Case Study 3

Ronnie Kupfer

2024-07-21

```r
library(plsdepot)
library(ggplot2)
library(visreg)
library(caret)
```

```
## Loading required package: lattice
```

```r
# Working directory
wd = "C:/Users/ronku/OneDrive/Desktop/WPI Master's Program/DS 501/Case Study 3"
setwd(wd)
library(readxl)
# Create data frame
concrete = read_excel("Concrete_Data.xls")
# Simplify column names without modifying source data file
colnames(concrete)[c(1,2,3,4,5,6,7,8,9)] = c("Cement","Slag","Ash","Water", "Superplasticizer","C_Aggreg
#View(concrete)
attach(concrete)
#detach(concrete)
#str(concrete)
head(concrete)
```

```
## # A tibble: 6 x 9
##   Cement  Slag   Ash Water Superplasticizer C_Aggregate F_Aggregate   Age
##    <dbl> <dbl> <dbl> <dbl>            <dbl>       <dbl>       <dbl> <dbl>
## 1   540      0     0   162              2.5       1040         676     28
## 2   540      0     0   162              2.5       1055         676     28
## 3   332.   142.    0   228              0          932         594    270
## 4   332.   142.    0   228              0          932         594    365
## 5   199.   132.    0   192              0          978.        826.   360
## 6   266    114     0   228              0          932         670     90
## # i 1 more variable: Compressive_Strength <dbl>
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

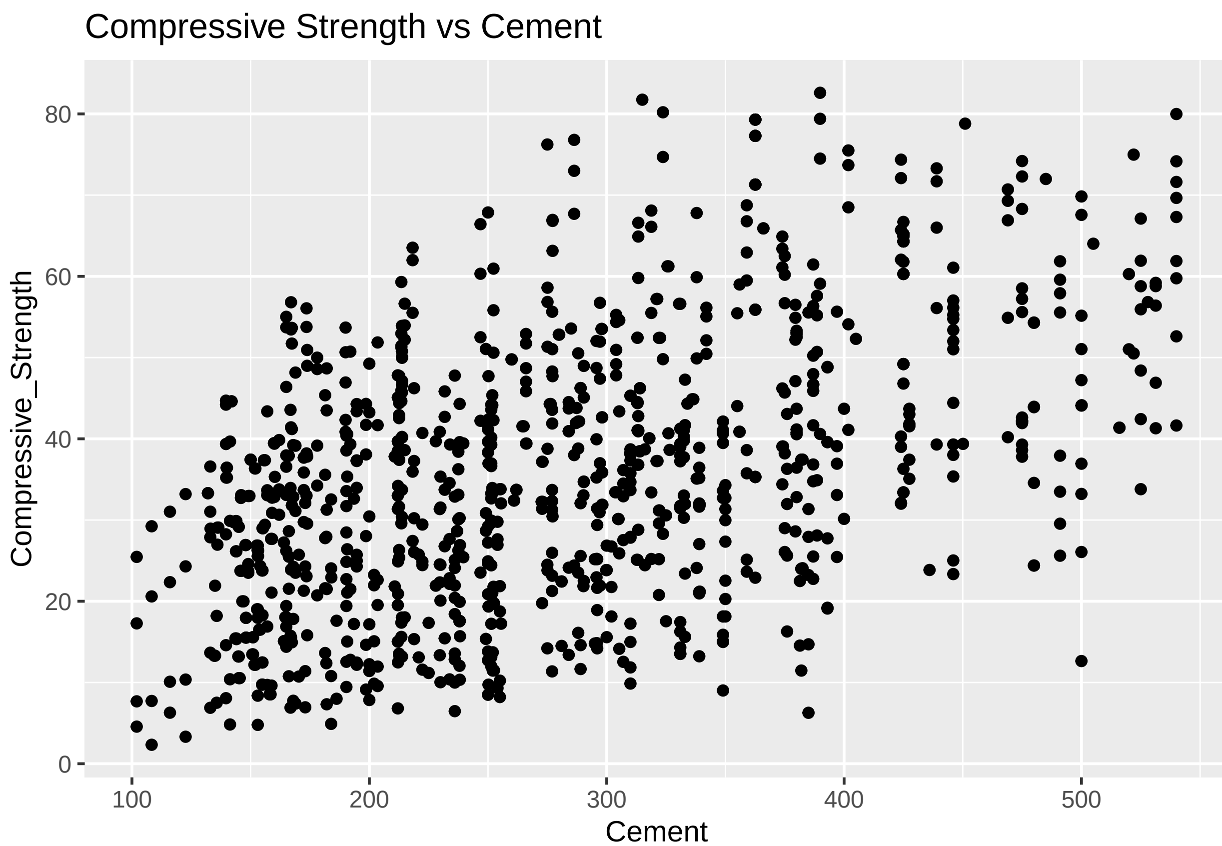
Exploratory Data Analysis -

The author of the posted data stated that there is a strong non-linear relationship between age and compressive strength. The following plots examine the relationship of each input variable to the output variable, compressive strength.

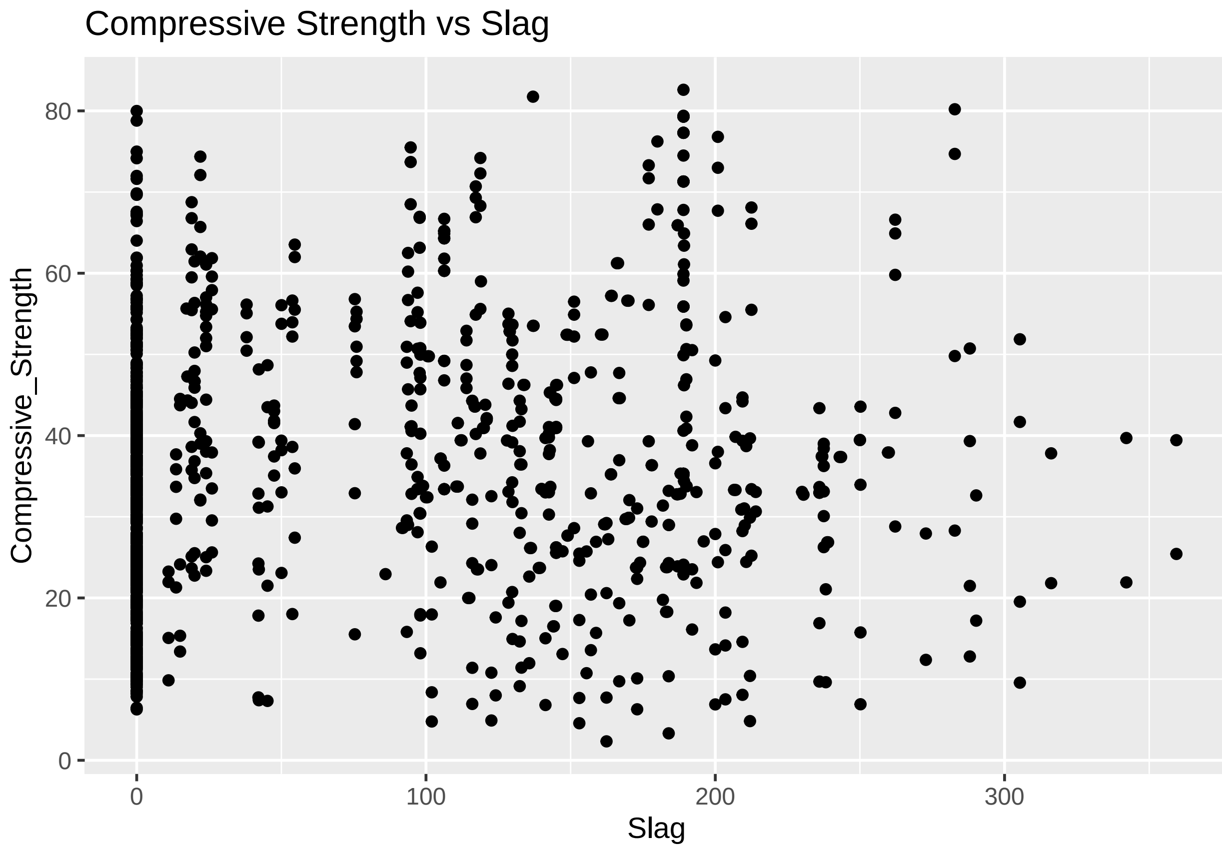
```
strengthVsCement = qplot(Cement, Compressive_Strength, main = "Compressive Strength vs Cement")
```

```
## Warning: ‘qplot()‘ was deprecated in ggplot2 3.4.0.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

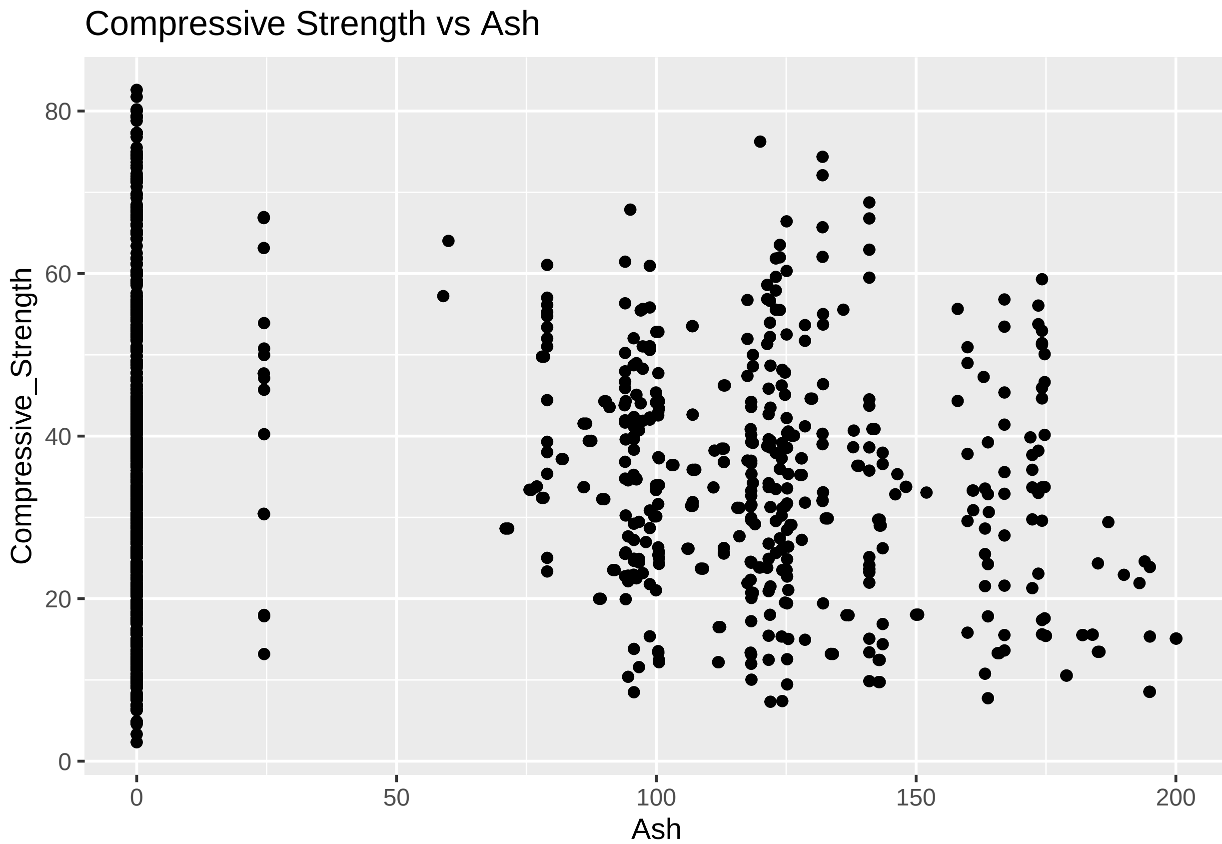
```
strengthVsCement
```



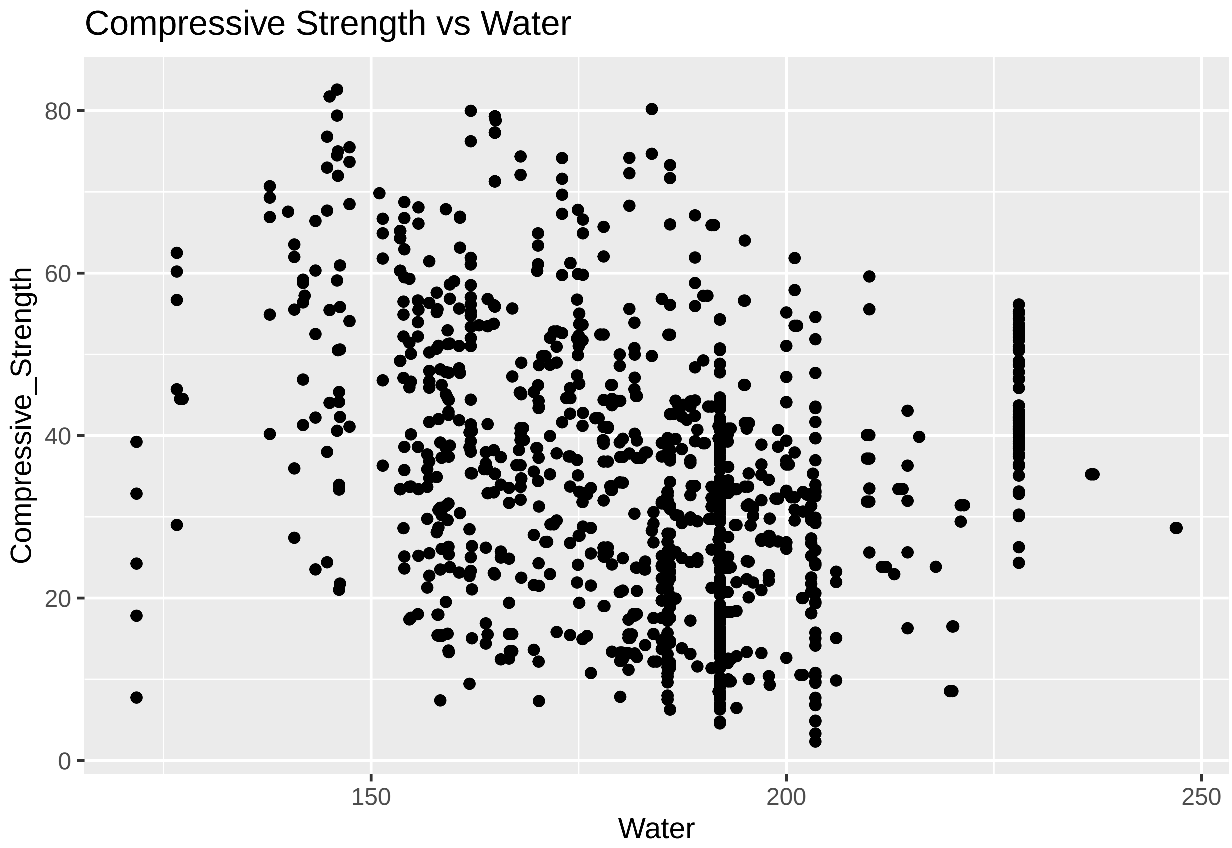

```
strengthVsSlag = qplot(Slag, Compressive_Strength, main = "Compressive Strength vs Slag")
strengthVsSlag
```



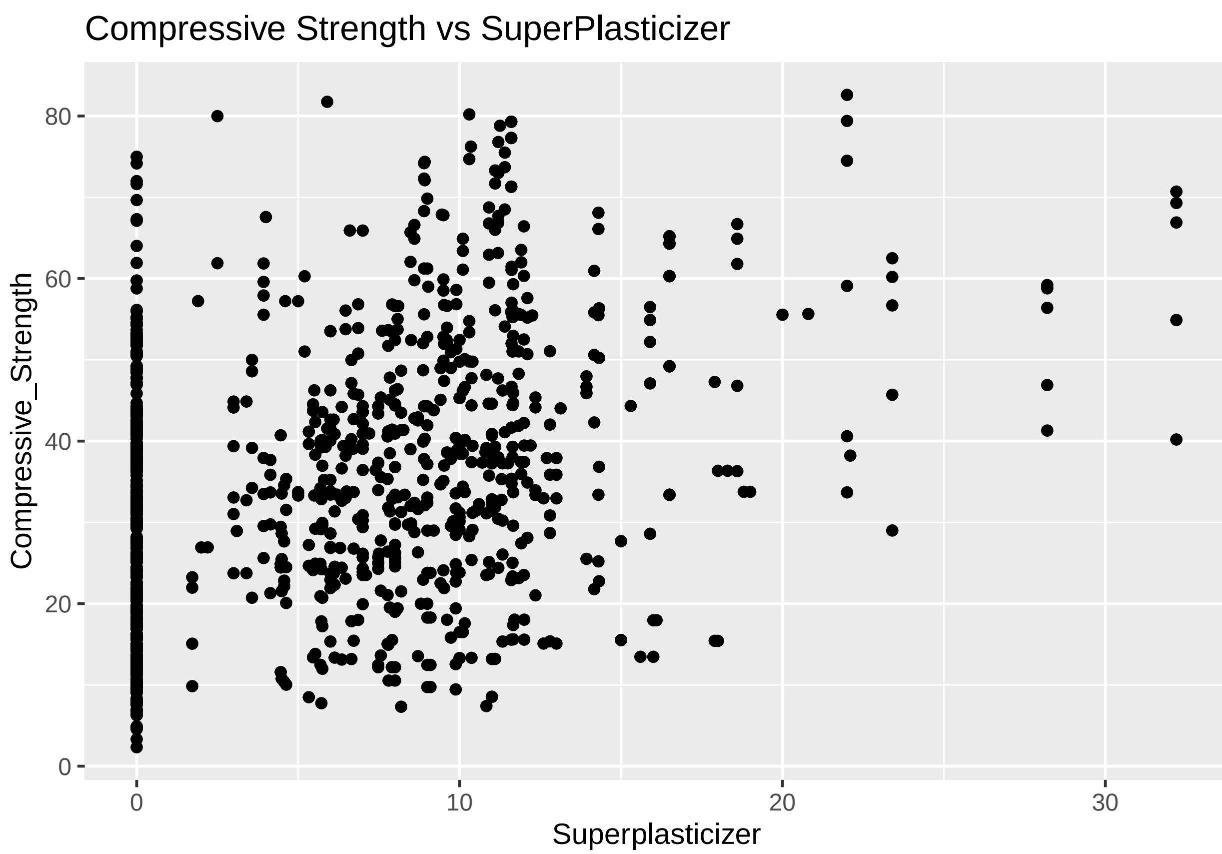

```
strengthVsAsh = qplot(Ash, Compressive_Strength, main = "Compressive Strength vs Ash")
strengthVsAsh
```



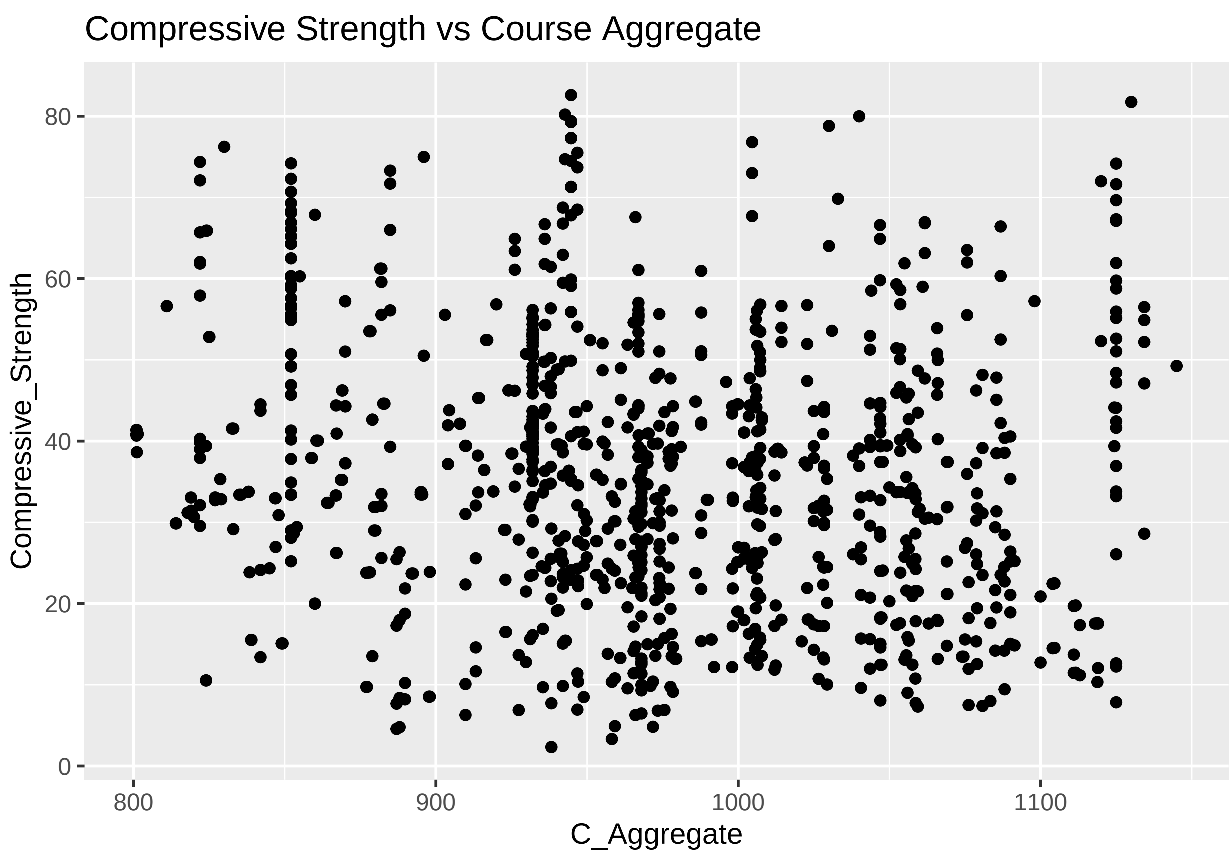

```r
strengthVsWater = qplot(Water, Compressive_Strength, main = "Compressive Strength vs Water")
strengthVsWater
```



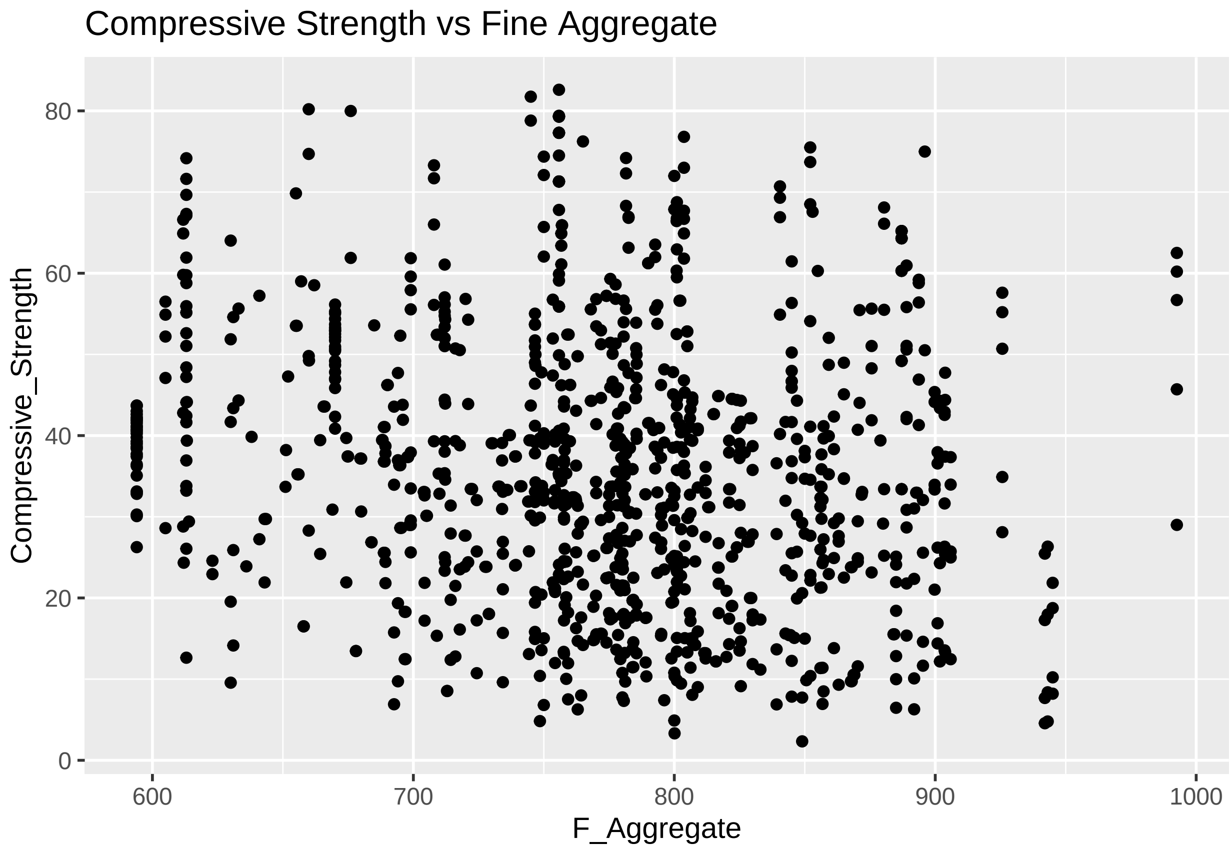

```
strengthVsPlasticizer = qplot(Superplasticizer, Compressive_Strength, main = "Compressive Strength vs Su
strengthVsPlasticizer
```



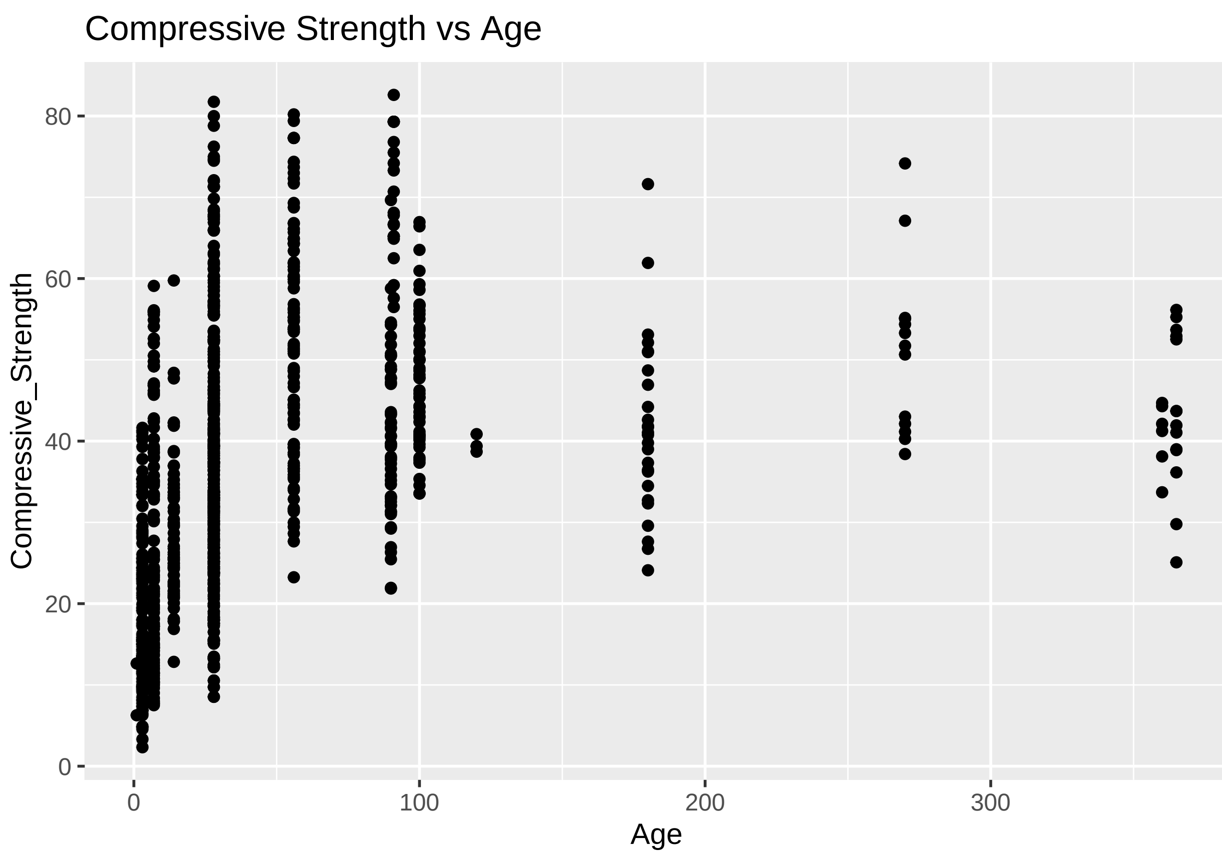

```
strengthVsC_Aggregate = qplot(C_Aggregate, Compressive_Strength, main = "Compressive Strength vs Course
strengthVsC_Aggregate
```




```
strengthVsF_Aggregate = qplot(F_Aggregate, Compressive_Strength, main = "Compressive Strength vs Fine Ag
strengthVsF_Aggregate
```




```r
strengthVsAge = qplot(Age, Compressive_Strength, main = "Compressive Strength vs Age")
strengthVsAge
```



#{r scatter plot Compressive_Strength to Age, echo=T, eval=T, error=TRUE} #plot(concrete) # Reviewing scatter plots each of the input variables to the output variable, Compressive Strength, once can draw three conclusions: 1) It appears that as the amount of cement added to the mixture increases the compressive strength of the concrete increases in a fairly linear manner. 2) The data provider's statement that there is a highly nonlinear function of concrete compressive strength to age and ingredients seems to have merit. 3) Most of the input variables seem to have optimal ranges.

```
corStrengthToCement = cor(Compressive_Strength, Cement)
corStrengthToCement
```

```
## [1] 0.4978327
```

```
corStrengthToSlag = cor(Compressive_Strength, Slag)
corStrengthToSlag
```

```
## [1] 0.1348244
```

```
corStrengthToAsh = cor(Compressive_Strength, Ash)
corStrengthToAsh
```

```
## [1] -0.1057533
```

```
corStrengthToWater = cor(Compressive_Strength, Water)
corStrengthToWater
```

```
## [1] -0.2896135
```

```r
corStrengthToPlasticizer = cor(Compressive_Strength, Superplasticizer)
corStrengthToPlasticizer
```

```
## [1] 0.3661023
```

```r
corStrengthToC_Aggregate = cor(Compressive_Strength, C_Aggregate)
corStrengthToC_Aggregate
```

```
## [1] -0.1649278
```

```r
corStrengthToF_Aggregate = cor(Compressive_Strength, F_Aggregate)
corStrengthToF_Aggregate
```

```
## [1] -0.167249
```

```r
corStrengthToAge = cor(Compressive_Strength, Age)
corStrengthToAge
```

```
## [1] 0.328877
```

The correlation between the strength of the concrete to the amount of cement is nearly .5. This is the strongest correlation of input variable to the output variable and reflects the trend in the graph. Several of the input variables have a slightly negative correlation. If the relationship is non-linear, the negative correlations may not indicate anything about dependence of the variables.

Split the data as Training and Test sets

```r
set.seed(2024)
concreteDF = data.frame(concrete)
splitConcrete = caret::createDataPartition(concreteDF[,1], p = 0.8, list=F, times=1)
#splitConcrete
trainConcrete = concreteDF[splitConcrete,]
head(trainConcrete)
```

```
##   Cement  Slag Ash Water Superplasticizer C_Aggregate F_Aggregate Age
## 1  540.0   0.0   0   162              2.5        1040         676  28
## 2  540.0   0.0   0   162              2.5        1055         676  28
## 3  332.5 142.5   0   228              0.0         932         594 270
## 4  332.5 142.5   0   228              0.0         932         594 365
## 6  266.0 114.0   0   228              0.0         932         670  90
## 7  380.0  95.0   0   228              0.0         932         594 365
##   Compressive_Strength
## 1             79.98611
## 2             61.88737
## 3             40.26954
## 4             41.05278
## 6             47.02985
## 7             43.69830
```

```
testConcrete = concreteDF[-splitConcrete,]
#testConcrete
```

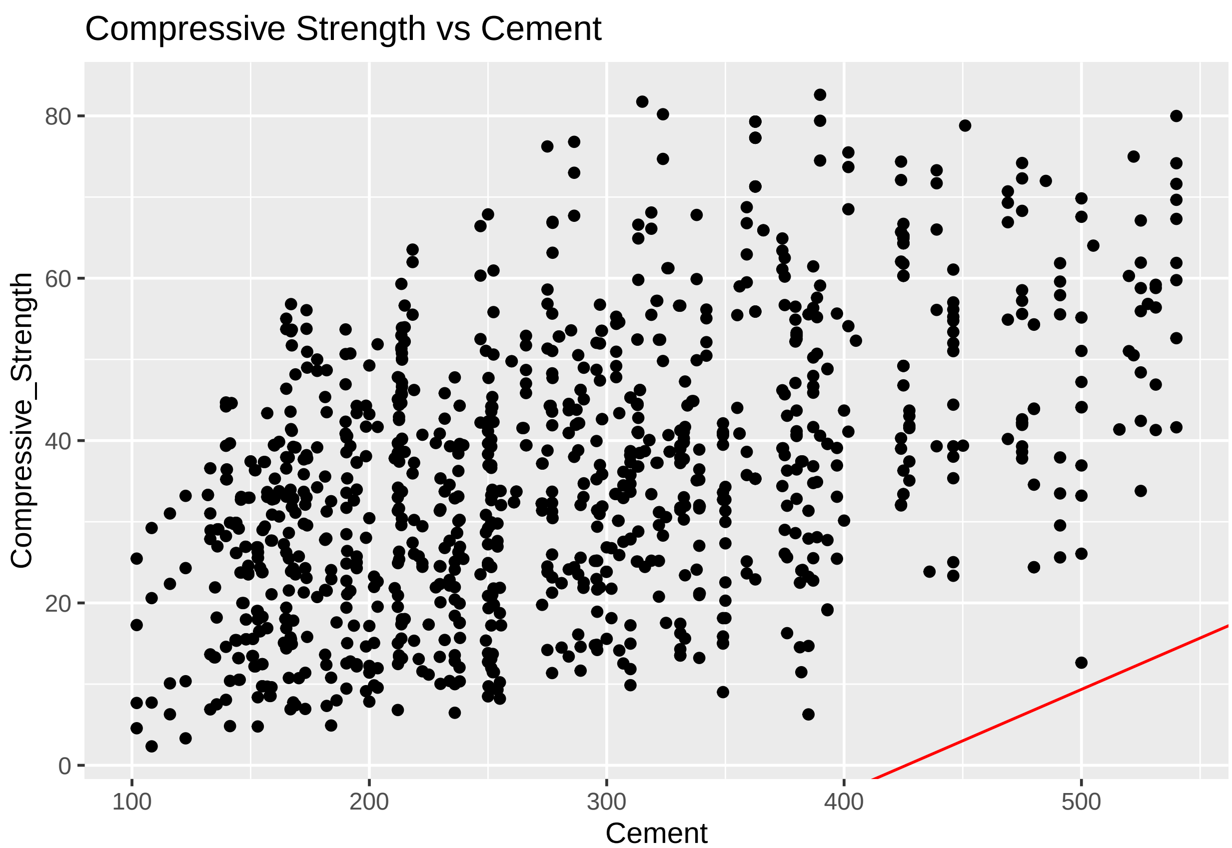
Generating a multiple linear regression model

```
mlr = lm(Compressive_Strength ~ Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggrega
#fitted(mlr)
#resid(mlr)
#mlr
summary(mlr)
```

```
##
## Call:
## lm(formula = Compressive_Strength ~ Cement + Slag + Ash + Water +
##     Superplasticizer + C_Aggregate + F_Aggregate + Age, data = trainConcrete)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.851  -6.312   0.474   6.793  34.164
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -53.950023  29.053197  -1.857 0.063680 .
## Cement             0.126592   0.009162  13.817  < 2e-16 ***
## Slag               0.109257   0.010969   9.960  < 2e-16 ***
## Ash                0.093200   0.013580   6.863 1.33e-11 ***
## Water             -0.110109   0.044433  -2.478 0.013410 *
## Superplasticizer   0.384435   0.105033   3.660 0.000268 ***
## C_Aggregate        0.029932   0.010271   2.914 0.003663 **
## F_Aggregate        0.031474   0.011763   2.676 0.007606 **
## Age                0.111409   0.005931  18.785  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.34 on 817 degrees of freedom
## Multiple R-squared:  0.6227, Adjusted R-squared:  0.619
## F-statistic: 168.6 on 8 and 817 DF,  p-value: < 2.2e-16
```

```
#plot(mlr)
```

```
p1 = strengthVsCement + geom_abline(intercept = mlr[1]$coefficients[1], slope = mlr[1]$coefficients[2],
p1
```

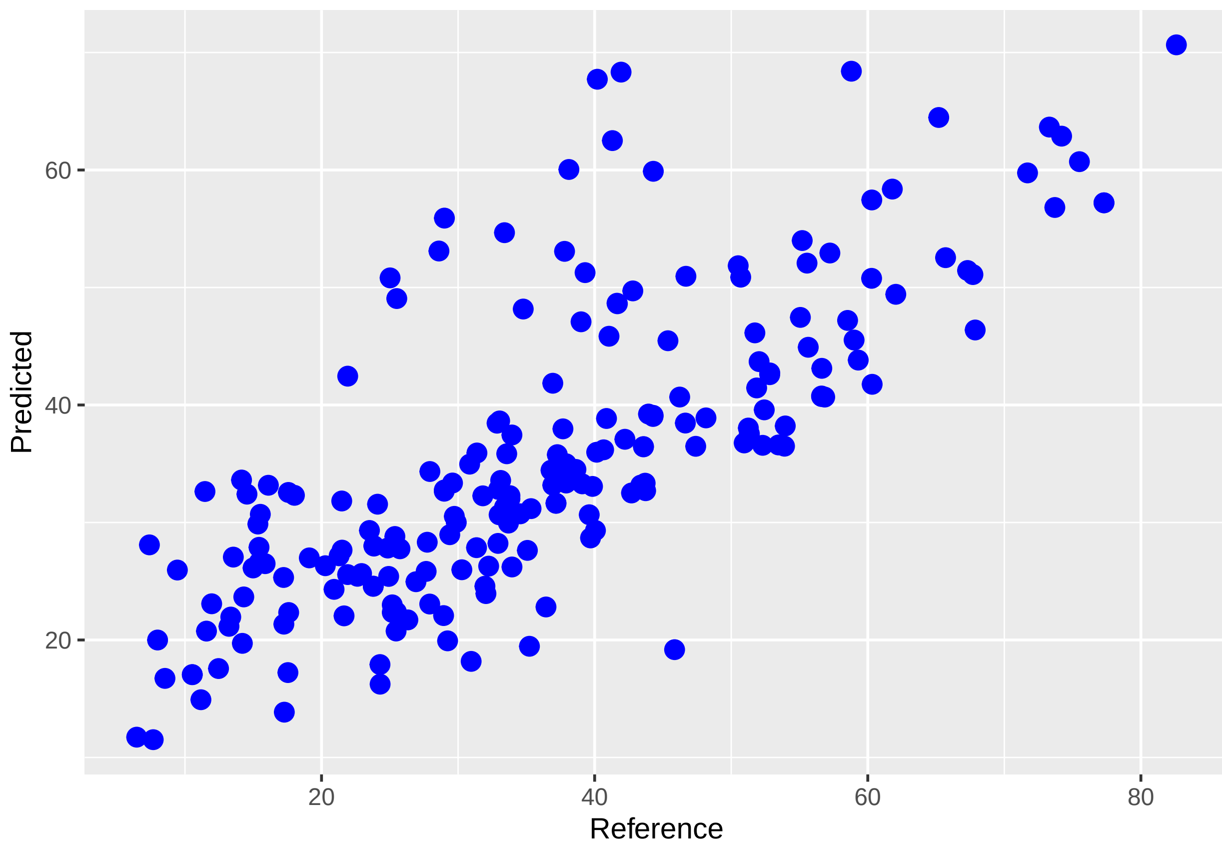



```r
predict(mlr, newdata=testConcrete)
```

```
##        5        9       46       53       57       59       62       66 
## 59.88984 19.17717 27.62453 33.57783 68.33219 36.76999 46.13804 47.45548 
##       68       75       76       77       80       81       86       95 
## 25.97773 55.90629 53.07744 67.72511 62.50311 54.66653 53.10547 49.71150 
##      120      125      127      134      148      149      152      153 
## 58.37663 50.87755 57.45174 51.10031 53.99699 68.40777 56.81088 57.20757 
##      156      159      168      173      175      181      182      185 
## 57.20757 59.75504 62.88139 64.47048 60.71018 63.65434 70.65302 20.75366 
##      193      194      199      203      206      215      221      228 
## 25.83579 30.73776 34.47909 31.17964 24.30879 25.95297 22.38665 38.45155 
##      234      240      243      251      255      261      270      273 
## 36.49260 21.95076 27.85541 25.41017 27.05262 28.80774 26.60648 32.51113 
##      275      291      295      296      299      309      314      317 
## 25.31937 27.64829 28.09407 29.31956 38.90070 44.91364 43.69221 37.44771 
##      319      322      331      333      337      338      342      345 
## 45.46912 34.96442 37.08158 41.76073 37.55365 40.67309 36.48316 32.55770 
##      348      354      365      368      369      370      373      374 
## 38.46235 43.82052 32.31282 38.21748 43.11945 29.86856 35.77321 40.67519 
##      381      386      389      395      401      402      404      411 
## 52.93410 50.77951 52.07427 36.56795 46.38364 47.19946 45.52763 27.16347 
##      413      423      425      430      437      441      452      456 
## 30.69545 32.26849 30.65629 33.35980 29.94869 27.82555 36.60011 38.03895 
##      463      467      471      477      481      488      490      491
```

```
## 37.97010 35.84696 35.97861 50.81969 51.26532 50.95401 49.06068 48.16879
##      494      508      511      512      520      537      538      547
## 48.61443 49.42031 47.08073 52.53975 32.70419 30.65878 39.23035 24.57594
##      551      556      561      568      580      585      586      587
## 28.31920 34.33924 27.76032 23.08091 25.42049 42.45393 38.64528 32.41232
##      589      596      598      599      602      606      610      626
## 19.99638 22.33596 35.91498 21.16242 23.94763 11.72711 31.55653 28.22016
##      630      640      644      648      654      655      656      660
## 14.91512 33.34386 26.31996 33.18096 20.76243 16.23765 26.21408 19.92614
##      670      678      681      682      687      693      707      713
## 33.16740 11.51553 13.85510 17.90894 33.61541 28.68604 21.71106 31.82954
##      720      721      723      727      730      732      735      740
## 33.12141 41.45193 21.34667 23.66531 33.35785 26.49416 36.18670 22.98939
##      741      744      750      754      771      775      780      781
## 18.17822 33.17696 41.85049 51.44683 26.13035 32.63737 22.34434 17.22476
##      782      783      784      792      800      804      812      815
## 19.71360 22.05318 28.96051 45.84966 48.66162 26.98692 23.05860 60.04623
##      826      828      830      848      849      854      859      869
## 33.28707 51.85432 31.64093 42.58063 31.29911 29.32003 39.58656 39.02344
##      883      885      893      902      903      905      913      916
## 22.80964 28.02528 26.28366 30.00342 40.76592 24.56357 36.43688 32.66100
##      918      919      923      924      930      935      938      942
## 24.95739 17.05611 32.02042 33.06891 25.56601 25.65504 34.44796 22.07644
##      947      948      951      954      958      966      967      968
## 32.80925 27.99132 34.51552 16.73157 30.52072 40.71782 17.56750 24.58391
##      976      979      982      986      987      993      994     1005
## 36.46145 32.76523 17.04413 32.28224 31.63149 27.88523 31.30673 42.72968
##     1014     1020     1025     1026
## 38.85186 19.46307 34.99607 39.12007
```

```
predCompressive_Strength = data.frame(predict(mlr, newdata=testConcrete))
names(predCompressive_Strength)[1] = 'Predicted'
predCompressive_Strength$Reference = testConcrete[,c('Compressive_Strength')]
plotpredmlr = qplot(Reference, Predicted, data=predCompressive_Strength) + geom_point(colour = "blue", s
plotpredmlr
```



Model evaluation - $RMSEP$ and R^2 * Calculating predicted residual sum of squares (PRESS)

$$PRESS = \sum_{i=1}^{n}(y_i^{ref} - y_i^{pred})^2$$

```
PRESS = sum((predCompressive_Strength$Reference - predCompressive_Strength$Predicted)^2)
PRESS
```

```
## [1] 23498.91
```

- Root mean squared error of prediction (RMSEP)

$$RMSEP = \sqrt{\frac{1}{n_T}\sum_{1}^{n_T}(y_i^{ref} - y_i^{pred})^2}$$

```
RMSEP = sqrt(PRESS/ nrow(predCompressive_Strength))
RMSEP
```

```
## [1] 10.7327
```

- Total sum of squares (SST)

$$SST = \sum_{i=1}^{n}(y_i^{ref} - y_i^{mean})^2$$

```r
SST = sum((predCompressive_Strength$Reference - mean(predCompressive_Strength$Reference))^2)
SST
```

```
## [1] 55712.38
```

- Calculating R^2

$$R^2 = 1 - \frac{PRESS}{SST}$$

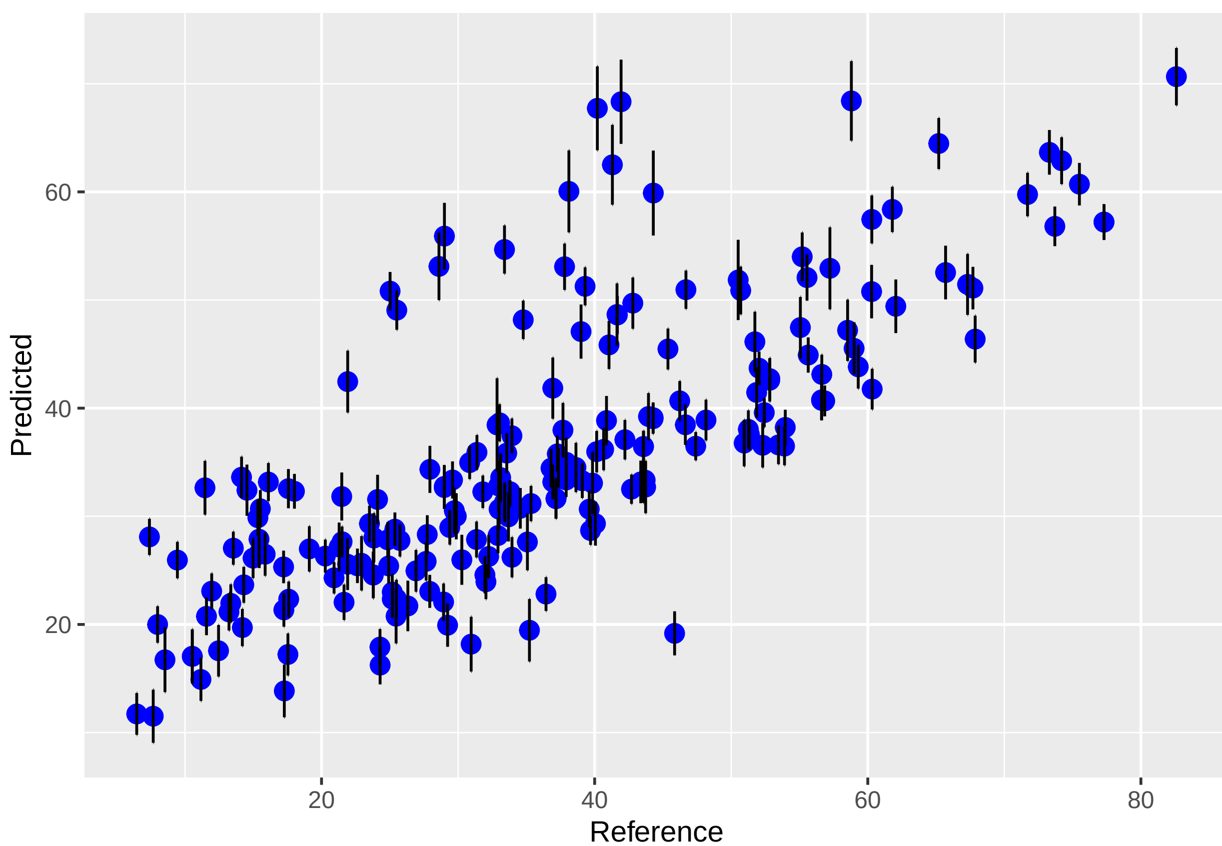
```r
R2 = 1 - (PRESS/SST)
R2
```

```
## [1] 0.5782102
```

Predicted versus Reference

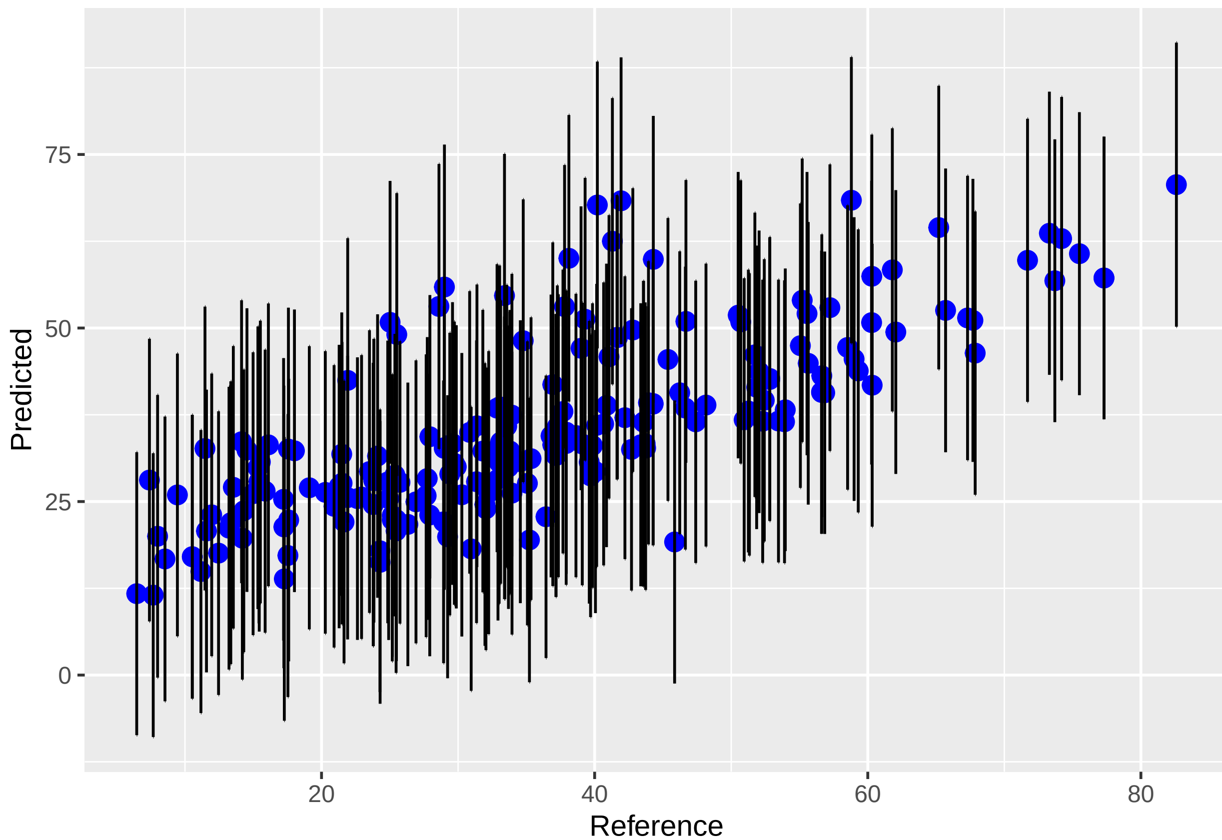
- Confidence Intervals (Narrow)

```r
predCompressive_Strength$lower = predict(mlr, newdata=testConcrete, interval = "confidence")[,2]
predCompressive_Strength$upper = predict(mlr, newdata=testConcrete, interval = "confidence")[,3]
#predCompressive_Strength
qplot(Reference, Predicted, data=predCompressive_Strength) + geom_point(colour = "blue", size = 3) +
  geom_errorbar(aes(ymin = lower,ymax = upper))
```



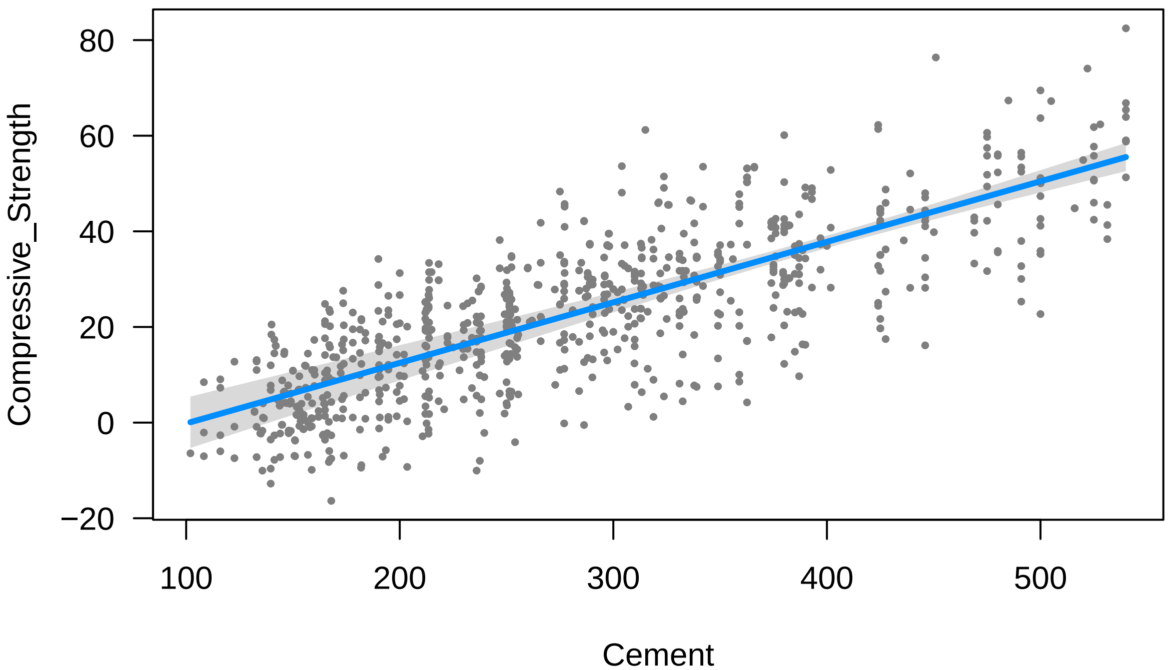
* Prediction Intervals (Tolerance/Wide)

```r
predCompressive_Strength$lower = predict(mlr, newdata=testConcrete, interval = "prediction")[,2]
predCompressive_Strength$upper = predict(mlr, newdata=testConcrete, interval = "prediction")[,3]
#predCompressive_Strength
qplot(Reference, Predicted, data=predCompressive_Strength) + geom_point(colour = "blue", size = 3) +
  geom_errorbar(aes(ymin = lower,ymax = upper))
```

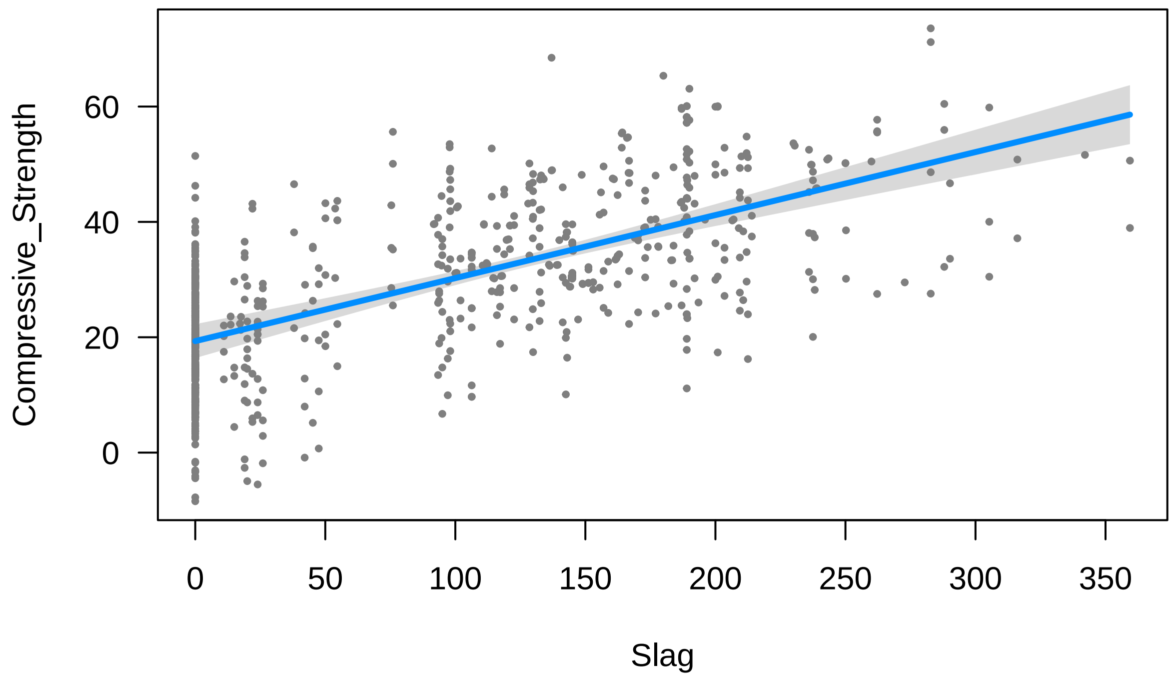


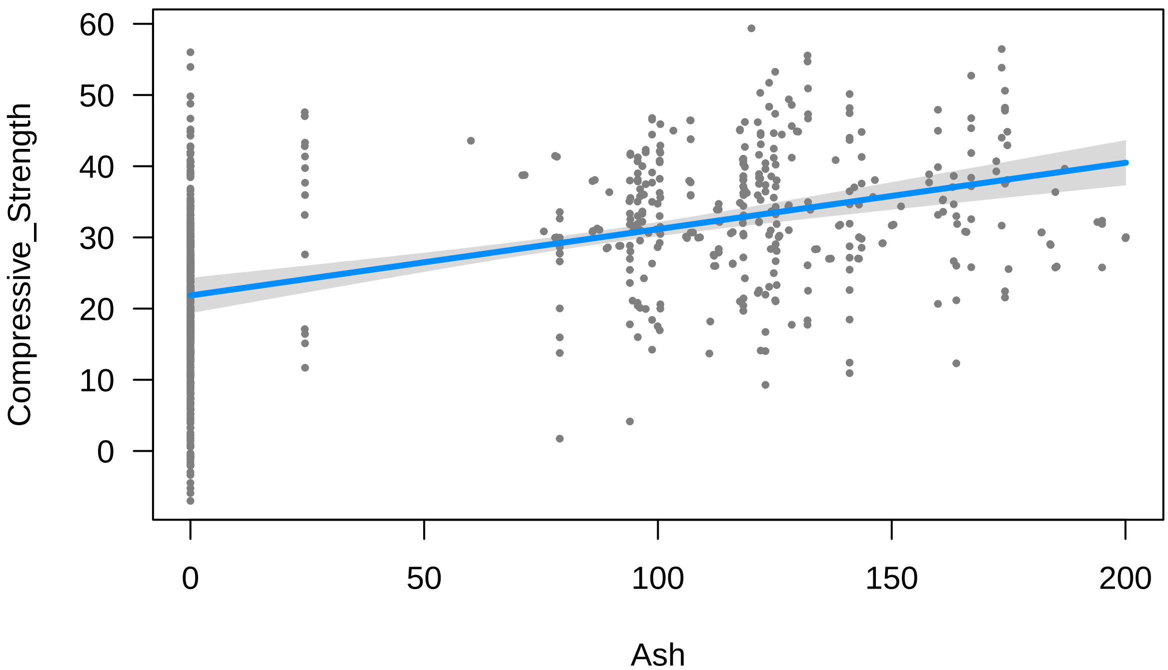
Visualizing using `visreg` package

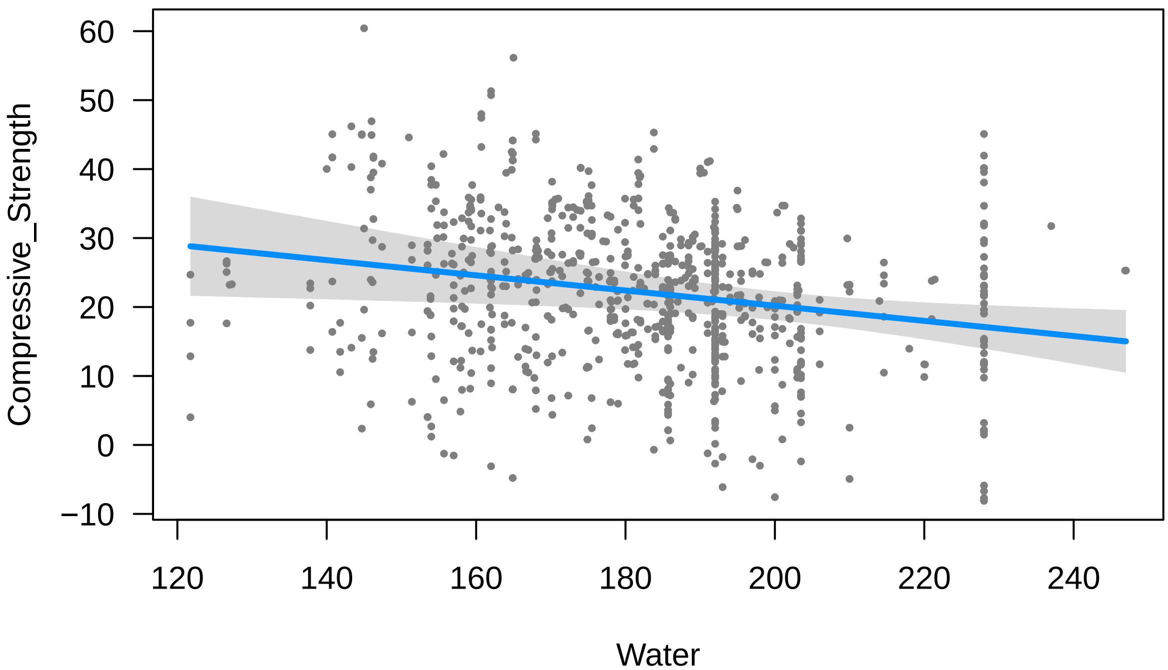
```r
visreg::visreg(mlr)
```

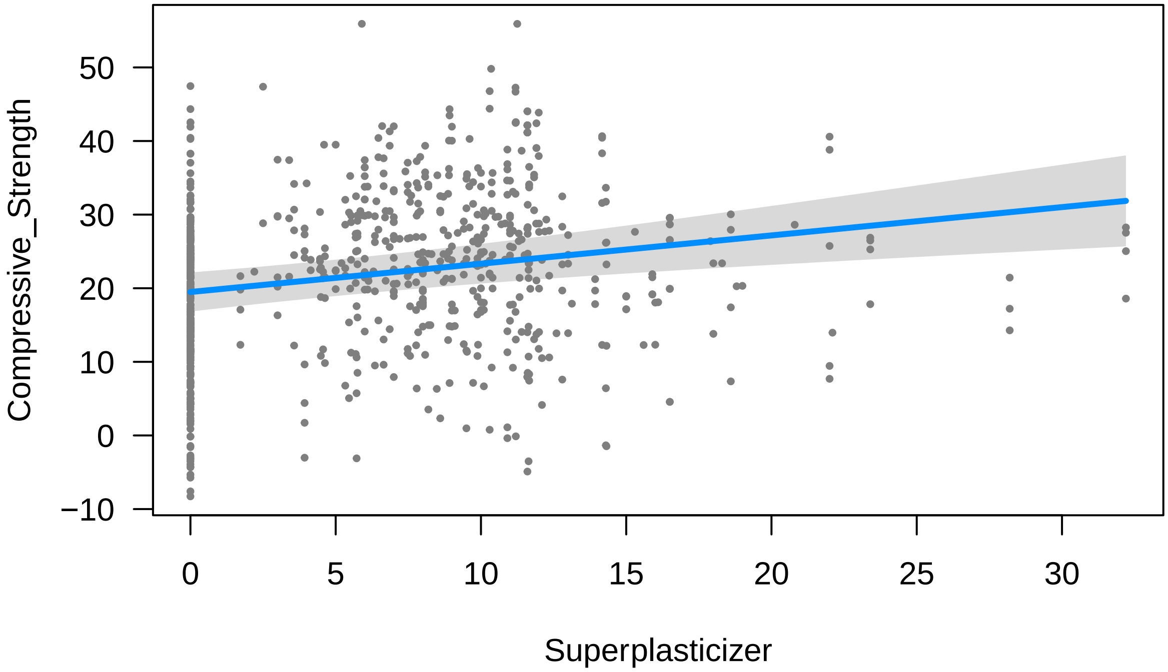


Compressive_Strength
80
60
40
20
0
−20
100
200
300
400
500
Cement

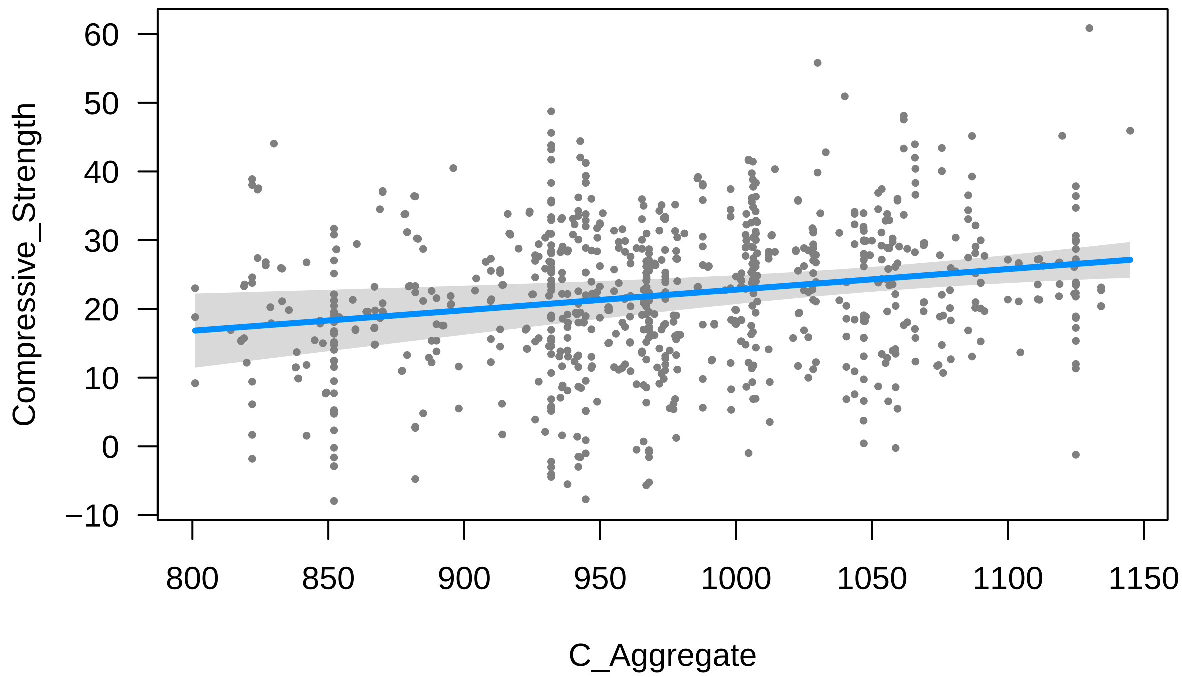

Compressive_Strength
60
40
20
0
0
50
100
150
200
250
300
350
Slag

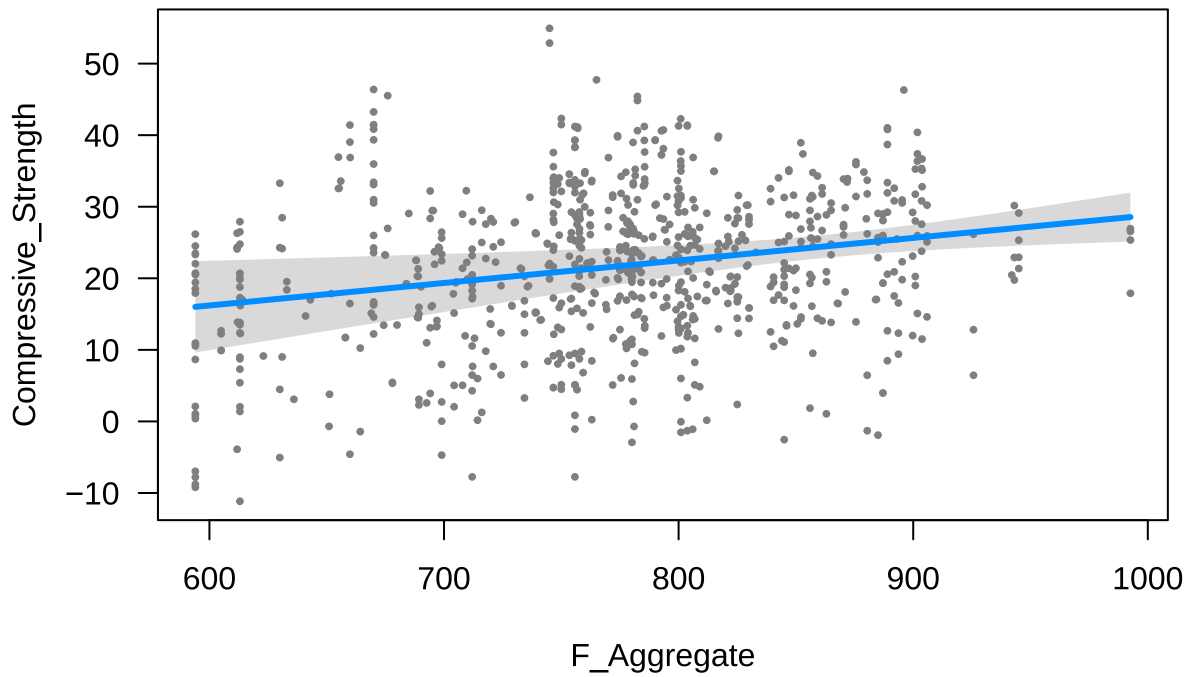

60
50
40
30
20
10
0
Compressive_Strength
0
50
100
150
200
Ash


Compressive_Strength
60
50
40
30
20
10
0
−10
120
140
160
180
200
220
240
Water

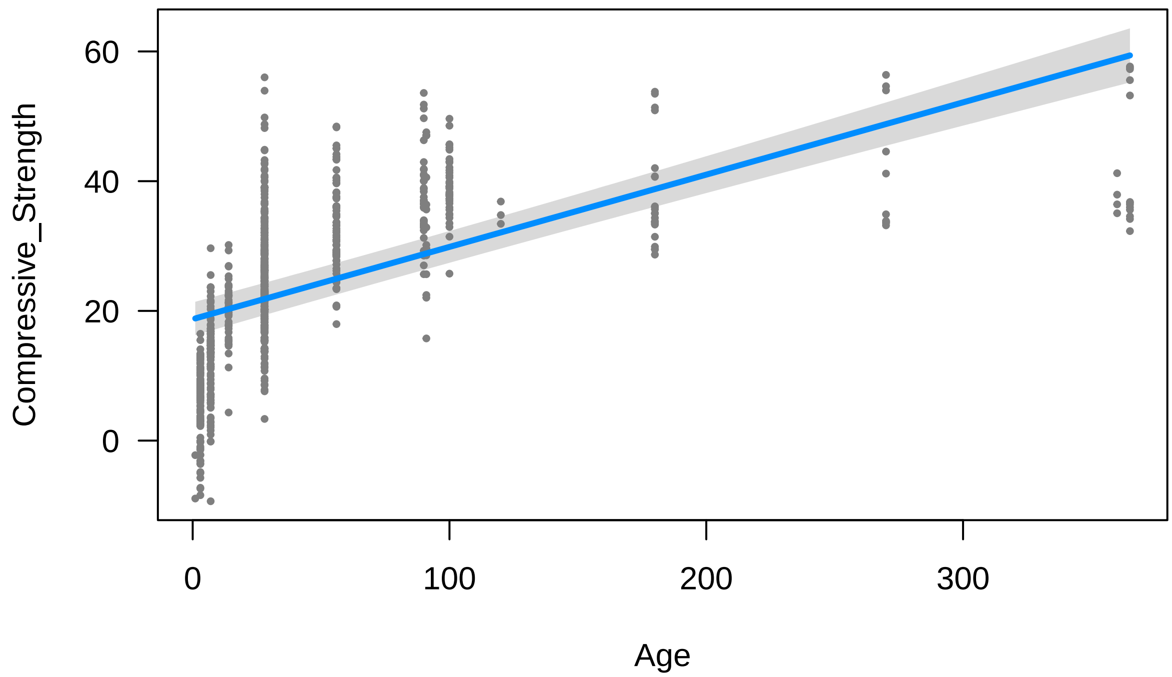


Compressive_Strength
50
40
30
20
10
0
−10
0
5
10
15
20
25
30
Superplasticizer


Compressive_Strength
60
50
40
30
20
10
0
−10
800
850
900
950
1000
1050
1100
1150
C_Aggregate



Compressive_Strength
50
40
30
20
10
0
−10
600
700
800
900
1000
F_Aggregate



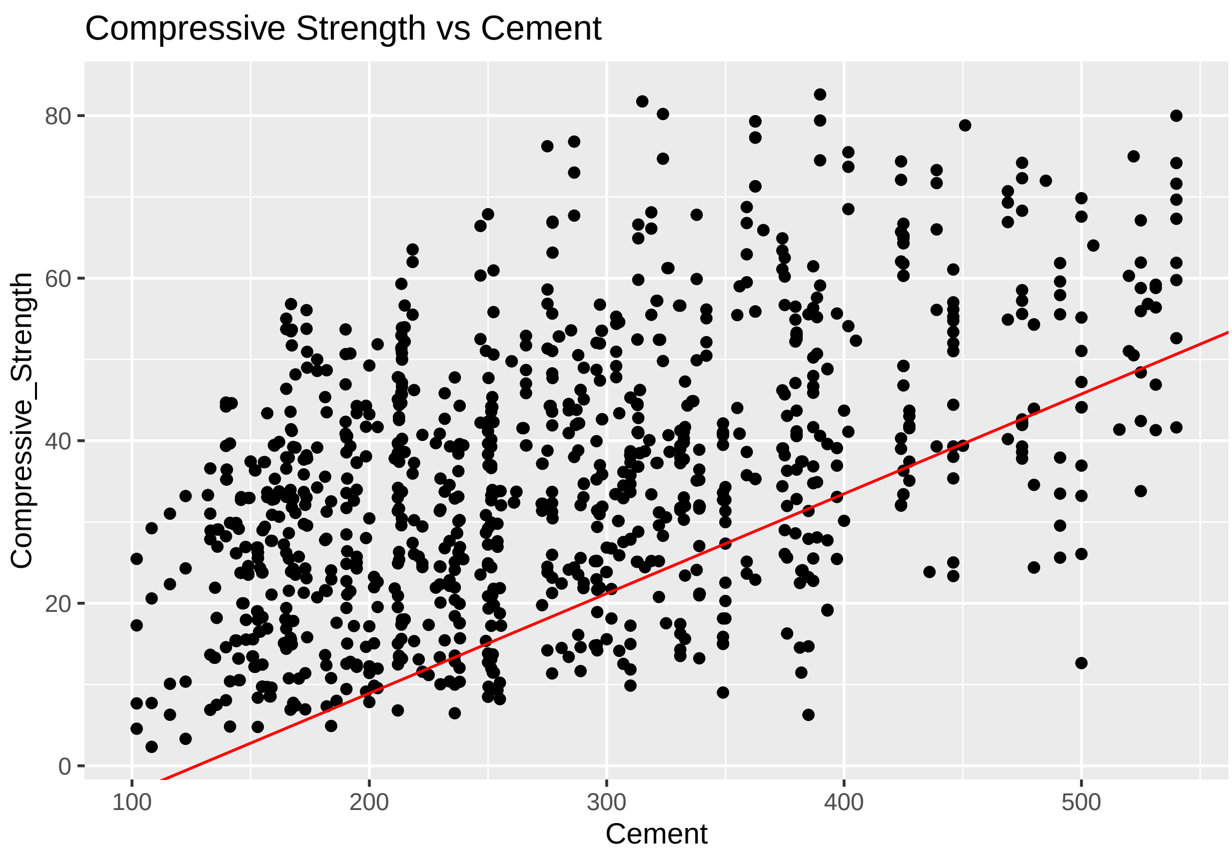
Reviewing the statistics above would indicate that Cement, Slag, Ash, and Age are the most significant input variables in the linear model. The following code creates a new model with only these input variables.

```
mlr2 = lm(Compressive_Strength ~ Cement + Slag + Ash + Age, data=trainConcrete)
#fitted(mlr2)
#resid(mlr2)
#mlr2
summary(mlr2)
```

```
##
## Call:
## lm(formula = Compressive_Strength ~ Cement + Slag + Ash + Age,
##     data = trainConcrete)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.949  -7.630  -0.007   7.737  43.322
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept) -15.619180   1.930572   -8.09 2.13e-15 ***
## Cement        0.122720   0.004744   25.87  < 2e-16 ***
## Slag          0.094326   0.005549   17.00  < 2e-16 ***
## Ash           0.107360   0.008020   13.39  < 2e-16 ***
## Age           0.088191   0.006369   13.85  < 2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.62 on 821 degrees of freedom
## Multiple R-squared:  0.5212, Adjusted R-squared:  0.5189
## F-statistic: 223.4 on 4 and 821 DF,  p-value: < 2.2e-16
```

```
#plot(mlr2)
p2 = strengthVsCement + geom_abline(intercept = mlr2[1]$coefficients[1], slope = mlr2[1]$coefficients[2]
p2
```

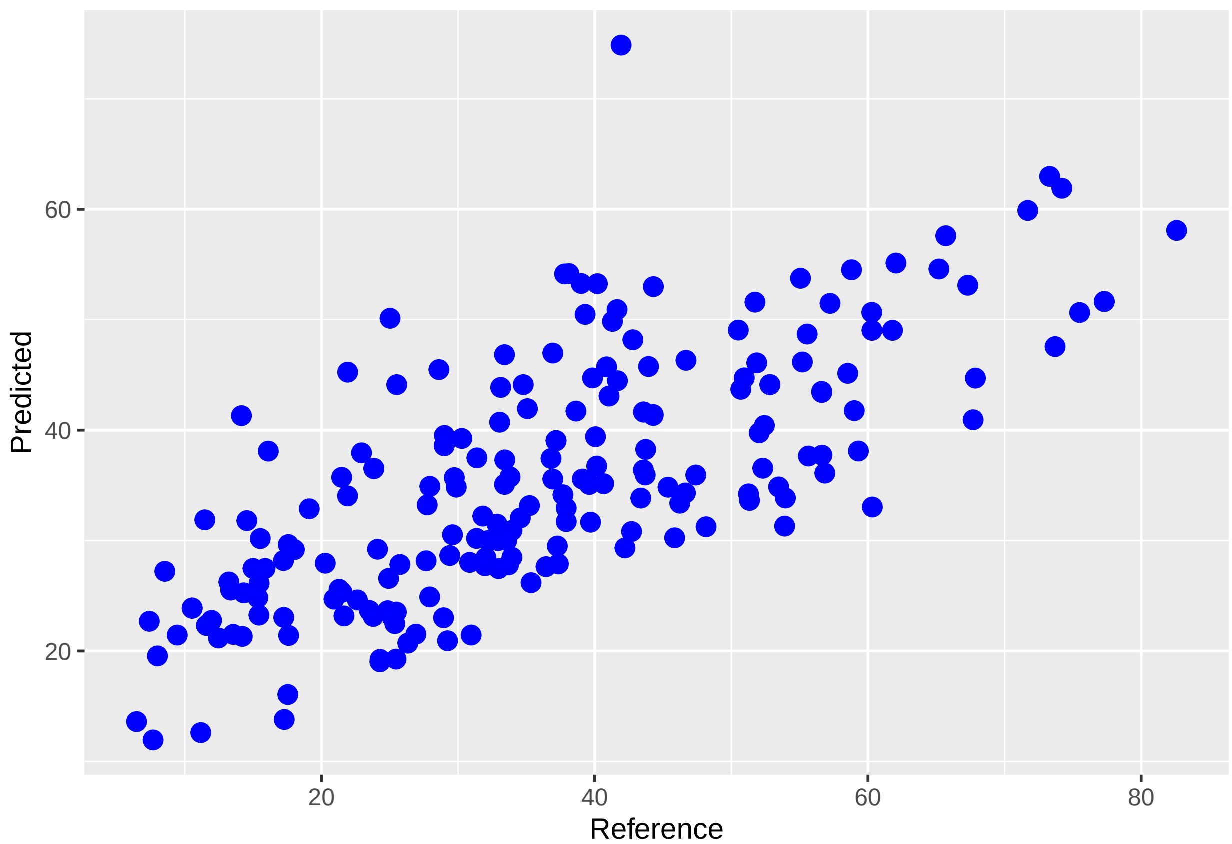



```
predict(mlr2, newdata=testConcrete)
```

```
##        5        9       46       53       57       59       62       66
## 52.99048 30.24674 41.94132 43.86628 74.86247 44.73077 51.58899 53.74696
##       68       75       76       77       80       81       86       95
## 39.24386 39.51304 54.14315 53.25591 49.84640 46.82809 45.47956 48.17841
##      120      125      127      134      148      149      152      153
## 49.03287 43.69808 49.03287 40.93484 46.16743 54.52053 47.56095 51.64523
##      156      159      168      173      175      181      182      185
## 51.64523 59.88911 61.90397 54.58891 50.64764 62.97581 58.08217 22.31186
##      193      194      199      203      206      215      221      228
## 28.16675 32.04716 27.88285 26.18268 24.69781 21.44321 23.52670 31.48833
##      234      240      243      251      255      261      270      273
## 31.30761 25.51733 30.19146 26.56872 21.50153 22.47991 26.14283 30.81696
##      275      291      295      296      299      309      314      317
```

```
## 28.19094 25.31457 22.69222 23.66232 31.24676 37.65524 39.75731 28.48180
##      319      322      331      333      337      338      342      345
## 34.83156 28.02147 29.33501 33.03904 33.63484 36.10419 35.93668 29.63561
##      348      354      365      368      369      370      373      374
## 34.30975 38.10702 29.17870 33.85283 37.73324 24.82922 29.50335 33.38376
##      381      386      389      395      401      402      404      411
## 51.47630 50.66445 48.69825 36.55167 44.70791 45.14206 41.76314 25.58925
##      413      423      425      430      437      441      452      456
## 30.18087 32.21419 30.26347 30.52258 27.79403 23.64799 34.85500 34.22661
##      463      467      471      477      481      488      490      491
## 34.14379 29.99775 36.74548 50.12366 50.47643 46.32108 44.11630 44.11630
##      494      508      511      512      520      537      538      547
## 44.46906 55.13004 53.27803 57.59939 38.25524 35.07904 45.75566 27.71585
##      551      556      561      568      580      585      586      587
## 33.22702 34.90490 27.83141 22.77160 24.62362 45.24702 40.71543 31.80347
##      589      596      598      599      602      606      610      626
## 19.55437 21.40639 37.48593 26.24739 28.45217 13.60726 29.21709 29.99299
##      630      640      644      648      654      655      656      660
## 12.61010 35.93808 27.95007 33.84585 19.26724 19.24208 30.92007 20.92704
##      670      678      681      682      687      693      707      713
## 38.09918 11.94738 13.79939 19.02245 41.30698 31.66281 20.71535 35.72610
##      720      721      723      727      730      732      735      740
## 36.38792 46.08908 23.04128 25.26563 32.93826 27.47459 35.14722 23.17522
##      741      744      750      754      771      775      780      781
## 21.44593 35.56992 46.97537 53.11884 27.47459 31.87711 23.05250 16.05747
##      782      783      784      792      800      804      812      815
## 21.32321 23.17522 28.64307 43.08442 50.91406 32.87426 24.89330 54.17276
##      826      828      830      848      849      854      859      869
## 35.56992 49.05787 39.06037 44.11570 35.08449 39.40242 40.42044 41.32499
##      883      885      893      902      903      905      913      916
## 27.63432 36.54931 30.01507 34.83605 43.50575 23.13330 41.64940 38.58012
##      918      919      923      924      930      935      938      942
## 21.51966 23.86199 35.74866 44.72209 34.04201 37.94102 37.42996 23.01140
##      947      948      951      954      958      966      967      968
## 27.45760 36.50329 41.72313 27.21346 35.69812 43.40666 21.17821 23.14649
##      976      979      982      986      987      993      994     1005
## 41.63053 38.61670 23.90034 35.76661 39.02379 23.25001 37.30403 44.12467
##     1014     1020     1025     1026
## 45.72832 33.16396 31.71799 41.40629
```

```r
predCompressive_Strength2 = data.frame(predict(mlr2, newdata=testConcrete))
names(predCompressive_Strength2)[1] = 'Predicted'
predCompressive_Strength2$Reference = testConcrete[,c('Compressive_Strength')]
plotpredmlr2 = qplot(Reference, Predicted, data=predCompressive_Strength2) + geom_point(colour = "blue"
plotpredmlr2
```



Model evaluation - $RMSEP$ and R^2

- Calculating predicted residual sum of squares (PRESS)

$$PRESS = \sum_{i=1}^{n} (y_i^{ref} - y_i^{pred})^2$$

```
PRESS2 = sum((predCompressive_Strength2$Reference - predCompressive_Strength2$Predicted)^2)
PRESS2
```

```
## [1] 26386.17
```

- Root mean squared error of prediction (RMSEP)

$$RMSEP = \sqrt{\frac{1}{n_T} \sum_{1}^{n_T} (y_i^{ref} - y_i^{pred})^2}$$

```
RMSEP2 = sqrt(PRESS2/ nrow(predCompressive_Strength2))
RMSEP2
```

```
## [1] 11.37295
```

- Total sum of squares (SST)

$$SST = \sum_{i=1}^{n}(y_i^{ref} - y_i^{mean})^2$$

```
SST2 = sum((predCompressive_Strength2$Reference - mean(predCompressive_Strength2$Reference))^2)
SST2
```

```
## [1] 55712.38
```

- Calculating R^2

$$R^2 = 1 - \frac{PRESS}{SST}$$

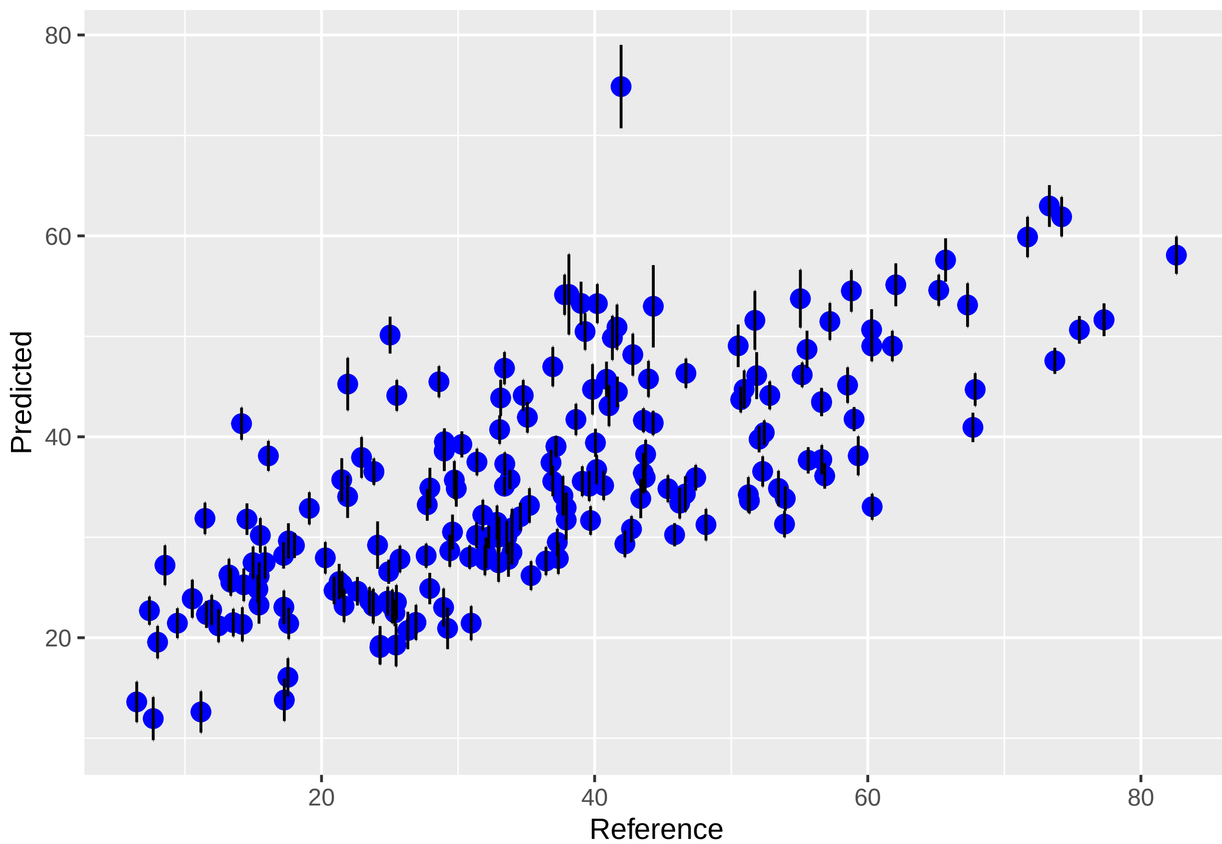
```
R22 = 1 - (PRESS2/SST2)
R22
```

```
## [1] 0.5263859
```

Predicted versus Reference

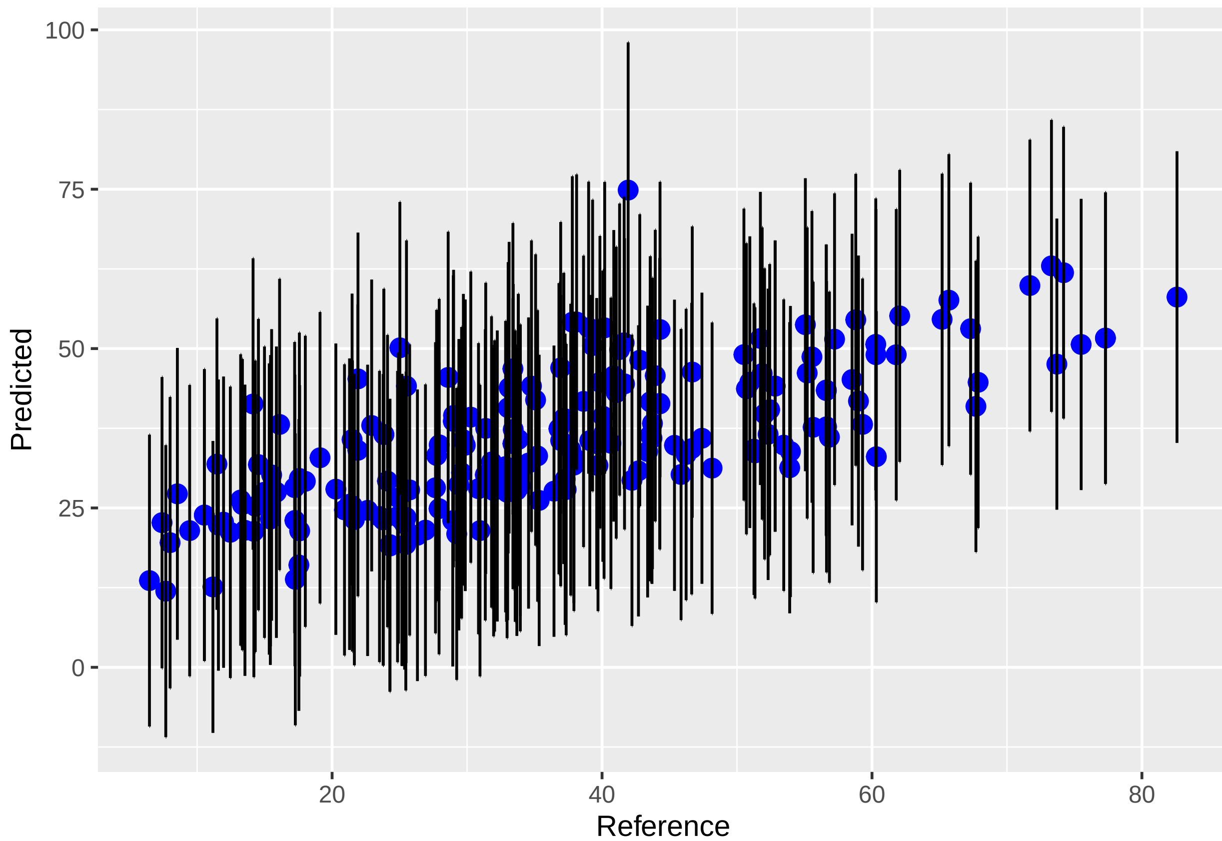
- Confidence Intervals (Narrow)

```
predCompressive_Strength2$lower = predict(mlr2, newdata=testConcrete, interval = "confidence")[,2]
predCompressive_Strength2$upper = predict(mlr2, newdata=testConcrete, interval = "confidence")[,3]
#predCompressive_Strength2
qplot(Reference, Predicted, data=predCompressive_Strength2) + geom_point(colour = "blue", size = 3) +
  geom_errorbar(aes(ymin = lower,ymax = upper))
```



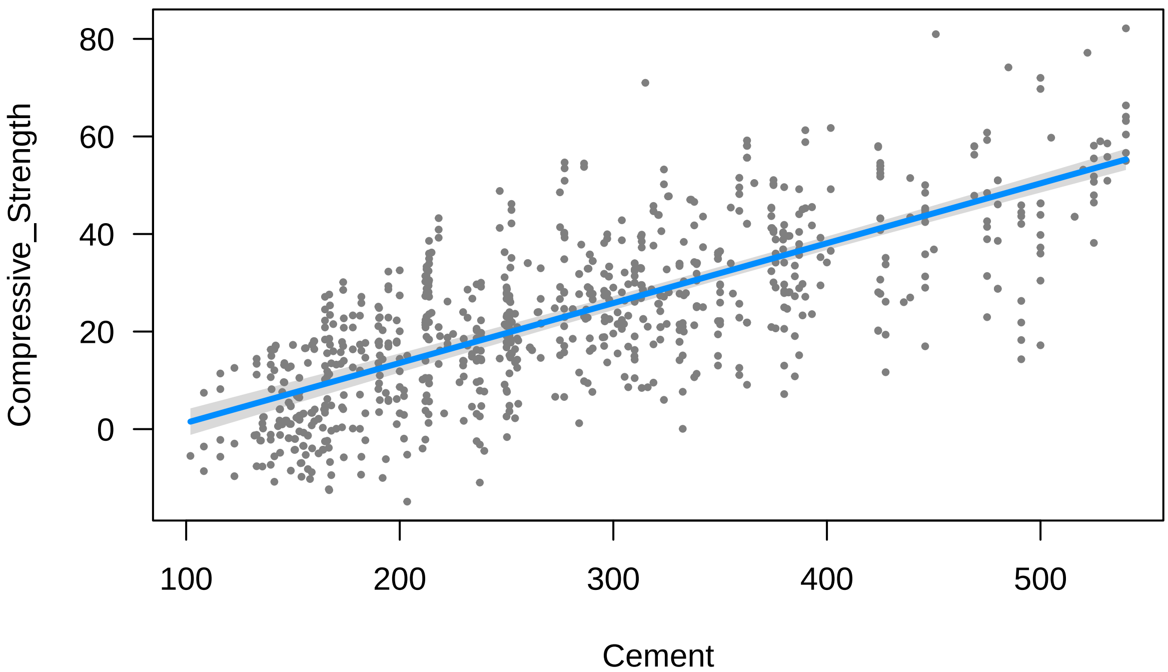
- Prediction Intervals (Tolerance/Wide)

```
predCompressive_Strength2$lower = predict(mlr2, newdata=testConcrete, interval = "prediction")[,2]
predCompressive_Strength2$upper = predict(mlr2, newdata=testConcrete, interval = "prediction")[,3]
#predCompressive_Strength2
qplot(Reference, Predicted, data=predCompressive_Strength2) + geom_point(colour = "blue", size = 3) +
  geom_errorbar(aes(ymin = lower,ymax = upper))
```

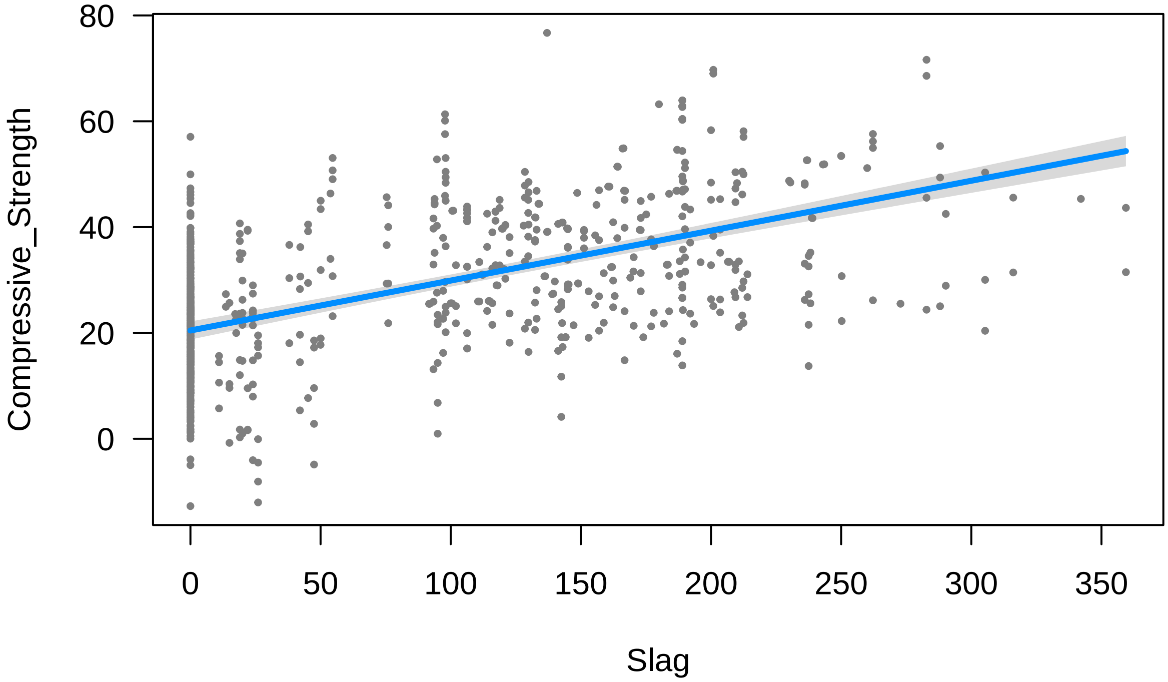


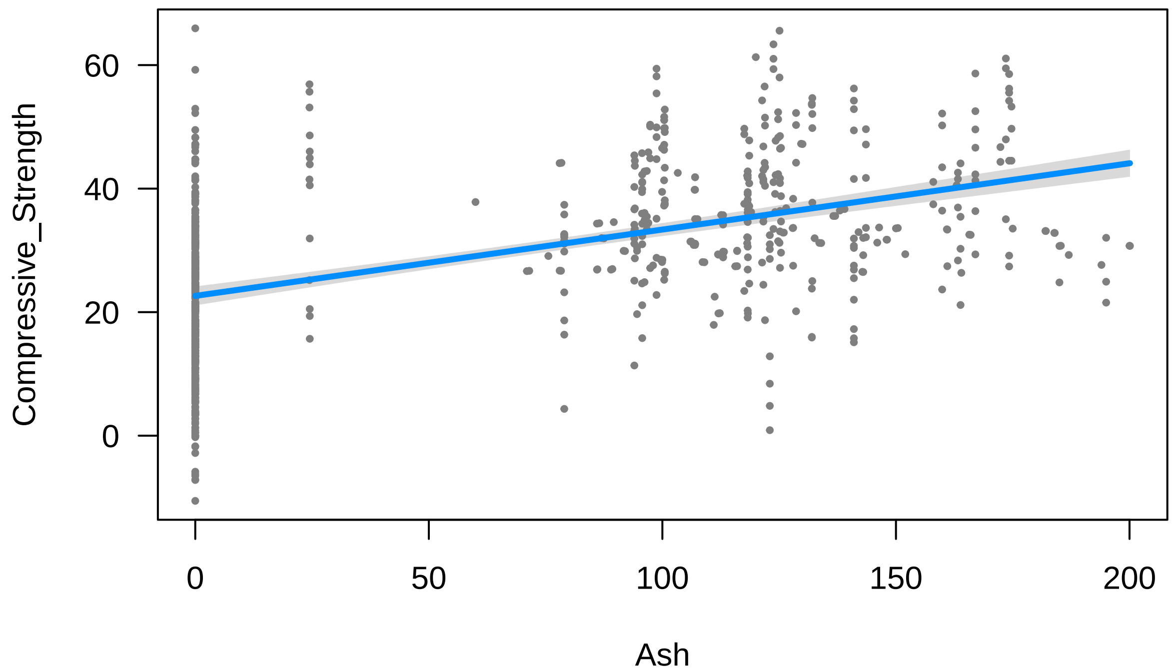
Visualizing using visreg package

```
visreg::visreg(mlr2)
```

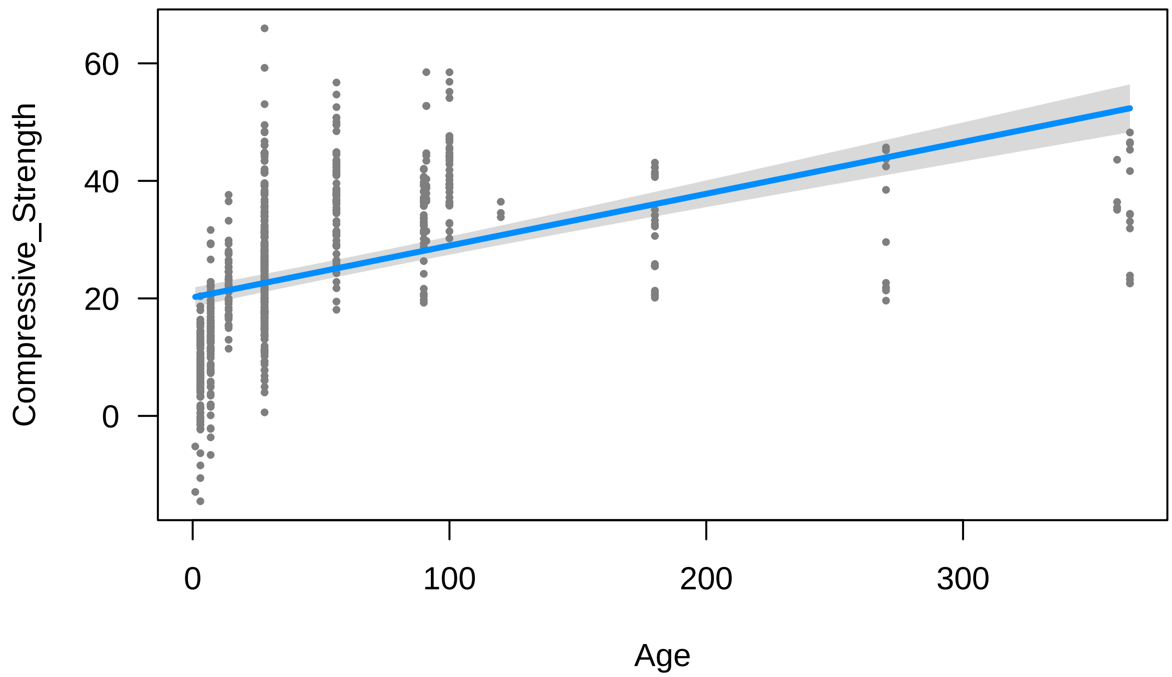


80
60
40
20
0
Compressive_Strength
100
200
300
400
500
Cement


Compressive_Strength
80
60
40
20
0
0
50
100
150
200
250
300
350
Slag



Compressive_Strength
60
40
20
0
0
50
100
150
200
Ash



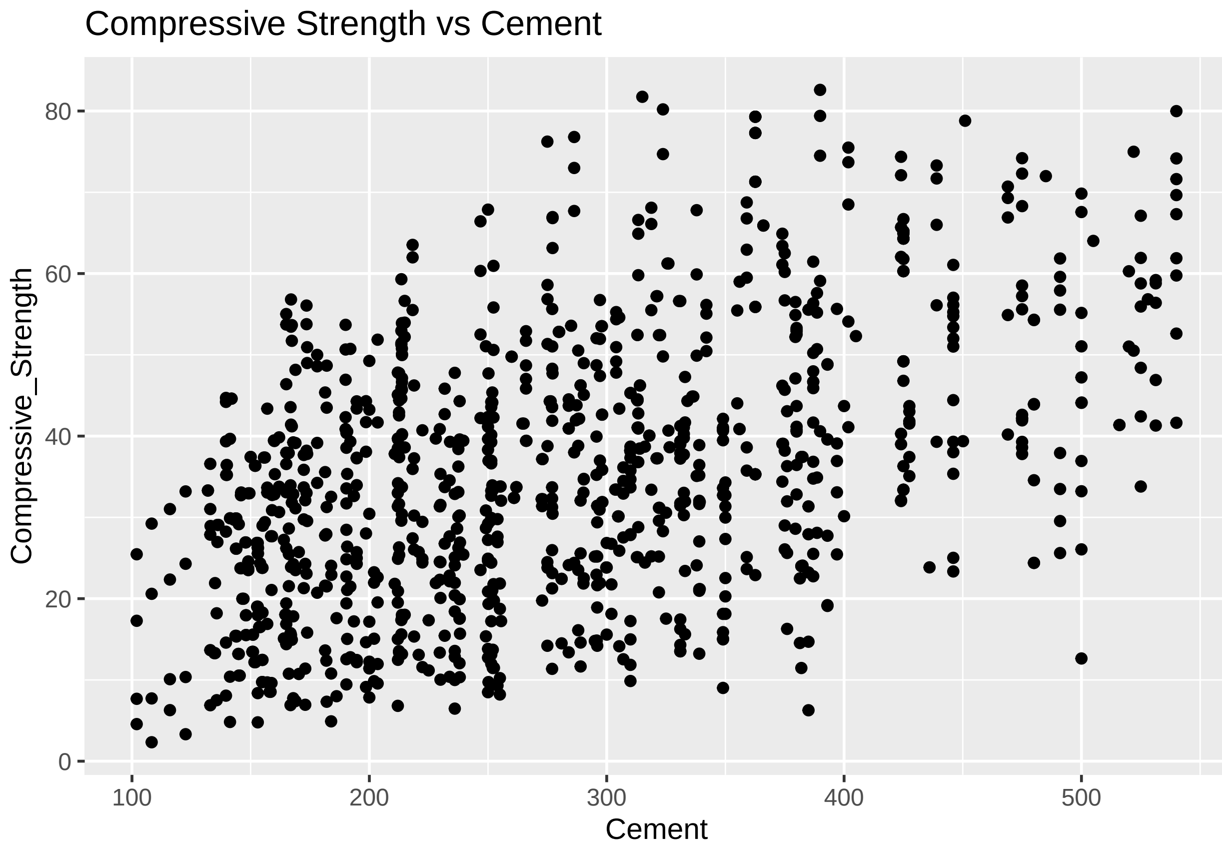
Generating a polynomial multiple linear regression model

```
polymlr = lm(Compressive_Strength ~ poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate +
#fitted(polymlr)
#resid(polymlr)
#polymlr
summary(polymlr)
```

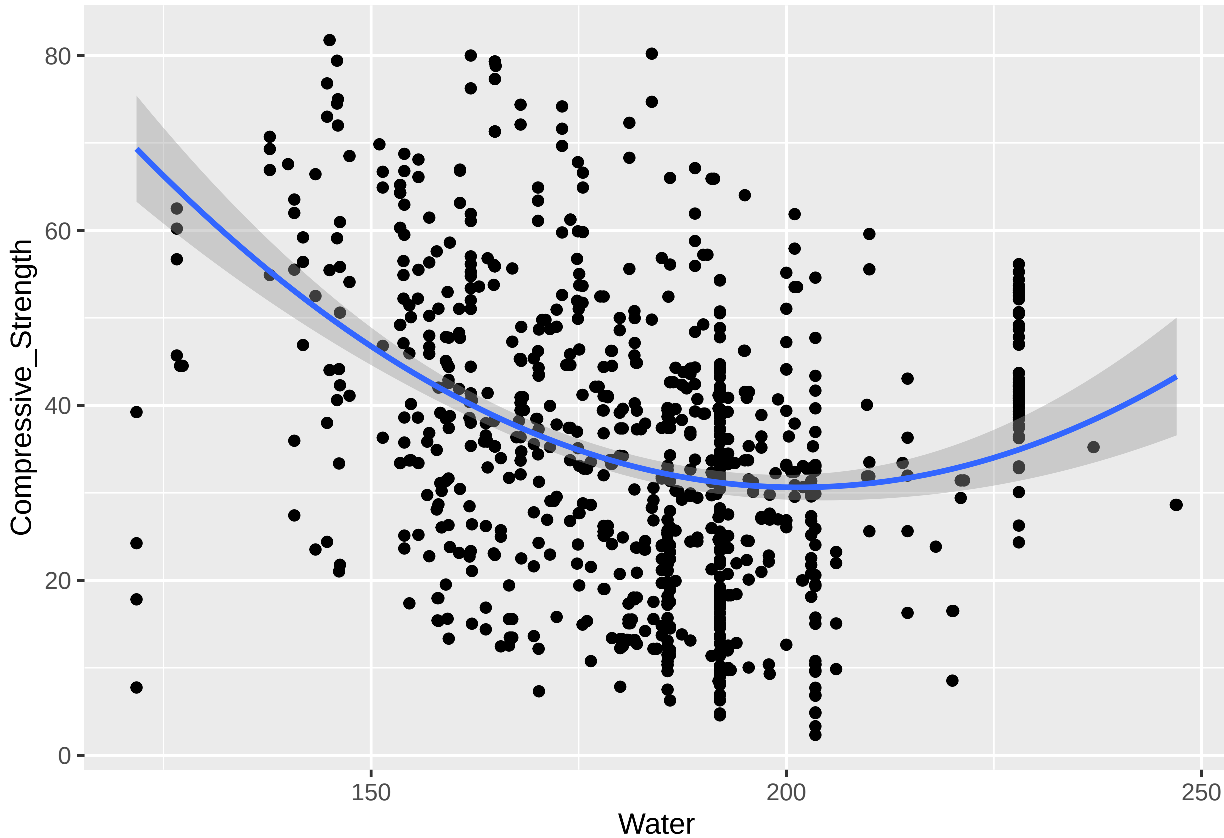
```
##
## Call:
## lm(formula = Compressive_Strength ~ poly(Cement + Slag + Ash +
##     Water + Superplasticizer + C_Aggregate + F_Aggregate + Age,
##     5), data = trainConcrete)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -31.823  -9.835  -0.885   8.943  45.690
##
## Coefficients:
##                                                                                     Estimate
## (Intercept)                                                                          35.7284
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)1 290.2800
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)2 -57.5966
```

```
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)3 -96.6034
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)4  43.5343
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)5  46.4451
##                                                                                             Std. Erro
## (Intercept)                                                                                     0.438
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)1    12.613
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)2    12.613
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)3    12.613
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)4    12.613
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)5    12.613
##                                                                                             t value
## (Intercept)                                                                                  81.408
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)1  23.013
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)2  -4.566
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)3  -7.659
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)4   3.451
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)5   3.682
##                                                                                             Pr(>|t|)
## (Intercept)                                                                                  < 2e-16
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)1  < 2e-16
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)2 5.73e-06
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)3 5.29e-14
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)4 0.000586
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)5 0.000246
##
## (Intercept)                                                                                 ***
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)1 ***
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)2 ***
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)3 ***
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)4 ***
## poly(Cement + Slag + Ash + Water + Superplasticizer + C_Aggregate + F_Aggregate + Age, 5)5 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.61 on 820 degrees of freedom
## Multiple R-squared:  0.4363, Adjusted R-squared:  0.4328
## F-statistic: 126.9 on 5 and 820 DF,  p-value: < 2.2e-16
```

```r
#plot(polymlr)
polyp1 = strengthVsCement + geom_abline(intercept = polymlr[1]$coefficients[1], slope = polymlr[1]$coef
polyp1
```




```
ggplot(trainConcrete, aes(Water, Compressive_Strength) ) + geom_point() +
stat_smooth(method = lm, formula = y ~ poly(x, 2, raw = TRUE))
```

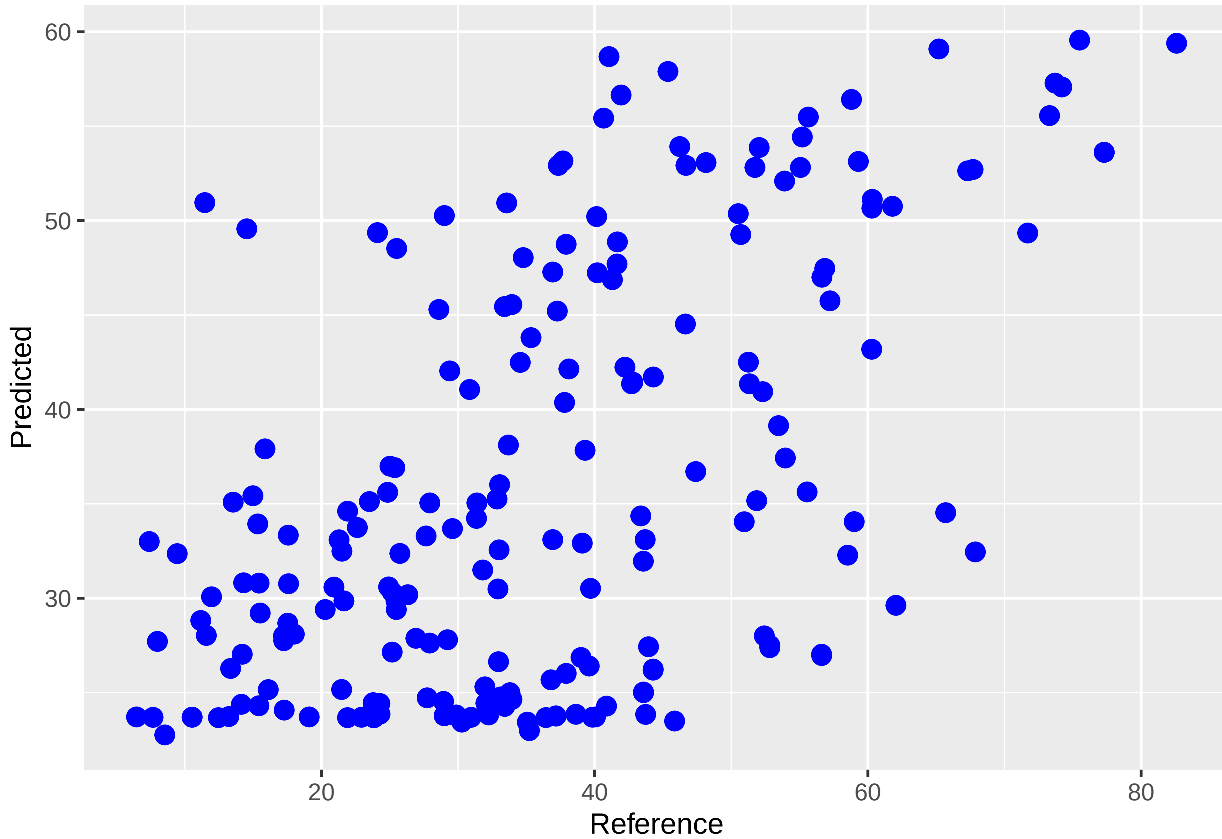



```r
predict(polymlr, newdata=testConcrete)
```

```
##        5        9       46       53       57       59       62       66
## 41.71701 23.49457 23.43686 24.77183 56.65083 34.04772 52.81526 52.81526
##       68       75       76       77       80       81       86       95
## 23.43686 50.26645 40.36885 47.23459 46.87155 45.44518 45.29260 41.44354
##      120      125      127      134      148      149      152      153
## 50.75817 49.26005 50.66048 52.70797 54.42773 56.41653 57.28469 53.61689
##      156      159      168      173      175      181      182      185
## 53.61689 49.34161 57.07640 59.09037 59.56060 55.56027 59.39423 28.02813
##      193      194      199      203      206      215      221      228
## 33.29687 42.48935 52.92193 43.80159 30.58789 32.35946 29.40020 35.25365
##      234      240      243      251      255      261      270      273
## 52.09433 26.27931 34.22738 30.59766 35.08800 36.91494 30.79942 41.35558
##      275      291      295      296      299      309      314      317
## 28.00084 32.49109 33.00150 35.11788 53.07374 55.48451 53.86999 45.55404
##      319      322      331      333      337      338      342      345
## 57.89908 41.05456 42.23423 51.12438 41.35558 47.46837 36.70709 33.34053
##      348      354      365      368      369      370      373      374
## 44.52247 53.13165 28.09738 37.42578 47.00205 33.93432 45.20749 53.92176
##      381      386      389      395      401      402      404      411
## 45.74969 43.18468 35.62769 40.93824 32.45134 32.28051 34.05350 33.09417
##      413      423      425      430      437      441      452      456
## 29.21381 31.49393 32.56175 33.68609 38.11366 35.60950 39.13878 42.50325
##      463      467      471      477      481      488      490      491
```

```
## 53.16141 50.93511 50.21287 36.99207 37.83330 52.92370 48.52471 48.04022
##      494      508      511      512      520      537      538      547
## 48.87237 29.62293 26.86068 34.52926 23.84496 26.40673 27.42898 25.30485
##      551      556      561      568      580      585      586      587
## 24.72395 35.04621 32.37025 30.07417 33.74127 34.61157 36.01362 49.56500
##      589      596      598      599      602      606      610      626
## 27.71154 30.76644 35.04621 23.73055 24.44262 23.71122 49.36198 30.48887
##      630      640      644      648      654      655      656      660
## 28.81885 33.10160 29.40020 34.35750 29.85589 23.86246 24.63752 27.79965
##      670      678      681      682      687      693      707      713
## 25.15659 23.68998 24.07236 24.42308 24.38021 30.52126 30.18464 25.16381
##      720      721      723      727      730      732      735      740
## 31.96066 35.16574 27.74918 30.81594 48.74646 37.90964 55.42877 30.32795
##      741      744      750      754      771      775      780      781
## 23.69068 33.10160 47.27716 52.63611 35.42612 50.95254 27.14558 28.67828
##      782      783      784      792      800      804      812      815
## 27.03097 29.85589 42.03806 58.68487 47.70121 23.71122 27.62441 42.14214
##      826      828      830      848      849      854      859      869
## 32.91656 50.36544 23.76810 27.38094 24.34788 23.71122 28.00474 26.18811
##      883      885      893      902      903      905      913      916
## 23.67697 23.67023 23.81687 23.81687 27.03097 24.44262 25.03003 23.78306
##      918      919      923      924      930      935      938      942
## 27.87596 23.69861 24.96252 23.69861 23.67427 23.69381 25.67672 24.53421
##      947      948      951      954      958      966      967      968
## 26.63535 23.67014 23.85192 22.75941 23.80100 26.96318 23.67250 24.46746
##      976      979      982      986      987      993      994     1005
## 24.99602 23.77393 23.69772 24.99602 23.75570 24.30325 24.26883 27.50166
##     1014     1020     1025     1026
## 24.28590 22.98643 26.01668 26.24674
```

```r
predCompressive_Strength3 = data.frame(predict(polymlr, newdata=testConcrete))
names(predCompressive_Strength3)[1] = 'Predicted'
predCompressive_Strength3$Reference = testConcrete[,c('Compressive_Strength')]
plotpredpolymlr = qplot(Reference, Predicted, data=predCompressive_Strength3) + geom_point(colour = "blu
plotpredpolymlr
```



Model evaluation - $RMSEP$ and R^2

- Calculating predicted residual sum of squares (PRESS)

$$PRESS = \sum_{i=1}^{n} (y_i^{ref} - y_i^{pred})^2$$

```
PRESS3 = sum((predCompressive_Strength3$Reference - predCompressive_Strength3$Predicted)^2)
PRESS3
```

```
## [1] 38521.65
```

- Root mean squared error of prediction (RMSEP)

$$RMSEP = \sqrt{\frac{1}{n_T} \sum_{1}^{n_T} (y_i^{ref} - y_i^{pred})^2}$$

```
RMSEP3 = sqrt(PRESS3/ nrow(predCompressive_Strength3))
RMSEP3
```

```
## [1] 13.7416
```

- Total sum of squares (SST)

$$SST = \sum_{i=1}^{n} (y_i^{ref} - y_i^{mean})^2$$

```
SST3 = sum((predCompressive_Strength3$Reference - mean(predCompressive_Strength3$Reference))^2)
SST3
```

```
## [1] 55712.38
```

- Calculating R^2

$$R^2 = 1 - \frac{PRESS}{SST}$$

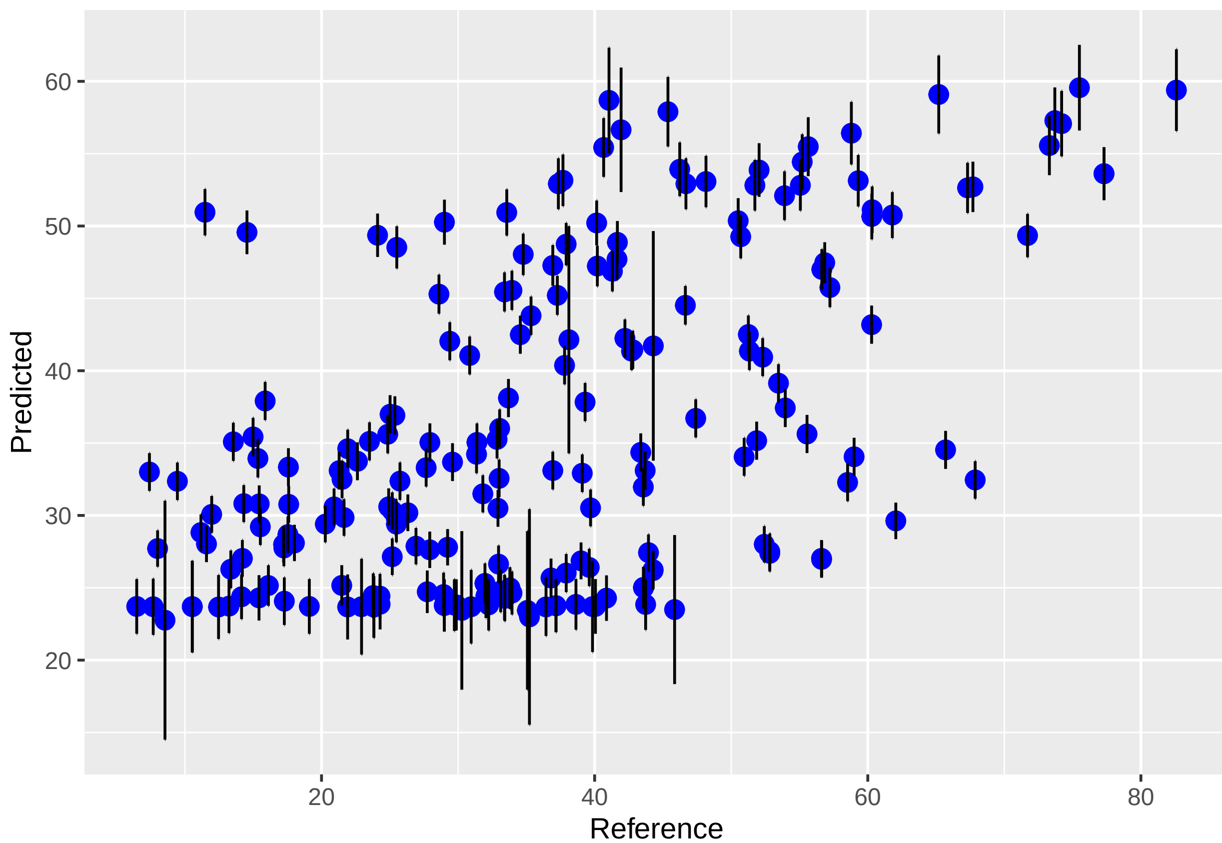
```
R23 = 1 - (PRESS3/SST3)
R23
```

```
## [1] 0.3085622
```

Predicted versus Reference

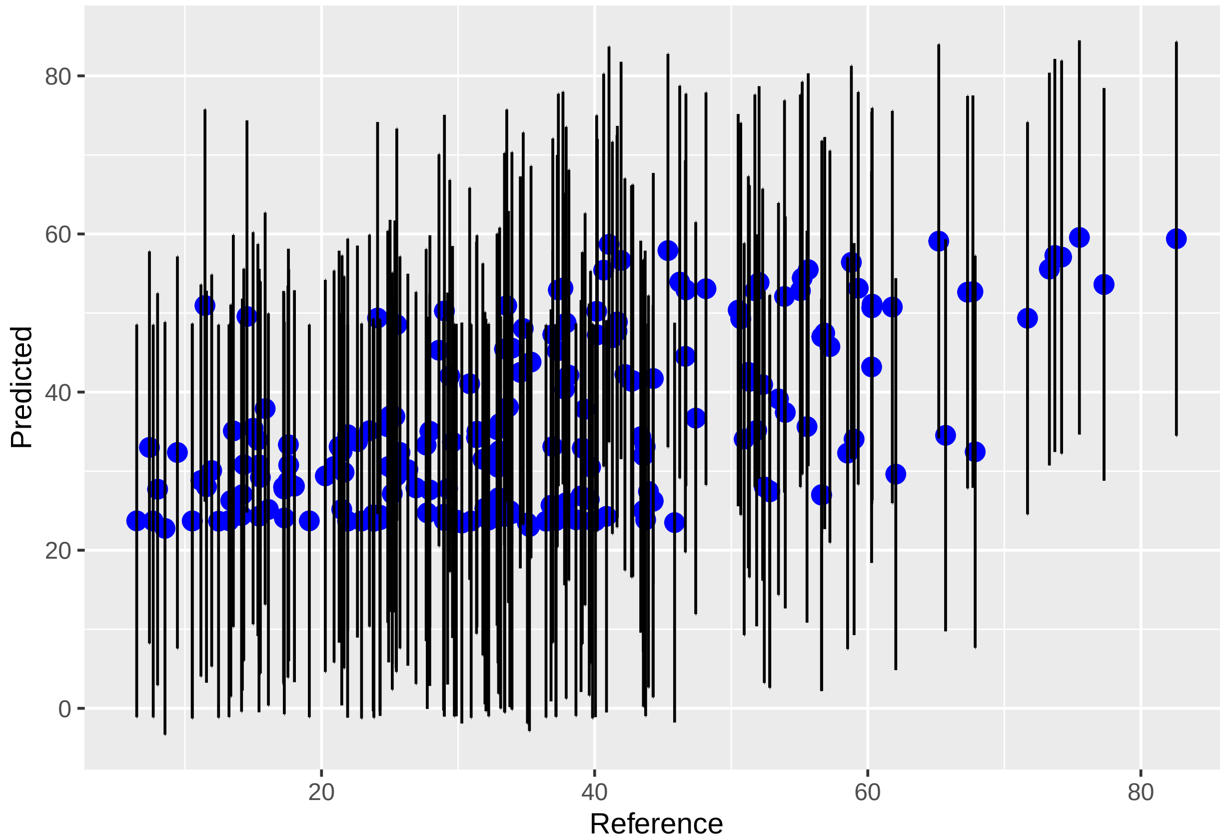
- Confidence Intervals (Narrow)

```
predCompressive_Strength3$lower = predict(polymlr, newdata=testConcrete, interval = "confidence")[,2]
predCompressive_Strength3$upper = predict(polymlr, newdata=testConcrete, interval = "confidence")[,3]
#predCompressive_Strength3
qplot(Reference, Predicted, data=predCompressive_Strength3) + geom_point(colour = "blue", size = 3) +
  geom_errorbar(aes(ymin = lower,ymax = upper))
```



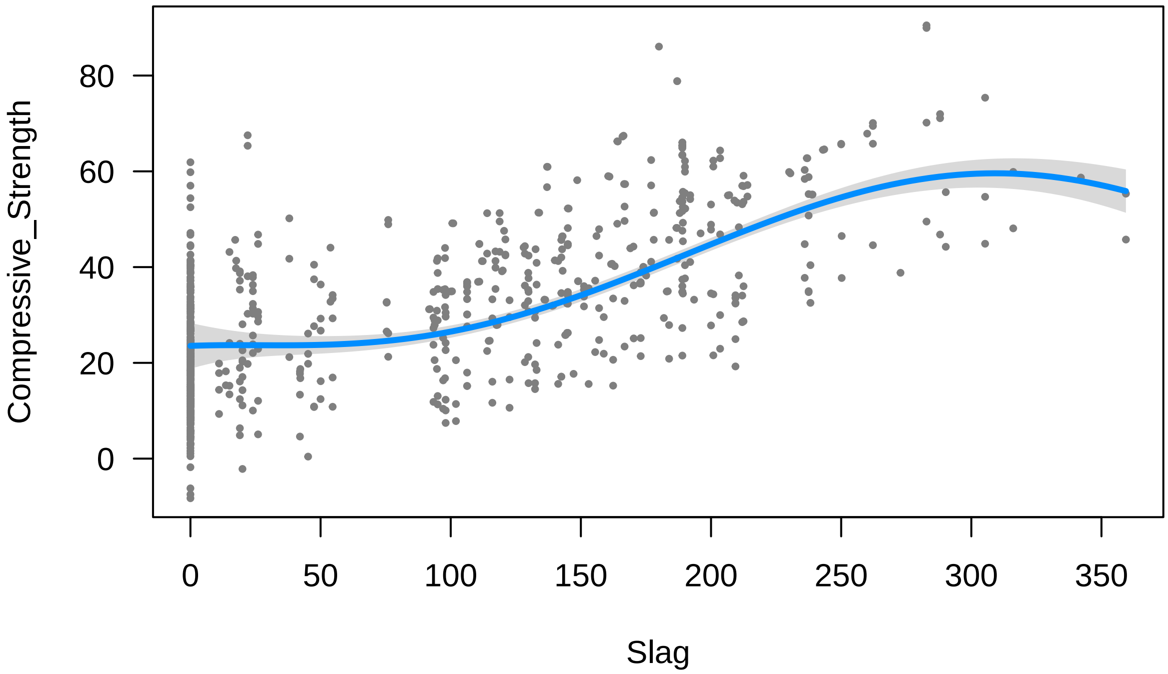
* Prediction Intervals (Tolerance/Wide)

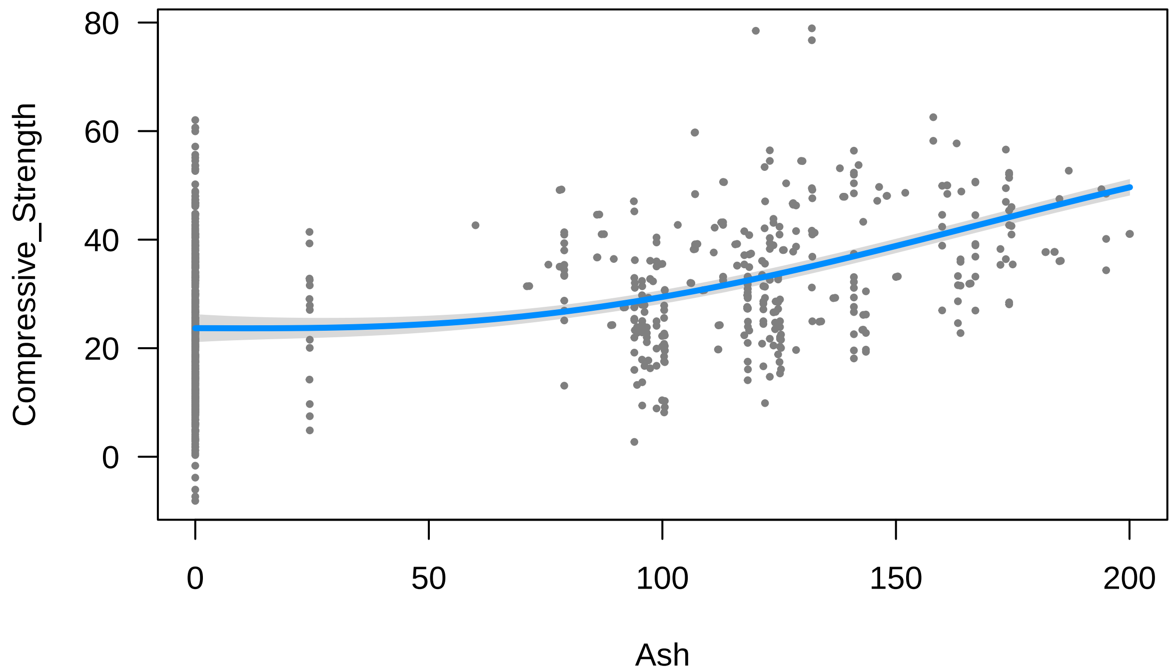
```
predCompressive_Strength3$lower = predict(polymlr, newdata=testConcrete, interval = "prediction")[,2]
predCompressive_Strength3$upper = predict(polymlr, newdata=testConcrete, interval = "prediction")[,3]
#predCompressive_Strength3
qplot(Reference, Predicted, data=predCompressive_Strength3) + geom_point(colour = "blue", size = 3) +
  geom_errorbar(aes(ymin = lower,ymax = upper))
```



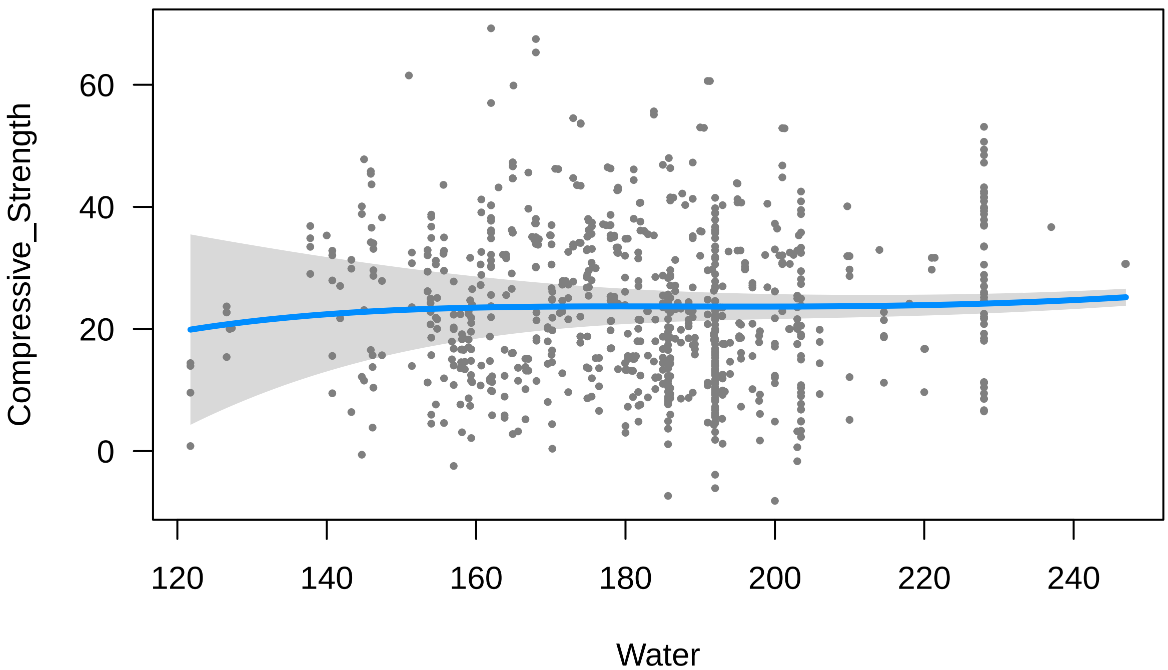
Visualizing using `visreg` package

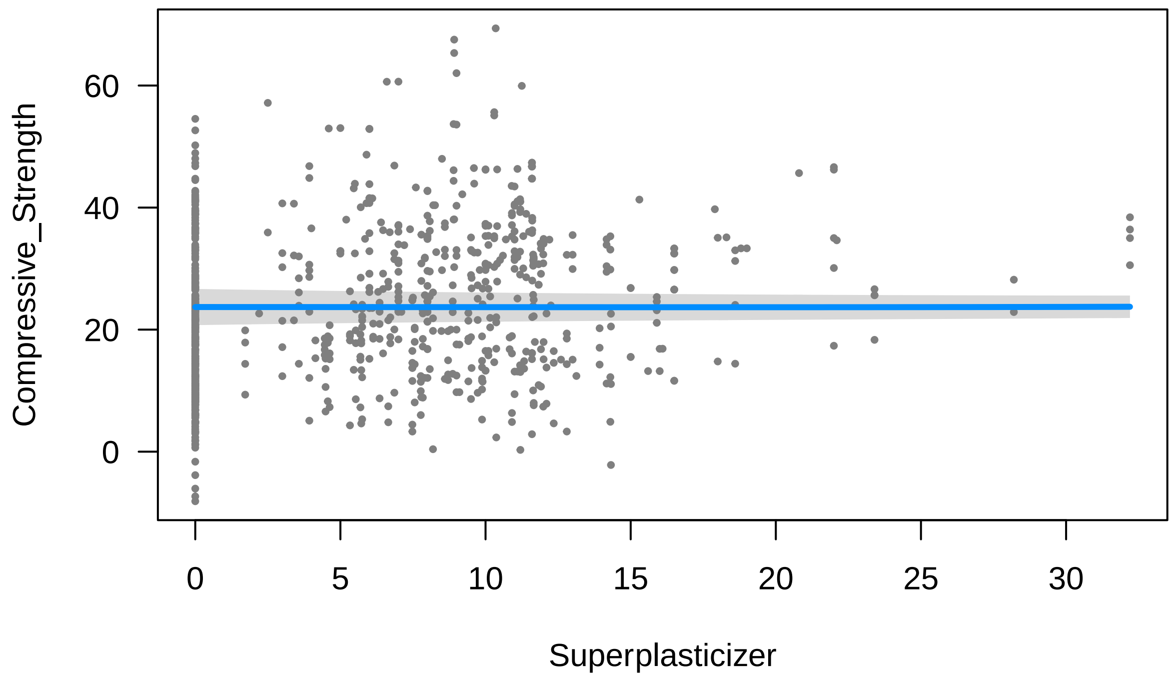
```
visreg::visreg(polymlr)
```

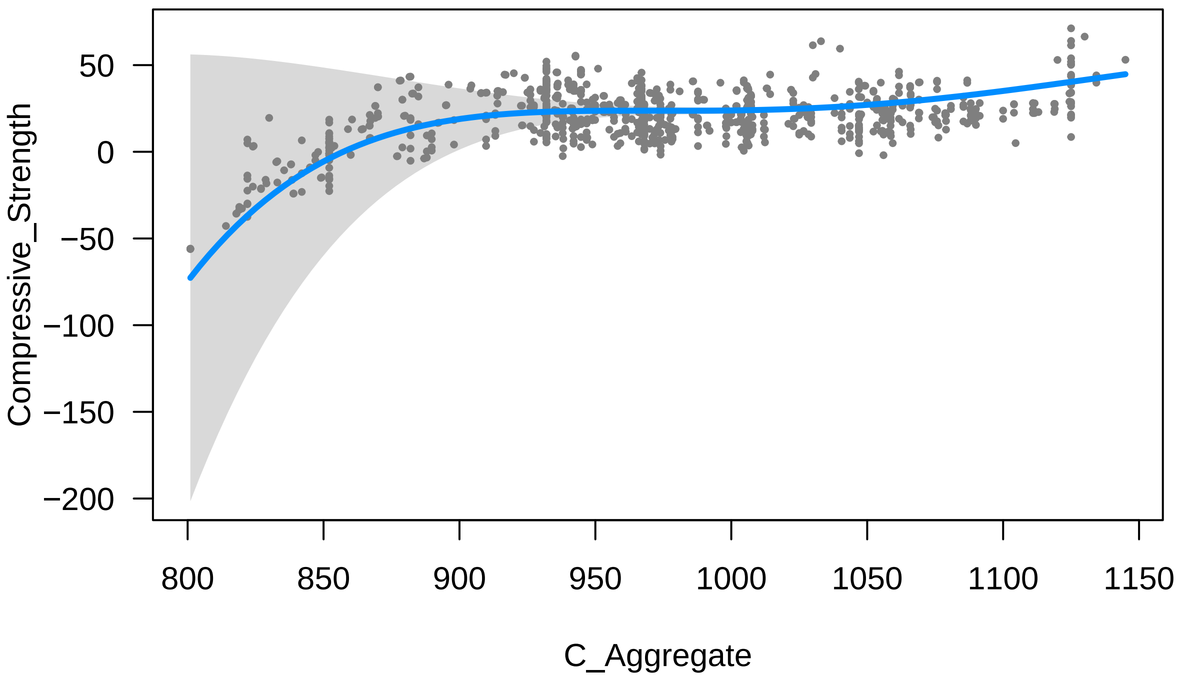

Compressive_Strength
80
60
40
20
0
0
50
100
150
200
250
300
350
Slag

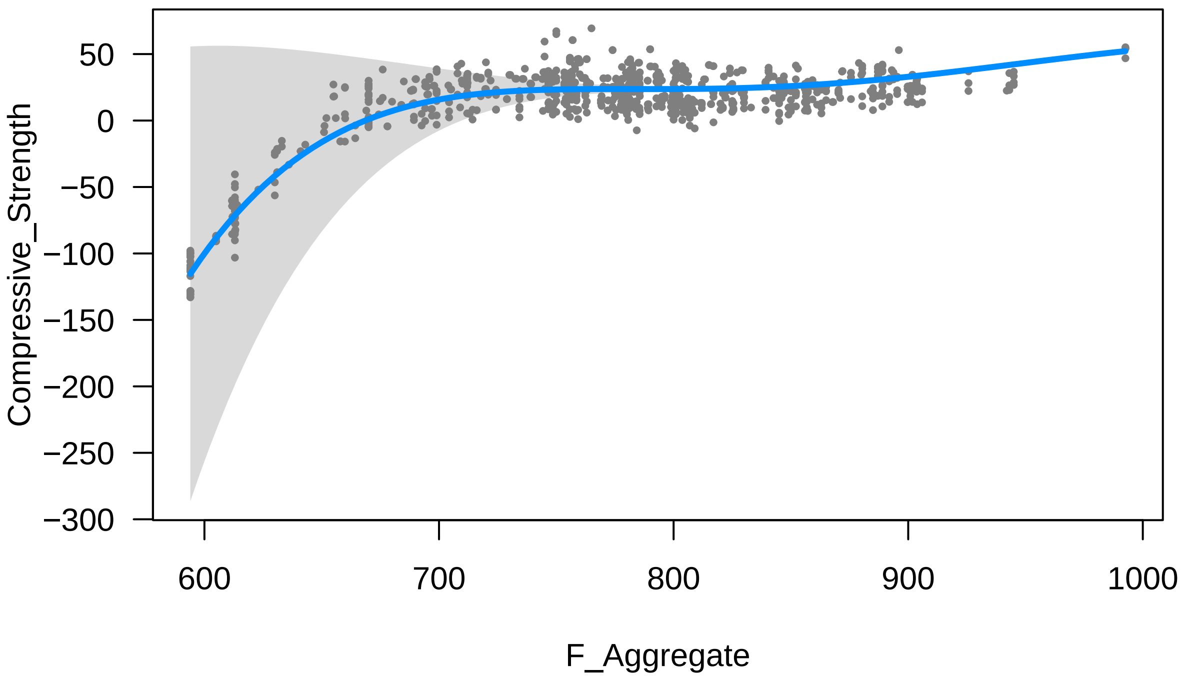


80
60
40
20
0
Compressive_Strength
0
50
100
150
200
Ash


Compressive_Strength
60
40
20
0
120
140
160
180
200
220
240
Water


Compressive_Strength
60
40
20
0
0
5
10
15
20
25
30
Superplasticizer


Compressive_Strength
50
0
−50
−100
−150
−200
800
850
900
950
1000
1050
1100
1150
C_Aggregate



50
0
−50
−100
−150
−200
−250
−300
Compressive_Strength
600
700
800
900
1000
F_Aggregate