing cement content, fine aggregate content, superplasticizer content, and slag content. However, the predicted concrete strength initially increases with increasing coarse aggregate content, but eventually reaches a peak and then declines.

```
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
```

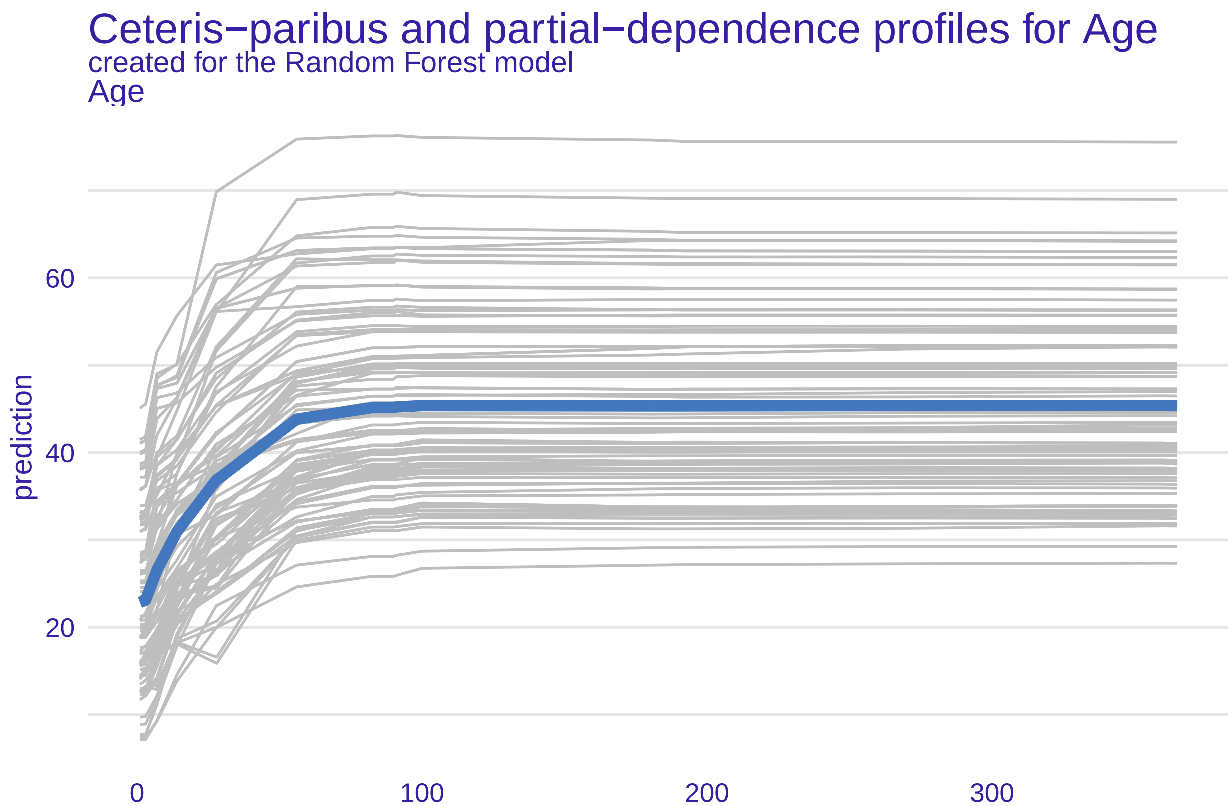
```
plot(cp_rf,facet_ncol=2)
```



The findings indicate that the predicted concrete strength increases linearly with increasing cement content and fine aggregate content. In contrast, the predicted concrete strength increases non-linearly with increasing superplasticizer content and slag content.

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.

```r
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.