Assignment 10

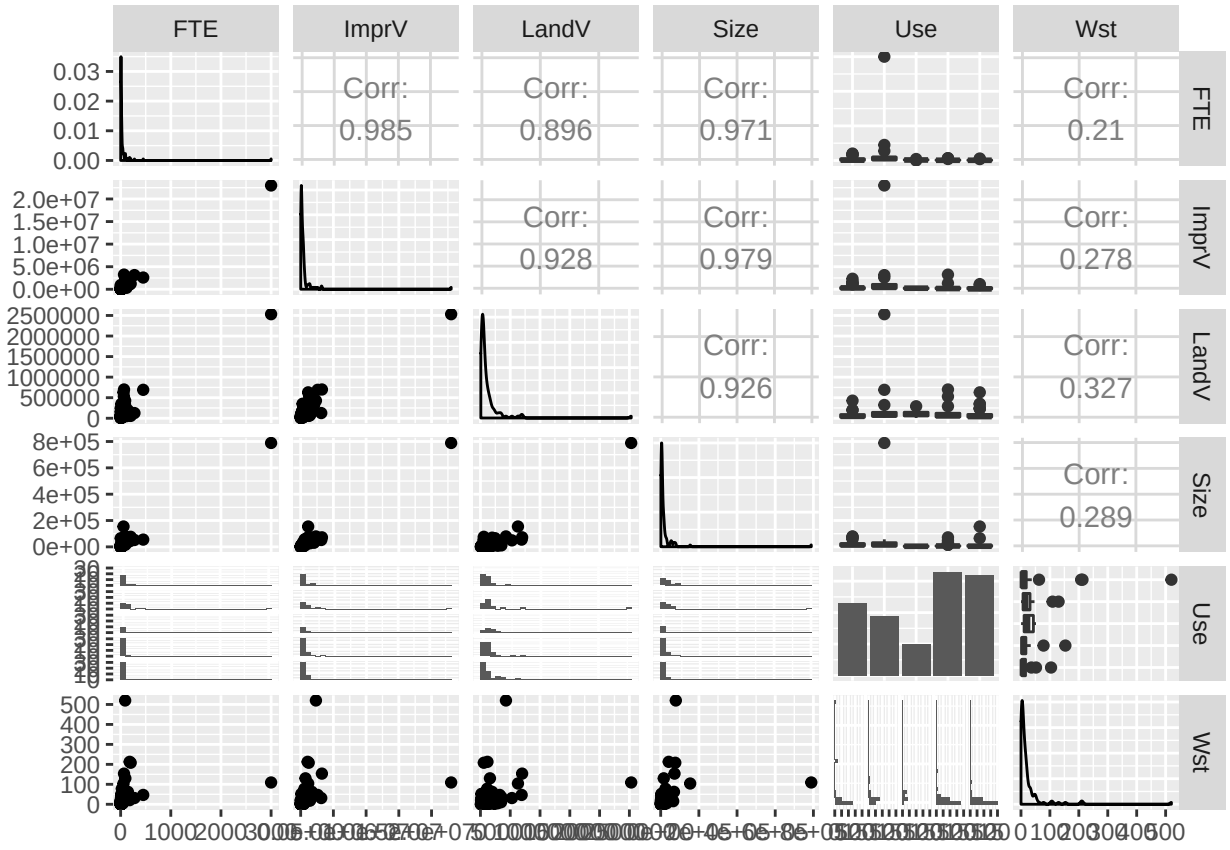
Chase Baggett

Transforming the Data

Looking at our training data, we have very non-normal data due to considerible right skew. What is happening is that we have alot of very small companies in our data, and very few large ones. These large companies have very extreme values of all of our predictors relative to the mean. Because we have this right skewed nature to our data, we are going to take the log of each of our regressors.

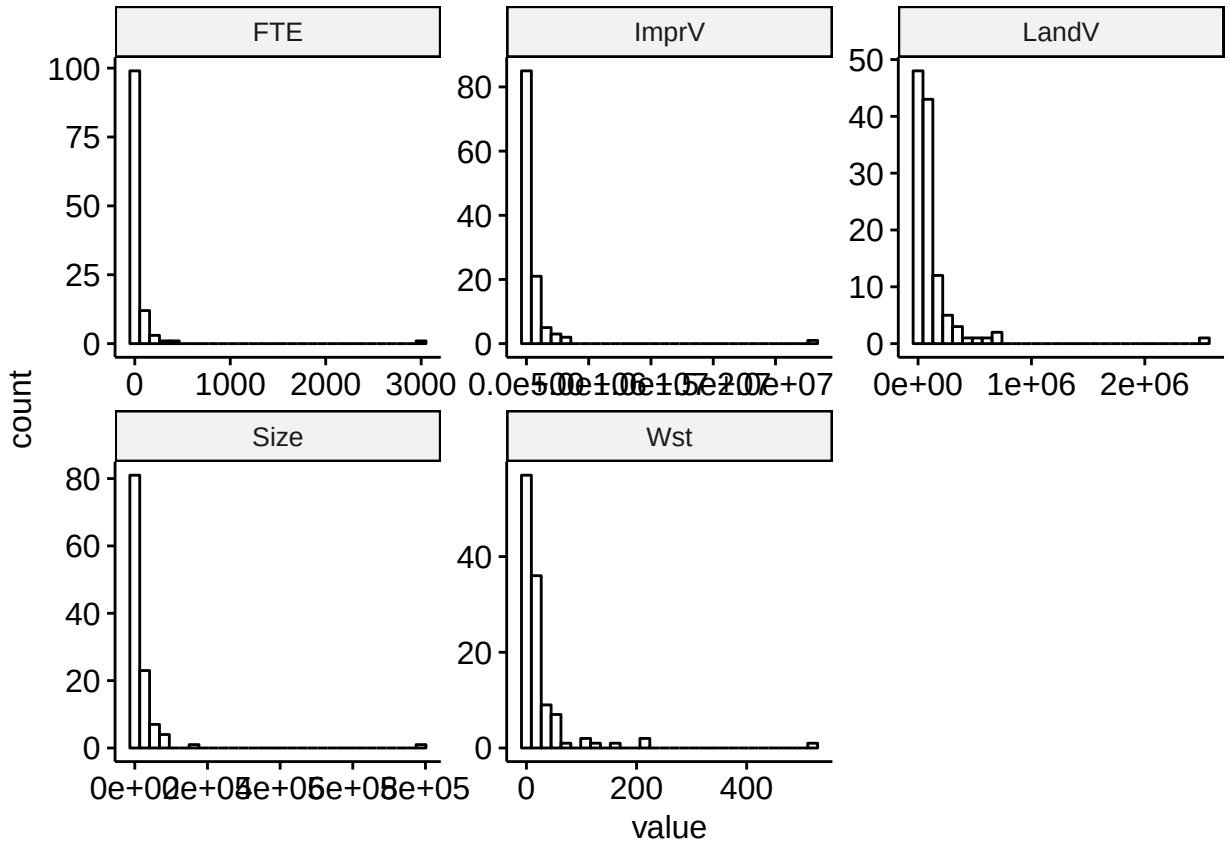
```
ggpairs(train)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



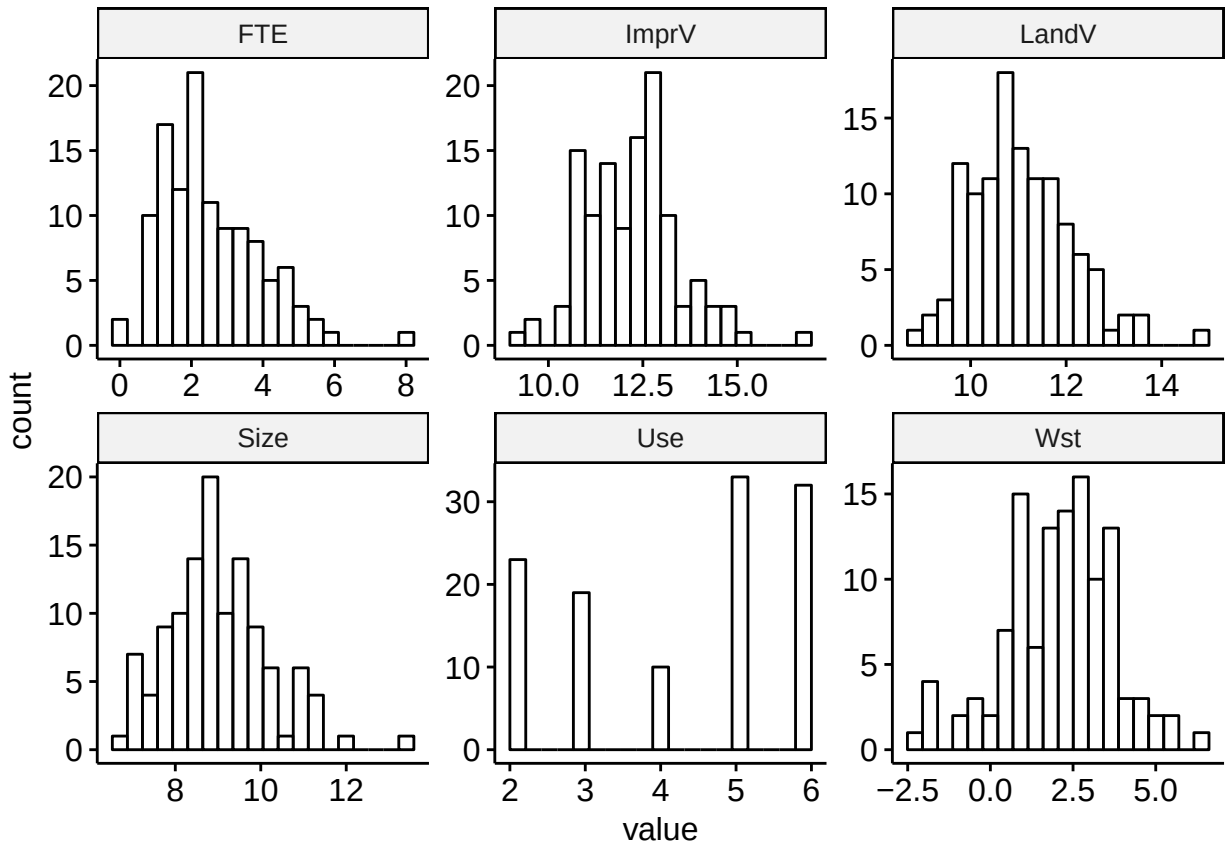
Before Transformation

```
all_features <- gather(train[,c("FTE","ImprV","LandV","Size","Wst")])
all_features$value <- as.numeric(all_features$value)
gghistogram(all_features,x = "value") + facet_wrap(~key,scales = "free")
```



Before Transformation

```
train$ImprV <- log(train$ImprV)
train$FTE <- log(train$FTE)
train$LandV <- log(train$LandV)
train$Size <- log(train$Size)
train$Wst <- log(train$Wst)
all_features <- gather(train[,c("FTE","ImprV","LandV","Size","Wst","Use")])
all_features$value <- as.numeric(all_features$value)
gghistogram(all_features,x = "value",bins=20) + facet_wrap(~key,scales = "free")
```



Forming a Simple Model to Compete Against our Complex Model

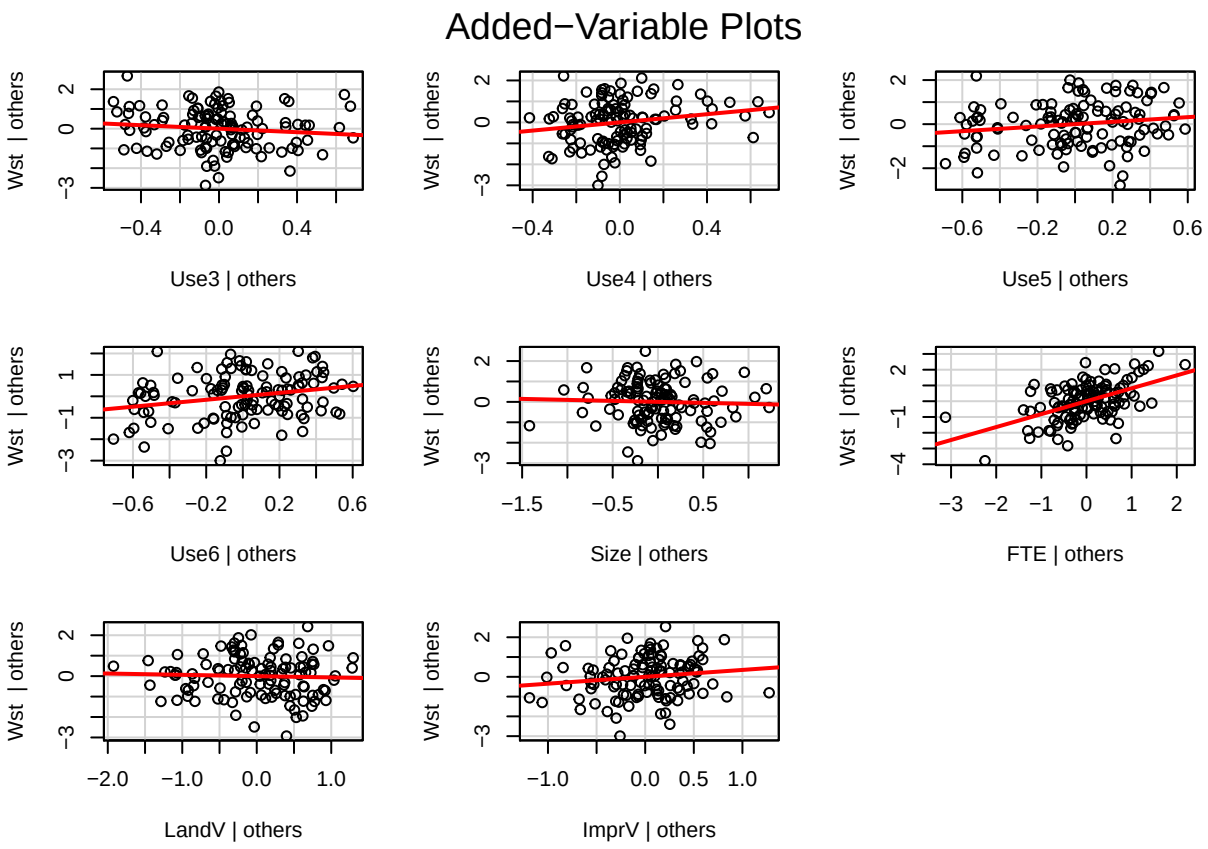
I'm going to use a non-interacted model.

```
simple_model <- lm(Wst ~ Use  + Size + FTE + LandV + ImprV,data=train)
summary(simple_model)
```

```
##
## Call:
## lm(formula = Wst ~ Use + Size + FTE + LandV + ImprV, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9050 -0.7874  0.1044  0.6310  2.4660
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.07356    1.48553  -2.069   0.0409 *
## Use3        -0.44408    0.36439  -1.219   0.2256
## Use4         0.97618    0.46987   2.078   0.0401 *
## Use5         0.53068    0.30827   1.721   0.0880 .
## Use6         0.80584    0.31106   2.591   0.0109 *
## Size        -0.09770    0.22502  -0.434   0.6650
## FTE          0.82028    0.12851   6.383 4.46e-09 ***
## LandV       -0.06377    0.14859  -0.429   0.6687
```

```
## ImprV        0.34781    0.22868   1.521   0.1312
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.047 on 108 degrees of freedom
## Multiple R-squared:  0.6023, Adjusted R-squared:  0.5728
## F-statistic: 20.44 on 8 and 108 DF,  p-value: < 2.2e-16
```

```
avPlots(simple_model)
```



Looking at the added variable plots, and the p-values, we remove Size, LandV, and ImprV as they seem to give little additional information and lack significance.

```
simple_model <- lm(Wst ~ Use + FTE,data=train)
summary(simple_model)
```

```
##
## Call:
## lm(formula = Wst ~ Use + FTE, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.13864 -0.77056  0.05003  0.64503  2.63500
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.75814    0.31334  -2.420   0.0172 *
## Use3        -0.44060    0.33658  -1.309   0.1932
```

```
## Use4         0.96505    0.39785   2.426   0.0169 *
## Use5         0.49320    0.29021   1.699   0.0920 .
## Use6         0.72245    0.29499   2.449   0.0159 *
## FTE          0.97270    0.08026  12.119   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.05 on 111 degrees of freedom
## Multiple R-squared:  0.589,  Adjusted R-squared:  0.5705
## F-statistic: 31.81 on 5 and 111 DF,  p-value: < 2.2e-16
```

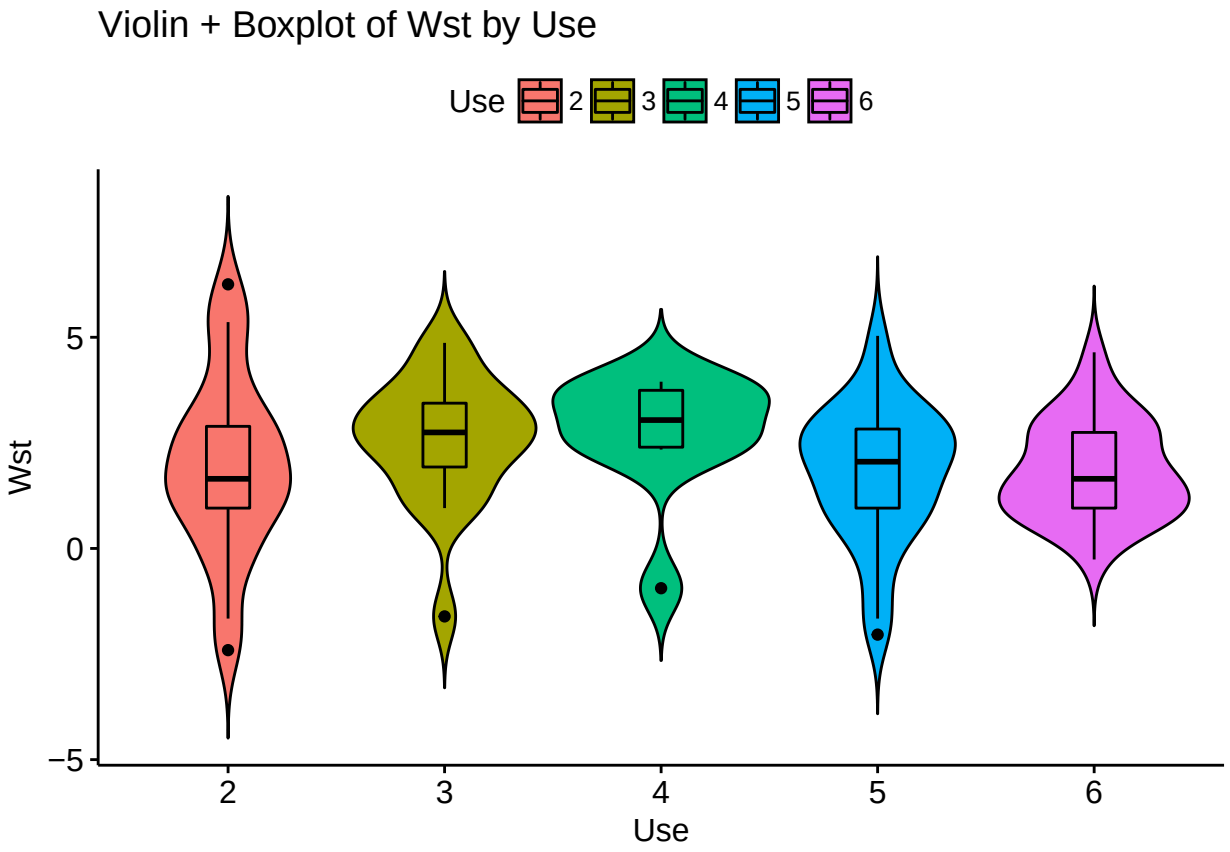
Forming an Alternative Hypothesis

If we think about the variables we have, many of them are measures of the size of the company in question. The basic hypothesis at play here is that as companies become larger, they generate more waste. However, each of these variables measures a different type of company size. Number of employees, size of their location, land value, etc.

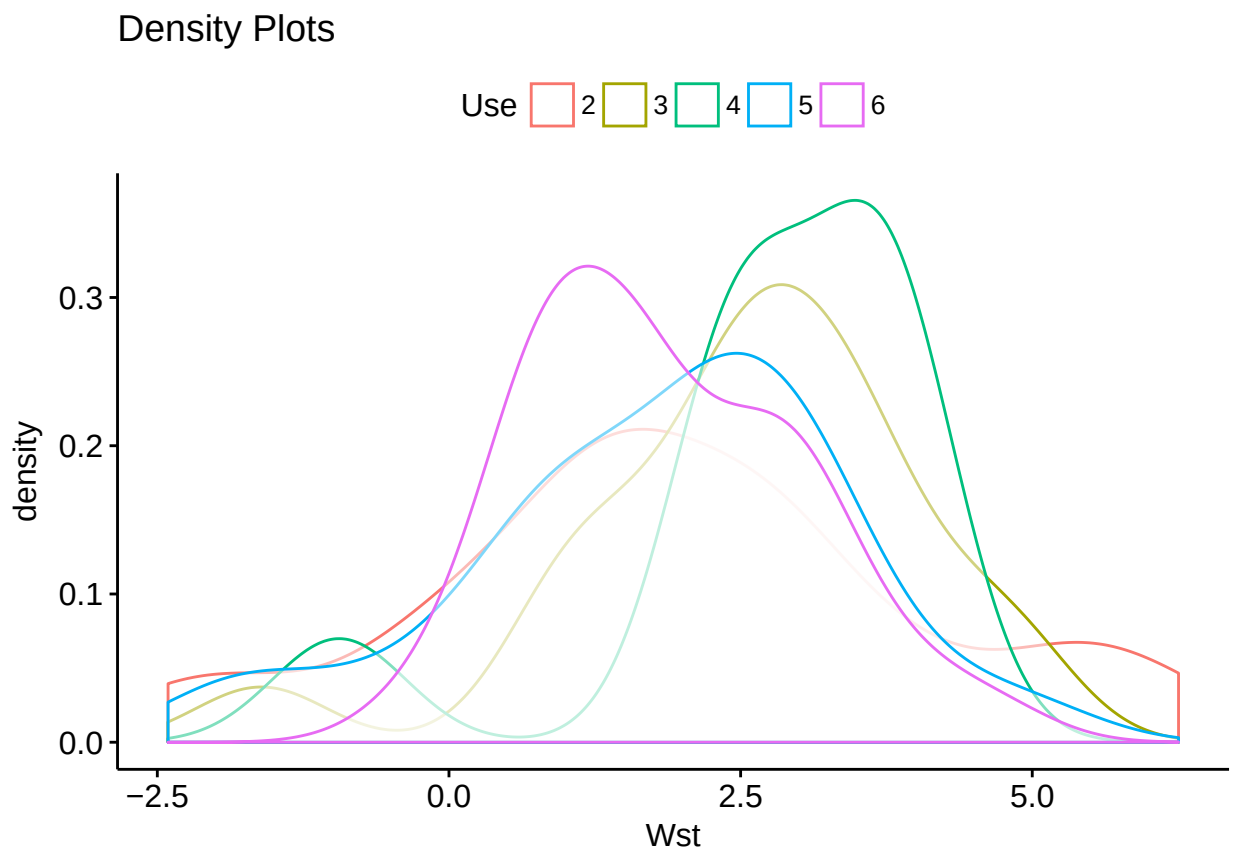
However, my exploration of the data will largely be focused around the interaction of these with the Use variable, which gives us a type of company we are dealing with. It seems likely to me that a restaurant may be able to generate more waste with fewer employees than an office building, as an example. In addition, number of employees may be an excellent predictor for one Use, but not for another.

We have one categorical variable, which is the type of company. We can see looking at it that there's a difference, but that overlap between groups for a given Wst value is also high.

```
ggviolin(train,y="Wst",x="Use",fill="Use",title = "Violin + Boxplot of Wst by Use",add="boxplot")
```

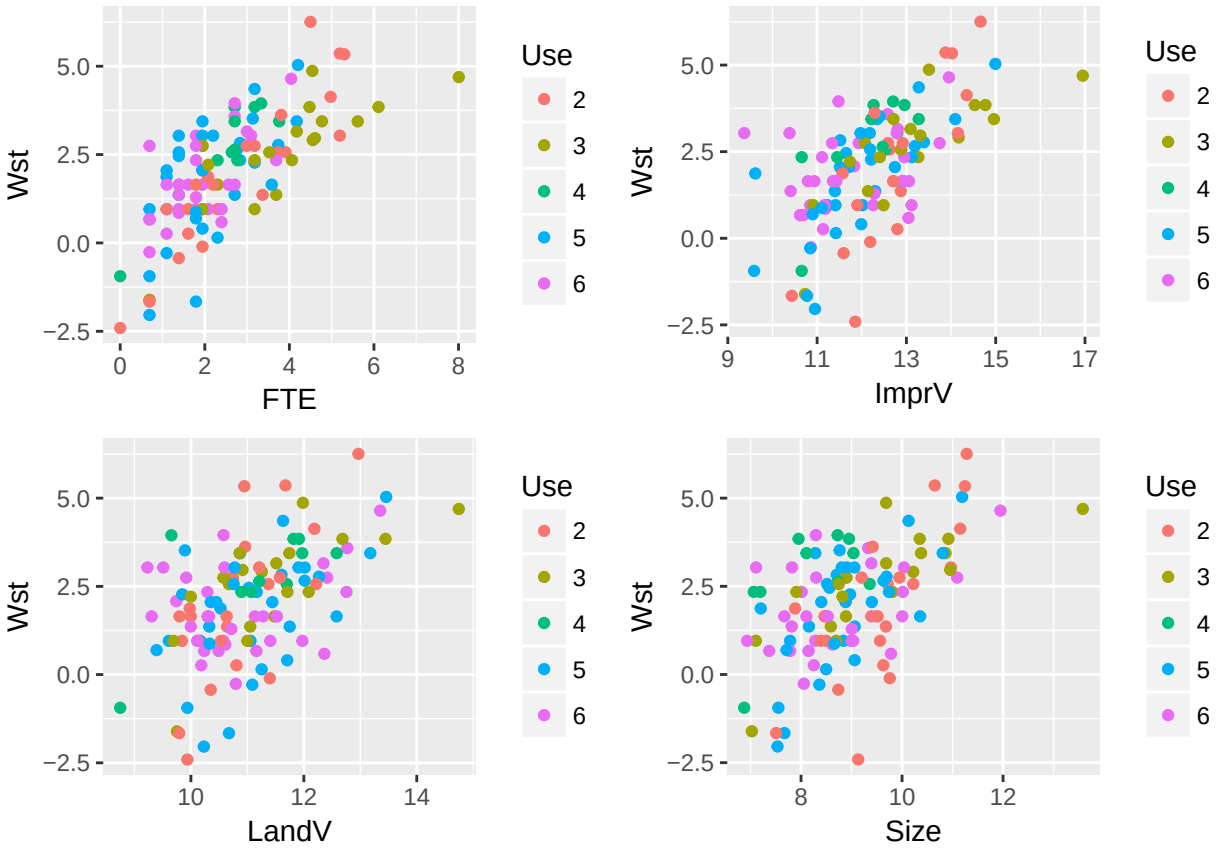



```
ggdensity(train,x="Wst",color = "Use") + ggtitle("Density Plots")
```



For our continuous variables, let's make sure our relationships are linear. We're also going to color by Use, and see if we can notice any differences in how we may need to model different uses separately.

```
library(gridExtra)
plots <- ggpairs(train,mapping=ggplot2::aes(colour = Use))
grid.arrange(getPlot(plots,6,1),
             getPlot(plots,6,2),
             getPlot(plots,6,3),
             getPlot(plots,6,4)
             )
```

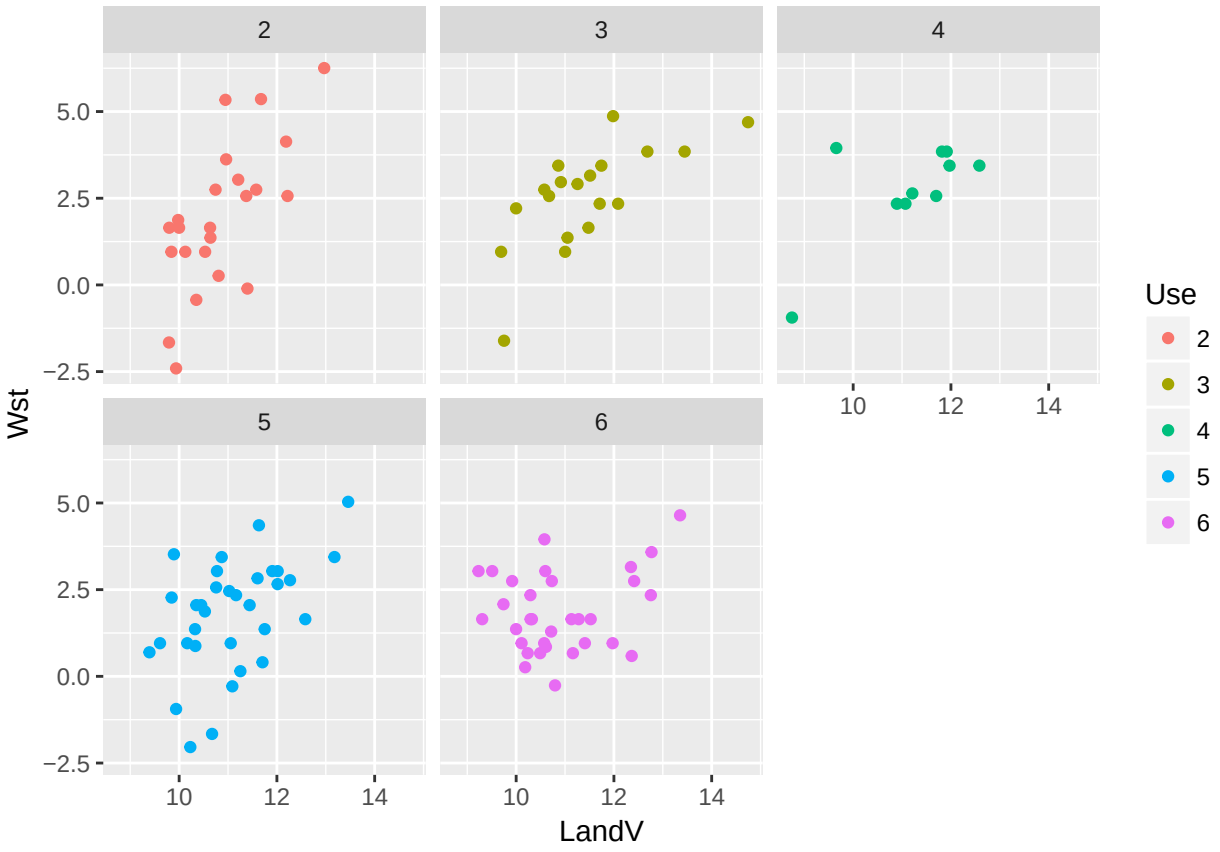


Looking specifically at size, it looks as if some Uses respond differently to changes in Size. For group 5, we see a considerible increase as size increases– this is retail, so it makes sense. For group 6, which is a restaurant or entertainment, changes in size don't seem to create alot of additional waste. For Use 4, it would be a poor predictor.

```
ggplot(train,aes(y=Wst,x=Size,color=Use)) + geom_point() + facet_wrap(~Use) + ggtitle("Change in Wst as
```

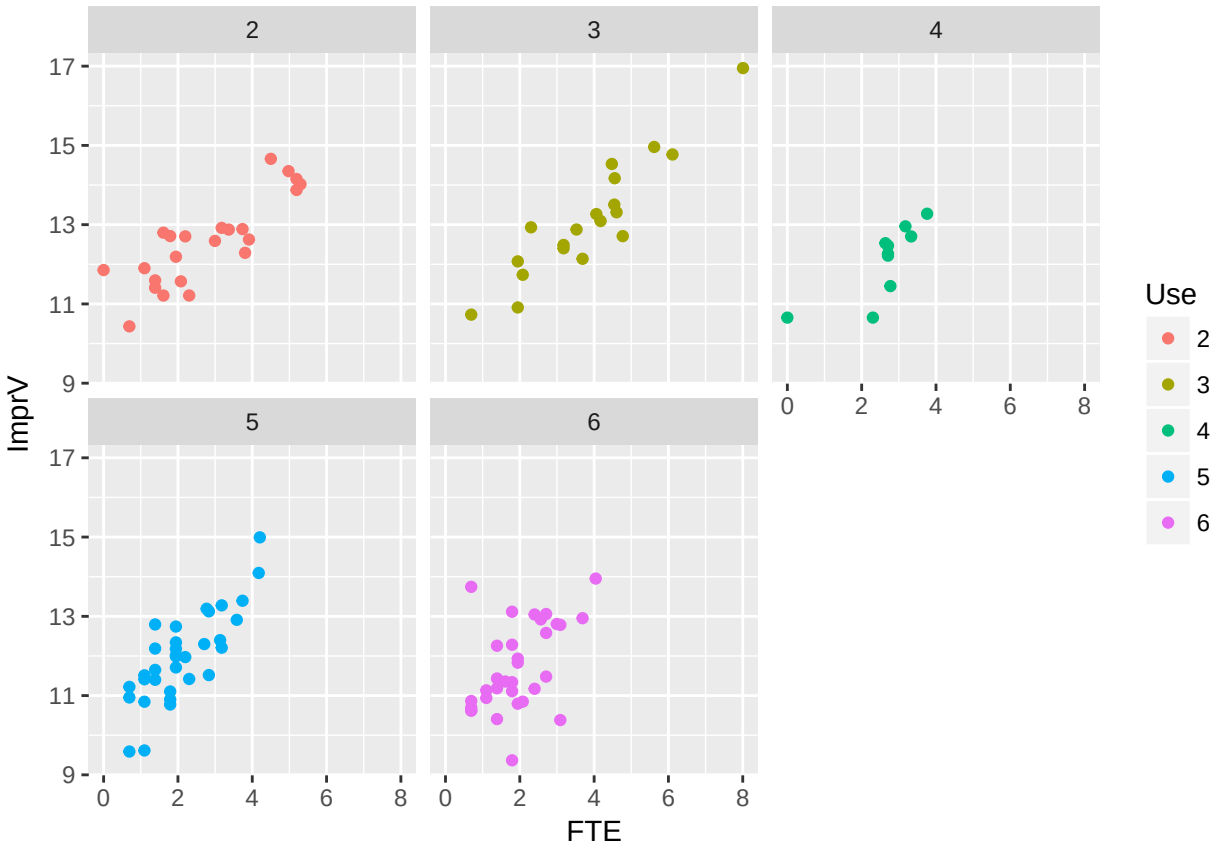
Similarly, if we look at how Wst Changes as LandV changes, we see that Use 2 seems to change very quickly as LandV changes.

```
ggplot(train,aes(y=Wst,x=LandV,color=Use)) + geom_point() + facet_wrap(~Use)
```



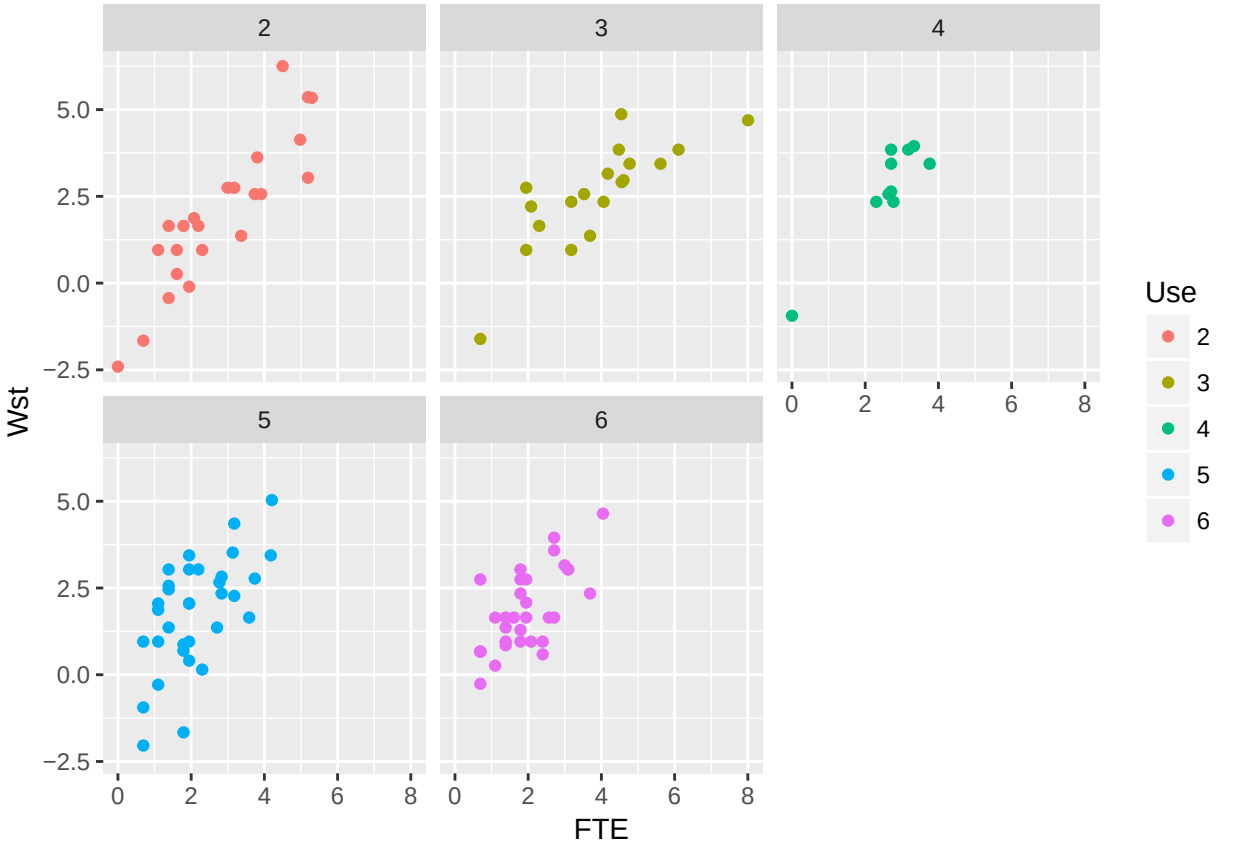
For Land Value increases, we see strong responses from Manufacturing, Warehousing, and Retail .

```
ggplot(train,aes(y=ImprV,x=FTE,color=Use)) + geom_point() + facet_wrap(~Use)
```



For FTE, we see strong responses across the board.

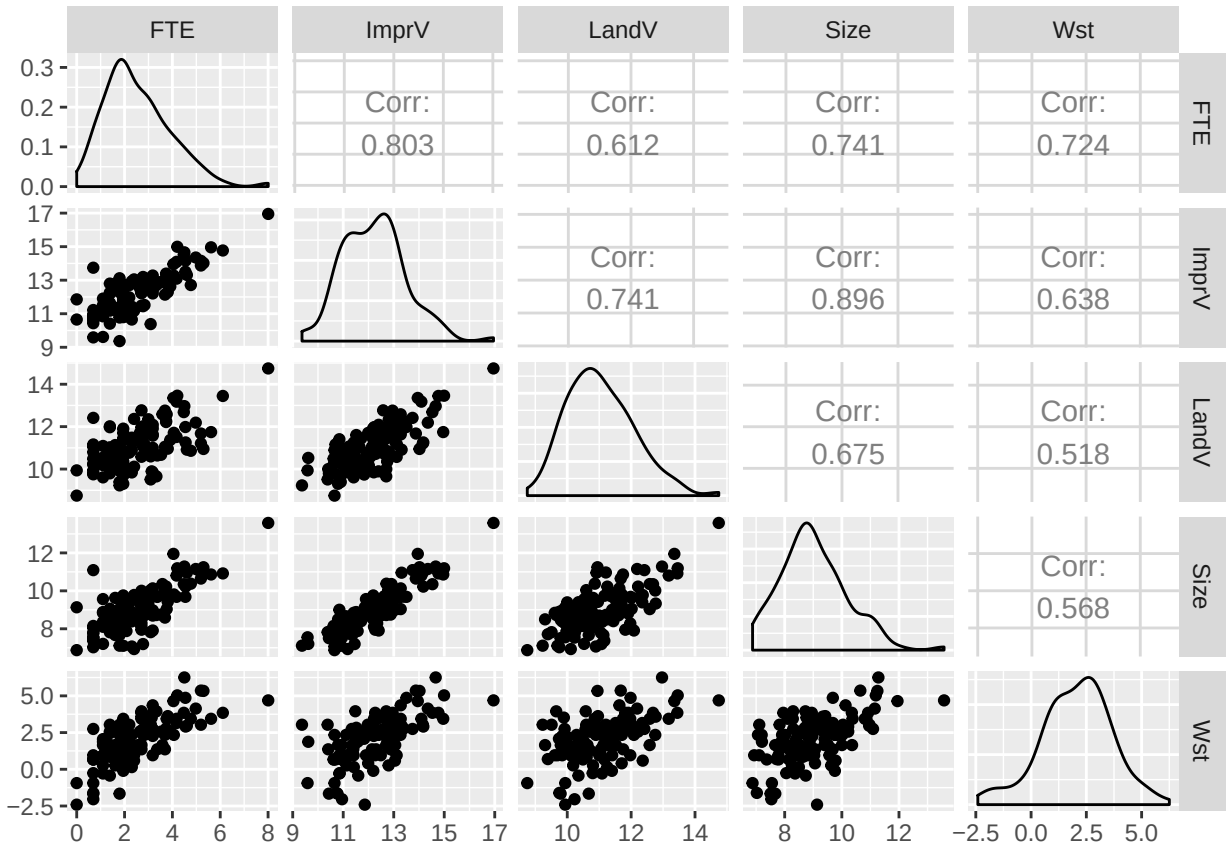
```
ggplot(train,aes(y=Wst,x=FTE,color=Use)) + geom_point() + facet_wrap(~Use)
```



A problem we are going to have is very strong colinearity between our predictors.

But concering is that they appear linear to each other, which could make it challenging to get strong P values.

```
ggpairs(train[,c("FTE","ImprV","LandV","Size","Wst")])
```



I'm going to fit a model with the interactions we're seeing, then extract the model matrix and refit to allow me to backward select the factors level * predictor interactions that are significant and/or informative one at a time, instead of pulling the entire factor out, which is the default behavior for the step function.

```
#Caret Helper Function
fit <- dummyVars(as.formula("Wst ~ Use + Use * (LandV + ImprV + FTE + Size) + Size*FTE + Size*LandV + Si
#Generates the Full Model Matrix
m_matrix <- as.data.frame(predict(fit,newdata=train))
m_matrix$Wst <- train$Wst
refit <- lm(Wst ~.,data=m_matrix)
summary(refit)
```

```
##
## Call:
## lm(formula = Wst ~ ., data = m_matrix)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.73954 -0.57646 -0.00797  0.43623  1.81826
##
## Coefficients: (5 not defined because of singularities)
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -171.04683  134.73949  -1.269  0.20791
## Use.2                   -3.99539    5.88196  -0.679  0.49891
## Use.3                   -5.34497    6.87968  -0.777  0.43947
## Use.4                  -12.23164    9.91464  -1.234  0.22088
## Use.5                   -9.11781    4.26705  -2.137  0.03563 *
```

```
## Use.6                       NA         NA      NA       NA
## LandV                    19.18326   12.65513   1.516  0.13345
## ImprV                     3.77821   12.50344   0.302  0.76329
## FTE                      39.38568   38.28035   1.029  0.30660
## Size                     30.75898   16.73798   1.838  0.06978 .
## `Use.2:LandV`             0.04545    0.60150   0.076  0.93995
## `Use.3:LandV`             0.14949    0.79906   0.187  0.85206
## `Use.4:LandV`            -0.33007    0.63336  -0.521  0.60369
## `Use.5:LandV`            -0.11420    0.37924  -0.301  0.76409
## `Use.6:LandV`                NA         NA      NA       NA
## `Use.2:ImprV`             0.78838    0.84924   0.928  0.35599
## `Use.3:ImprV`             0.03541    1.08615   0.033  0.97407
## `Use.4:ImprV`             2.43582    1.47274   1.654  0.10201
## `Use.5:ImprV`             1.55501    0.58800   2.645  0.00982 **
## `Use.6:ImprV`                NA         NA      NA       NA
## `Use.2:FTE`               0.53144    0.46797   1.136  0.25946
## `Use.3:FTE`              -0.46203    0.53929  -0.857  0.39412
## `Use.4:FTE`              -1.57454    1.02501  -1.536  0.12841
## `Use.5:FTE`              -0.67718    0.46185  -1.466  0.14645
## `Use.6:FTE`                  NA         NA      NA       NA
## `Use.2:Size`             -0.85930    0.93709  -0.917  0.36187
## `Use.3:Size`              0.50389    0.83360   0.604  0.54722
## `Use.4:Size`             -1.04828    1.10605  -0.948  0.34607
## `Use.5:Size`             -0.78147    0.67177  -1.163  0.24813
## `Use.6:Size`                 NA         NA      NA       NA
## `FTE:Size`               -6.93365    4.52190  -1.533  0.12909
## `LandV:Size`             -3.18662    1.54736  -2.059  0.04267 *
## `ImprV:Size`             -1.38254    1.37627  -1.005  0.31810
## `LandV:ImprV`            -0.70540    1.12298  -0.628  0.53167
## `LandV:FTE`              -3.49660    3.21363  -1.088  0.27980
## `ImprV:FTE`              -0.49888    3.05691  -0.163  0.87077
## `LandV:FTE:Size`          0.63669    0.36509   1.744  0.08496 .
## `ImprV:FTE:Size`          0.27472    0.28701   0.957  0.34133
## `LandV:ImprV:Size`        0.16477    0.12050   1.367  0.17528
## `LandV:ImprV:FTE`         0.06567    0.24413   0.269  0.78863
## `LandV:ImprV:FTE:Size`   -0.02780    0.02110  -1.317  0.19139
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9806 on 81 degrees of freedom
## Multiple R-squared:  0.7382, Adjusted R-squared:  0.6251
## F-statistic: 6.526 on 35 and 81 DF,  p-value: 1.889e-12
```

What we essentially have now is a model fit with all the possible interactions of a given Use with each of our 4 continuous predictors.

Now I am going to use backwards stepwise selection to remove the ones that are not good predictors. We have damaged our ability to do statistical inference on this data, but we have a holdout dataset I have not yet looked at to verify the findings of the model.

To avoid violating the marginality principle, I am going to force the lower scope to include the non-interacted terms.

```
min_model <- as.formula("Wst ~ LandV + ImprV + FTE + Size")
backwise_selected_model <- step(refit,scope=list(lower=min_model),direction="backward",trace=F,k=2)
summary(backwise_selected_model)
```

```
##
## Call:
## lm(formula = Wst ~ Use.5 + LandV + ImprV + FTE + Size + `Use.4:ImprV` +
##     `Use.5:ImprV` + `Use.2:FTE` + `Use.2:Size` + `Use.5:Size` +
##     `FTE:Size` + `LandV:Size` + `ImprV:Size` + `LandV:FTE` +
##     `LandV:FTE:Size` + `ImprV:FTE:Size` + `LandV:ImprV:Size` +
##     `LandV:ImprV:FTE:Size`, data = m_matrix)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.81161 -0.58651 -0.07177  0.58597  1.99334
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -83.330231  33.659186  -2.476 0.015013 *
## Use.5                   -4.284220   2.353420  -1.820 0.071747 .
## LandV                    8.624797   3.084943   2.796 0.006231 **
## ImprV                   -1.322525   1.688131  -0.783 0.435266
## FTE                     25.437765  10.436151   2.437 0.016593 *
## Size                    22.039096   8.034292   2.743 0.007237 **
## `Use.4:ImprV`            0.085015   0.030059   2.828 0.005675 **
## `Use.5:ImprV`            1.083776   0.471295   2.300 0.023594 *
## `Use.2:FTE`              0.845467   0.216458   3.906 0.000173 ***
## `Use.2:Size`            -0.284166   0.071495  -3.975 0.000135 ***
## `Use.5:Size`            -0.991216   0.546099  -1.815 0.072569 .
## `FTE:Size`              -4.989768   2.313826  -2.157 0.033491 *
## `LandV:Size`            -2.154879   0.725588  -2.970 0.003747 **
## `ImprV:Size`            -0.888082   0.482456  -1.841 0.068684 .
## `LandV:FTE`             -2.156965   0.956434  -2.255 0.026344 *
## `LandV:FTE:Size`         0.456415   0.208099   2.193 0.030654 *
## `ImprV:FTE:Size`         0.185692   0.096154   1.931 0.056347 .
## `LandV:ImprV:Size`       0.097537   0.035088   2.780 0.006521 **
## `LandV:ImprV:FTE:Size`  -0.018087   0.008421  -2.148 0.034178 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9362 on 98 degrees of freedom
## Multiple R-squared:  0.7113, Adjusted R-squared:  0.6582
## F-statistic: 13.41 on 18 and 98 DF,  p-value: < 2.2e-16
```

In addition, I wanted to try cross-validation using only my training set. I want to find variables that don't predict well when they are cross-validated. I've written my own step code because we're going to cross-validate at each step using MSE, but only within the training set. We will not be looking at the test set yet, but will be breaking the training set into 80/20 groups 5 times each. We will test each predictor 5 times by removing 20% of the data and attempting to predict the remaining 20%. If the variable does not predict well, we will remove it from the data.

```
seeds_to_use <- c(33,34,35,36,37)
diagnostics <- c()
keep_going = TRUE
best_fit <- backwise_selected_model

for (k in 1:100){
  #Extract Terms as it Will Change
  terms <- attr(best_fit$terms,"term.labels")
```

```
  mse_df <- data.frame(Terms=terms)
  for (s in seeds_to_use){
    mse_list <- c()
    set.seed(s)
    for (j in terms){
      #Version of the Model at Current Step
      bc_fit <- best_fit

      #Use a Random Sample of the Data
      bc_idx <- sample(nrow(bc_fit$model), nrow(bc_fit$model)*4/5)
      bc_train <- bc_fit$model[bc_idx,]
      bc_test <- bc_fit$model[bc_idx,]
      #Take off the Jth term
      eval(parse(text=paste("bc_fit <- update(bc_fit, . ~ . -",j,")",sep="")))
      #Fit on Train
      bc_fit <- lm(formula(bc_fit),data=bc_train)
      #Predict on Test
      bc_mse <- mean((predict(bc_fit,newdata = bc_test) - bc_test$Wst)^2)
      #Collect the AICs of Each Removed Regressor
      mse_list <- c(mse_list,bc_mse)
    }
    mse_df[[paste("seed",s,sep="")]] <- mse_list

  }

  mse_df$Number_Better <- ifelse(mse_df$seed33 < mean(best_fit$residuals^2),1,0) +
    ifelse(mse_df$seed34 < mean(best_fit$residuals^2),1,0) +
    ifelse(mse_df$seed35 < mean(best_fit$residuals^2),1,0) +
    ifelse(mse_df$seed36 < mean(best_fit$residuals^2),1,0) +
    ifelse(mse_df$seed37 < mean(best_fit$residuals^2),1,0)

  mse_df$Mean_MSE <- (mse_df$seed33 + mse_df$seed34 + mse_df$seed35 + mse_df$seed36 + mse_df$seed37)/5
  lowest_mse <- min(mse_df$Mean_MSE)
  lowest_term <- mse_df$Terms[match(lowest_mse,mse_df$Mean_MSE)]
  mse_df$Lowest_MSE_Before <- mean(best_fit$residuals^2)

  #If There is an Imporvement by Removing a term.
  if (lowest_mse <= mean(best_fit$residuals^2)){
    print(paste("Removing",lowest_term,"."))
    #Actually Modify the Best Fit Model
    eval(parse(text=paste("best_fit <- update(best_fit, . ~ . -",lowest_term,")",sep="")))
  } else {
    print("Done")
    break
  }
  diagnostics <- c(diagnostics,list(mse_df))
}
```

```
## [1] "Removing ‘LandV:FTE:Size‘ ."
## [1] "Removing ‘ImprV:FTE:Size‘ ."
## [1] "Removing ‘LandV:ImprV:Size‘ ."
## [1] "Removing ‘LandV:ImprV:FTE:Size‘ ."
## [1] "Removing ‘LandV:FTE‘ ."
## [1] "Done"
```

```
best_fit <- update(best_fit,formula. = as.formula("~. + FTE + Size"))
summary(best_fit)
```

```
##
## Call:
## lm(formula = Wst ~ Use.5 + LandV + ImprV + FTE + Size + `Use.4:ImprV` +
##     `Use.5:ImprV` + `Use.2:FTE` + `Use.2:Size` + `Use.5:Size` +
##     `FTE:Size` + `LandV:Size` + `ImprV:Size`, data = m_matrix)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.90992 -0.59676  0.06954  0.61558  1.91557
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   16.70900    8.96763   1.863 0.065275 .
## Use.5         -5.26193    2.24350  -2.345 0.020923 *
## LandV          0.28505    1.07228   0.266 0.790896
## ImprV         -2.42720    1.14482  -2.120 0.036395 *
## FTE            3.20231    0.78760   4.066 9.38e-05 ***
## Size          -1.60526    1.02146  -1.572 0.119125
## `Use.4:ImprV`  0.07579    0.03034   2.498 0.014075 *
## `Use.5:ImprV`  1.25489    0.44012   2.851 0.005262 **
## `Use.2:FTE`    0.73402    0.18587   3.949 0.000144 ***
## `Use.2:Size`  -0.25124    0.06199  -4.053 9.83e-05 ***
## `Use.5:Size`  -1.10623    0.52554  -2.105 0.037728 *
## `FTE:Size`    -0.28745    0.08394  -3.424 0.000886 ***
## `LandV:Size`  -0.02922    0.11710  -0.249 0.803472
## `ImprV:Size`   0.25161    0.12514   2.011 0.046972 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9612 on 103 degrees of freedom
## Multiple R-squared:  0.6801, Adjusted R-squared:  0.6398
## F-statistic: 16.85 on 13 and 103 DF,  p-value: < 2.2e-16
```

Checking Against our Holdout

Refitting our model on test to get accurate p-values.

```
test$ImprV <- log(test$ImprV)
test$FTE <- log(test$FTE)
test$LandV <- log(test$LandV)
test$Size <- log(test$Size)
test$Wst <- log(test$Wst)

#Generates the Full Model Matrix
test_matrix <- as.data.frame(predict(fit,newdata=test))
test_matrix$Wst <- test$Wst

test_fit_simple <- lm(formula(simple_model),data=test)
test_fit_complex <- lm(formula(best_fit),data=test_matrix)
```

Simple Model

```
mean(test_fit_simple$residuals^2)
```

```
## [1] 0.8708202
```

```
summary(test_fit_simple)
```

```
##
## Call:
## lm(formula = formula(simple_model), data = test)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.06369 -0.54116  0.03441  0.50214  2.14844
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.0528     1.1072  -0.951    0.351
## Use3          0.4354     1.1099   0.392    0.698
## Use4          1.9676     1.1729   1.678    0.106
## Use5          1.5071     1.1358   1.327    0.197
## Use6          1.2965     1.0901   1.189    0.246
## FTE           0.8375     0.1546   5.417 1.45e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.043 on 24 degrees of freedom
## Multiple R-squared:  0.625,  Adjusted R-squared:  0.5468
## F-statistic: 7.999 on 5 and 24 DF,  p-value: 0.0001485
```

```
saveRDS(test_fit_simple,"simple.rds")
```

```
mean(test_fit_complex$residuals^2)
```

```
## [1] 0.5399147
```

```
summary(test_fit_complex)
```

```
##
## Call:
## lm(formula = formula(best_fit), data = test_matrix)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9140 -0.3834  0.1181  0.4015  1.4852
##
## Coefficients: (1 not defined because of singularities)
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -21.71266   32.89435  -0.660   0.5181
## Use.5          -2.76255    5.10295  -0.541   0.5953
## LandV          -2.62975    3.55104  -0.741   0.4691
## ImprV           4.60997    4.41544   1.044   0.3111
## FTE            -2.32668    3.89992  -0.597   0.5586
## Size            2.12108    3.92763   0.540   0.5962
## `Use.4:ImprV`   0.10198    0.05325   1.915   0.0725 .
## `Use.5:ImprV`   1.05176    0.82531   1.274   0.2197
```

```
## `Use.2:FTE`    -0.38357    0.43275  -0.886   0.3878
## `Use.2:Size`         NA         NA      NA       NA
## `Use.5:Size`   -1.11692    1.02166  -1.093   0.2895
## `FTE:Size`      0.28531    0.46245   0.617   0.5454
## `LandV:Size`    0.33615    0.41008   0.820   0.4237
## `ImprV:Size`   -0.51414    0.50986  -1.008   0.3274
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9761 on 17 degrees of freedom
## Multiple R-squared:  0.7675, Adjusted R-squared:  0.6034
## F-statistic: 4.676 on 12 and 17 DF,  p-value: 0.002087
```

```
saveRDS(test_fit_complex,"complex.rds")
```

10.2.1

```
library(alr4)
```

```
## Loading required package: effects
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'carData'
```

```
## The following objects are masked from 'package:car':
##
##     Guyer, UN, Vocab
```

```
## Use the command
##     lattice::trellis.par.set(effectsTheme())
##   to customize lattice options for effects plots.
## See ?efffectTheme for details.
```

```
full_model <- lm(log(rate) ~ log(sigs1) + log(trks) + log(adt) + log(len) + lane + slim + shld + lwid +

#AIC Backward
summary(step(full_model,trace=F,direction="backward"))
```

```
##
## Call:
## lm(formula = log(rate) ~ log(sigs1) + log(adt) + log(len) + slim +
##     htype, data = Highway1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.56334 -0.12361 -0.02101  0.09588  0.45868
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.474546   0.684395   6.538 2.68e-07 ***
## log(sigs1)   0.208357   0.058412   3.567  0.00120 **
## log(adt)    -0.126910   0.082871  -1.531  0.13581
## log(len)    -0.261607   0.082063  -3.188  0.00327 **
## slim        -0.029736   0.009681  -3.072  0.00441 **
```

```
## htypeMA     -0.266488   0.253175  -1.053  0.30067
## htypeMC     -0.123807   0.336375  -0.368  0.71533
## htypePA     -0.495426   0.198670  -2.494  0.01819 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.243 on 31 degrees of freedom
## Multiple R-squared:  0.7753, Adjusted R-squared:  0.7245
## F-statistic: 15.28 on 7 and 31 DF,  p-value: 1.835e-08
```

```
#AIC Forward
summary(step(full_model,trace=F,direction="forward"))
```

```
##
## Call:
## lm(formula = log(rate) ~ log(sigs1) + log(trks) + log(adt) +
##     log(len) + lane + slim + shld + lwid + acpt + itg + htype,
##     data = Highway1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44802 -0.10192 -0.00692  0.12231  0.42116
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.191699   1.818482   2.305   0.0297 *
## log(sigs1)   0.192322   0.075367   2.552   0.0172 *
## log(trks)   -0.197560   0.239812  -0.824   0.4178
## log(adt)    -0.154625   0.111893  -1.382   0.1792
## log(len)    -0.214470   0.099986  -2.145   0.0419 *
## lane        -0.011133   0.057021  -0.195   0.8468
## slim        -0.027260   0.016799  -1.623   0.1172
## shld         0.002974   0.034159   0.087   0.9313
## lwid         0.042122   0.136821   0.308   0.7607
## acpt         0.006049   0.008101   0.747   0.4622
## itg          0.035722   0.242818   0.147   0.8842
## htypeMA     -0.381275   0.357472  -1.067   0.2964
## htypeMC     -0.237545   0.399822  -0.594   0.5578
## htypePA     -0.523326   0.290041  -1.804   0.0832 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2607 on 25 degrees of freedom
## Multiple R-squared:  0.7913, Adjusted R-squared:  0.6828
## F-statistic: 7.293 on 13 and 25 DF,  p-value: 1.247e-05
```

```
#BIC Backward
summary(step(full_model,trace=F,direction="backward",k=log(n)))
```

```
##
## Call:
## lm(formula = log(rate) ~ log(sigs1) + log(adt) + log(len) + acpt +
##     itg, data = Highway1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -0.52501 -0.16869 -0.03491  0.19708  0.57392
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.388826   0.334441   7.143 3.49e-08 ***
## log(sigs1)   0.188719   0.053034   3.558  0.00116 **
## log(adt)    -0.246023   0.077136  -3.189  0.00312 **
## log(len)    -0.247880   0.090854  -2.728  0.01012 *
## acpt         0.018412   0.005543   3.322  0.00220 **
## itg          0.424597   0.178027   2.385  0.02298 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2664 on 33 degrees of freedom
## Multiple R-squared:  0.7125, Adjusted R-squared:  0.669
## F-statistic: 16.36 on 5 and 33 DF,  p-value: 4.082e-08
```

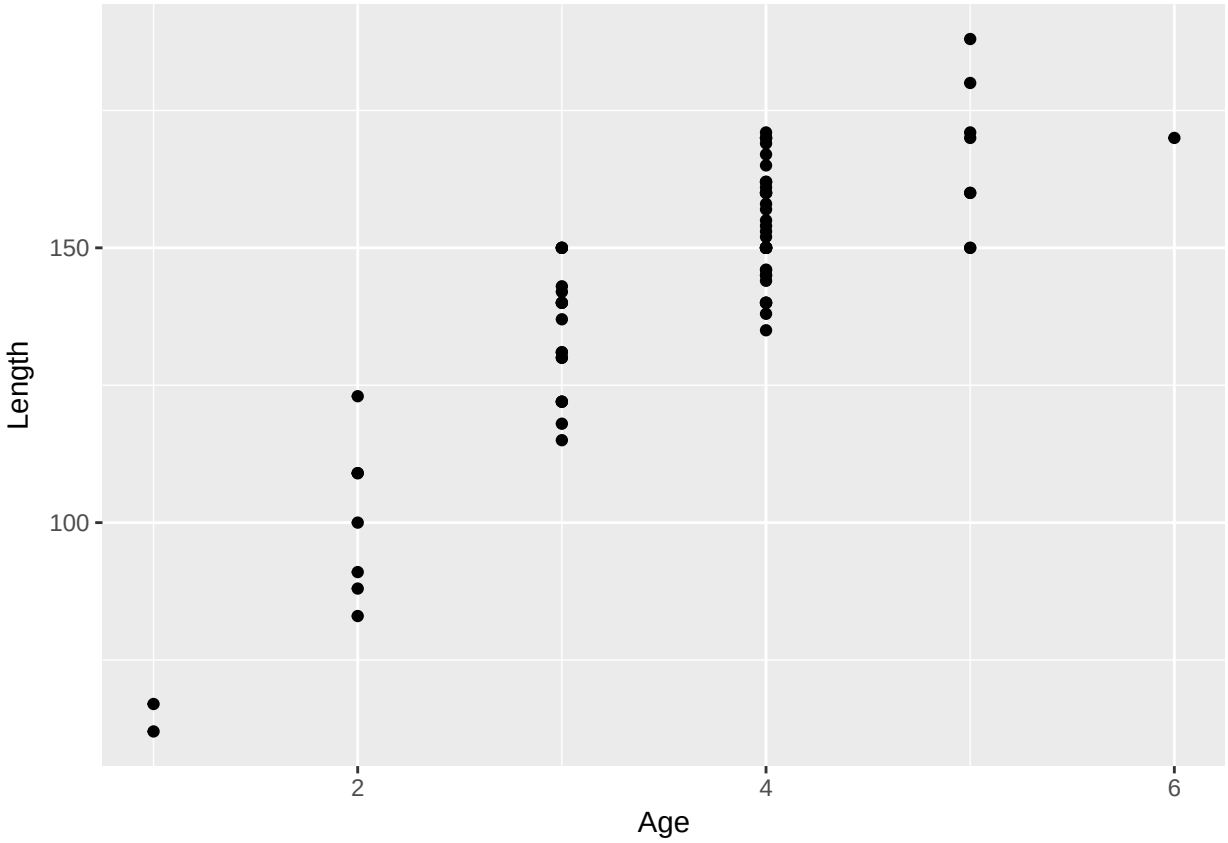
```r
#BIC Forward
summary(step(full_model,trace=F,direction="forward",k=log(n)))
```

```
##
## Call:
## lm(formula = log(rate) ~ log(sigs1) + log(trks) + log(adt) +
##     log(len) + lane + slim + shld + lwid + acpt + itg + htype,
##     data = Highway1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44802 -0.10192 -0.00692  0.12231  0.42116
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.191699   1.818482   2.305   0.0297 *
## log(sigs1)   0.192322   0.075367   2.552   0.0172 *
## log(trks)   -0.197560   0.239812  -0.824   0.4178
## log(adt)    -0.154625   0.111893  -1.382   0.1792
## log(len)    -0.214470   0.099986  -2.145   0.0419 *
## lane        -0.011133   0.057021  -0.195   0.8468
## slim        -0.027260   0.016799  -1.623   0.1172
## shld         0.002974   0.034159   0.087   0.9313
## lwid         0.042122   0.136821   0.308   0.7607
## acpt         0.006049   0.008101   0.747   0.4622
## itg          0.035722   0.242818   0.147   0.8842
## htypeMA     -0.381275   0.357472  -1.067   0.2964
## htypeMC     -0.237545   0.399822  -0.594   0.5578
## htypePA     -0.523326   0.290041  -1.804   0.0832 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2607 on 25 degrees of freedom
## Multiple R-squared:  0.7913, Adjusted R-squared:  0.6828
## F-statistic: 7.293 on 13 and 25 DF,  p-value: 1.247e-05
```

11.2

11.2.1

```
ggplot(lakemary,aes(y=Length,x=Age)) + geom_point()
```



11.2.2

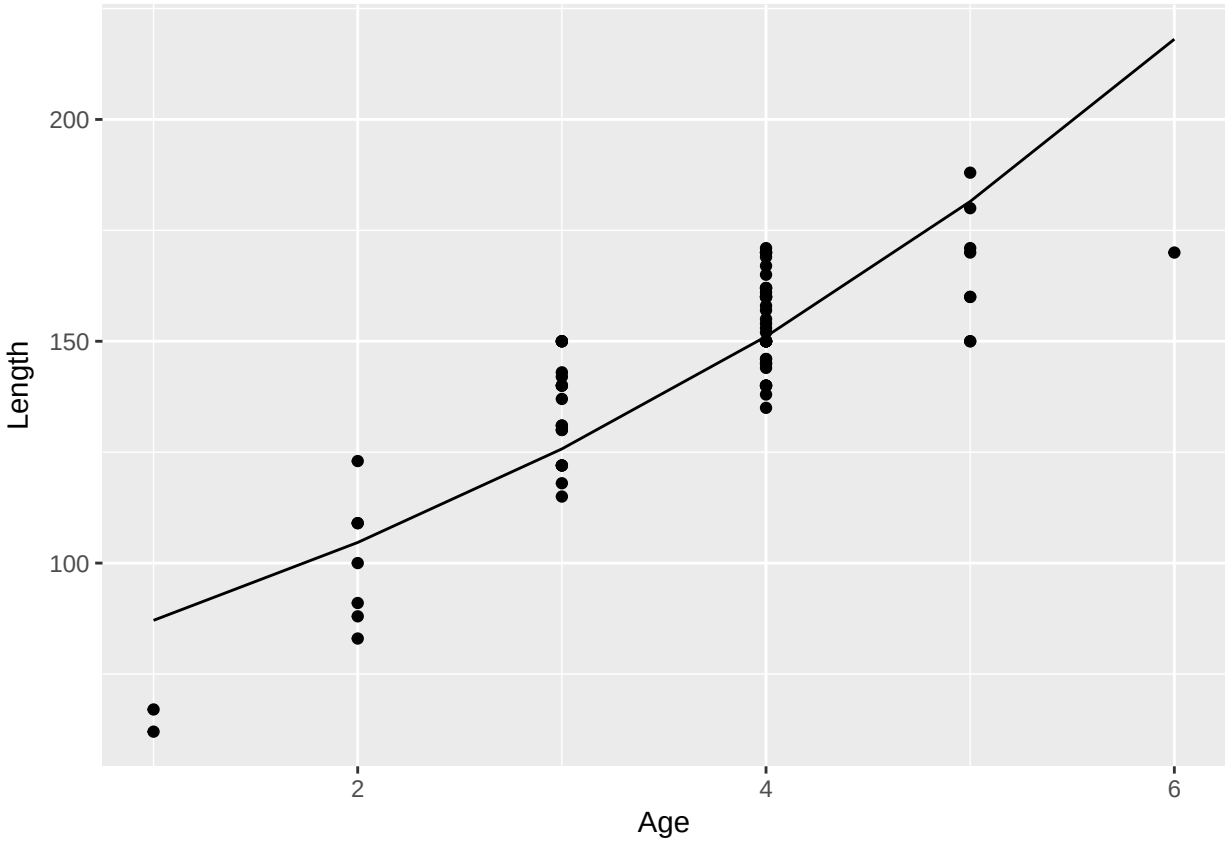
200 will be our initial value of L.

So $\frac{E(Length|Age)}{200} = 1 - exp(-K(t - t_0))$

If we log both sides, we should get

$$log(\frac{E(Length|Age)}{200}) = 0 + K(t - t_0)$$

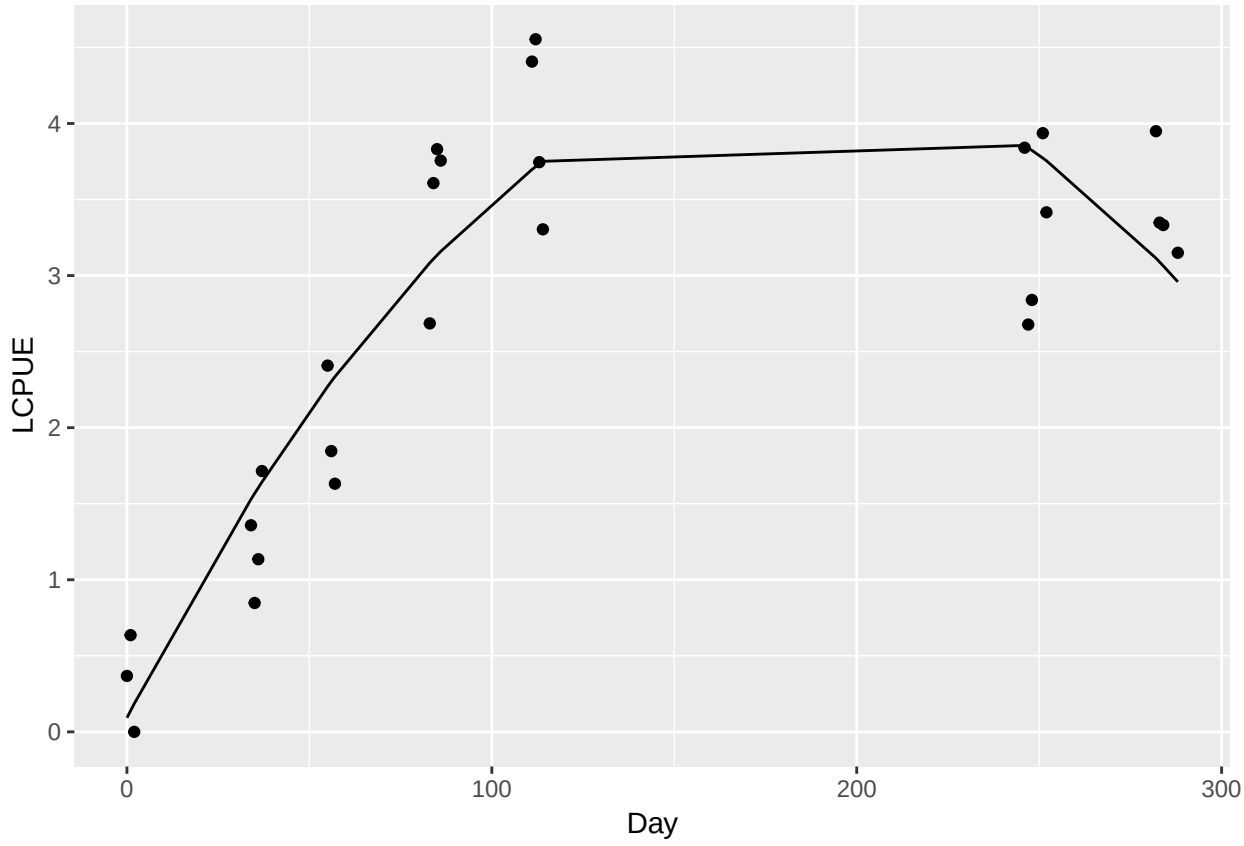
```
model <- lm(log(Length/200) ~ Age,data=lakemary)
lakemary$Predicted <- predict(model)
lakemary$Predicted <- exp(lakemary$Predicted)*200
ggplot(lakemary,aes(y=Length,x=Age)) + geom_point() + geom_line(aes(y=Predicted,x=Age))
```



11.4.1

```
swan96$DaySquare <- swan96$Day^2
quad_f <- as.formula("LCPUE ~ Day + DaySquare")
linear_fit <- lm(quad_f,data=swan96)
swan96$Predicted <- predict(linear_fit)

ggplot(swan96,aes(y=LCPUE,x=Day)) + geom_point() + geom_line(aes(y=Predicted,x=Day))
```



11.4.2

```
deltaMethod(linear_fit,g="Day + Day^2")
```

```
##                Estimate          SE      2.5 %     97.5 %
## Day + Day^2 0.04877729 0.005684939 0.03763502 0.05991957
```