Part A Report

Introduction:
The objective of this project is to take in two raw datasets, process it, and correctly merging them, ensuring that data is not mismatched. One of the additional challenges were to deal with missing data and then provide a basic understanding of the resulting merged and imputed dataset.
Methods:
Using the language R in a jupyter notebook, I was able to extract the appropriate dataset tasked to me using read.cv(). From there, since the datasets need to be merged based off their ID's, I checked to make sure that both datasets contained only unique IDs. I did this by using the unique(), and making sure that the length returned was the same as the length of the dataset. From there I merged the datasets, merging them on the "ID" column using the merge() function in R. Next is checking for missing values. To do this I used is.na(), is.nan(), is.null(), to make sure that any type of missing value is account for. Next to get a better understanding of missing values, I checked the amount of na's present only in DV, only in IV, and only in both. To verify my findings, I also used the MICE package to show missing values. Furthermore, I then used the MICE package to impute the missing values. My choice for method of imputation was to use linear regression, the bootstrap method (norm.boot). After imputing the data the total number of rows in the dataset became 634. Moving on, I calculated some basic statistics found in boxplots using basic R functions, and also constructing a histogram of the data. Then I created a linear regression model and found the bivariate statistics using basic R functions again. Using the knitir package, I was able to generate an ANOVA table for the model, plot the model, plot the residuals, and calculate the confidence intervals at various levels.
Results:
Using the methods above, here are the results I have obtained. Total number of rows before data cleaning was 641 before and after merge. Total number of nas in both IV and DV columns were 7, number of rows with only was 69, and only IV was 81. Total number of rows after imputation was 634. The results of the ANOVA table yielded an F value of 4861.673, a high F value. The slope and y intercept for the fitted regression equation is as follows: b1 = 8.181, and b0 = 50.542. Our multiple r^2 value was 0.885, meaning that 88% of the variance of the dependent variable is explained by the independent variable. Also, the confidence interval for the slope was (7.950484, 8.41129) and 95% and (7.877751, 8.484023) at 99% confidence. Both exclude 0, meaning that the null hypothesis that the slope is 0 is soundly rejected. Lastly, a graph of the residuals showed no apparent pattern. Additional statistics: Min: 69.75385, First Quartile: 85.45680, Median: 90.91380, Third Quartile: 97.20179, Maximum: 114.04317. Histogram revealed that values between 4 through 6 are the most common for the IV.
Conclusion:
Given the dataset only had 7 missing values in both IV and DV, I was able to impute the data well. Furthermore, the linear regression model created showed that there was a strong association between the IV and the DV, with an multiple r^2 value being 0.885. The model also yielded a slope of b1 = 8.181 and b0 = 50.542. The residuals had no pattern, and the null hypothesis for the slope being 0 was rejected, resulting in a strong linear regression model.

Part B Report

Introduction:
The problem being tasked is trying to find the function used to generate the dependent variable value based on the value of the independent variable. We do this by trying to find a good regression model after transforming our data.
Methods:
We start by using the basic R function read.csv(). After extracting the data, I checked to see how many rows and columns, and how many missing values there are. There were 449 total rows and 0 missing values. Immediately after I created the linear regression model using the lm() R function. Upon observing the current r^2 value without an transformations, I looked at the regression plot, to observe the behaviour of the data points. The IV and DV were positively correlated but it seemed like the variance increased exponentially as IV increased. I checked the residual plot and the same pattern matched. Thus, I began experimenting with transformations by applying them to the IV and/or DV, creating a linear regression model, and then looking at r^2 value in the summary statistics as a measure to find the best transformation. Upon deciding on a transformation, I selected that model for binning and a lack of fit test, using the remotes package and cran/alr3 package. Before the lack of fit test, I binned the data to rid it of nearly repeated data. Then I remade the model after applying the bin, and observed the ANOVA table.
Results:
The dataset consisted of a total of 449 rows. There were no missing values. An initial observation of the regression model showed a multiple r^2 value of 0.4018. B1 = 258.64 and B0 = -619.61. The residual plot showed a pattern of increasing variance exponentially as IV (x) increased (but more to the positive side). Upon testing multiple transformations, I found that a transformation on y = e^(0.1*ln(y)) (in R it is exp(0.1)^log(partBTest6$y)) and x = e^(2*ln(x)) (in R it is exp(2)^log(partBTest6$x)). This transformation yielded a multiple r^2 value of 0.5102. Observing the residual plot of the new model, it showed that there was no remaining pattern. Because of the transformation, the values of y became much smaller, so when selecting the size of the bin, I decided on using a 0.05 interval. Then after performing the lack of fit, the lack of fit p-value resulted in 0.7703236. This high p-value indicates that there is no significant lack of fit, meaning that the final model I selected fits the data well. The confidence interval for the slope at 95% was (0.02829126, 0.03396155) and 99% was (0.02739455, 0.03485826). Both excluding 0, so we can reject the null hypothesis, meaning that we can confidently say the slope is not 0. The F statistic was 448.3577139. The new b0 = 1.340884 and b1 = 0.031126.
Conclusion:
Given a clean dataset with an initial multiple r^2 of 0.4018, I was able to find a transformation to improve the r^2 value to 0.5102, meaning that 51% of the variance is explained by the independent variable x. I performed a lack of fit test after binning the data to rid it of near-repeated value. The lack of fit test indicated that the model was a good fit. Testing the slope of the regression model I decided on, the 99% confidence interval excluded 0, meaning that the null hypothesis claiming that the slope is 0 can be soundly rejected.

```r
getwd()
```

```
[1] "/Users/saatvik"
```

```r
wdir <- "/Users/saatvik/Documents/AMS315Project1WD/Data/378631"
```

```r
getwd()
```

```
[1] "/Users/saatvik"
```

```r
setwd(wdir)
getwd()
```

```
[1] "/Users/saatvik/Documents/AMS315Project1WD/Data/378631"
```

```r
partA_IV <- read.csv("378631_IV.csv", header = TRUE)
partA_IV
```

	ID	IV
1	478	4.958839
2	444	4.462000
3	500	3.864393
4	293	2.828281
5	392	5.988746
6	357	5.064720
7	511	3.742016
8	381	4.525937
9	177	5.830791
10	216	5.362566
11	345	4.145341
12	395	6.293983
13	307	3.967386
14	435	3.401253
15	420	NA
16	49	4.797480
17	298	NA
18	636	5.882720
19	302	5.464069
20	472	6.116958
21	331	4.932155
22	327	4.369092
23	532	5.422489
24	626	5.180817
25	317	5.594304
26	476	4.523304
27	388	5.335175
28	180	3.288556
29	73	6.093512
30	15	4.071437
⋮	⋮	⋮
612	25	5.699614

```
613 518 5.292556
614 413 5.165470
615 445 3.851572
616 279       NA
617 367 6.199381
618 179       NA
619  38       NA
620 356 3.434099
621 390 6.043801
622 495       NA
623 281 5.264373
624 370 4.799091
625  98 3.816197
626 166 4.930279
627  41 5.993219
628 571 4.564770
629 284 4.726344
630  55       NA
631 212 6.097972
632 143 4.719072
633 418 5.087994
634 155 3.483674
635 559 4.152491
636 324 3.765104
637 382 5.927676
638 355       NA
639 412 4.594120
640 137 5.441122
641 312 6.480558
```

```r
partA_DV <- read.csv("378631_DV.csv", header = TRUE)
partA_DV
```

	ID	DV
1	123	94.76442
2	217	74.29114
3	246	88.81691
4	485	95.00834
5	487	91.32801
6	489	93.33611
7	529	90.18231
8	136	104.48009
9	255	88.02639
10	348	98.03509
11	192	93.41010
12	234	106.24397
13	53	104.65715
14	543	83.32863
15	604	84.25496
16	452	89.18605

```
17  284  95.41087
18  632  88.40596
19  470 114.04317
20  526  91.98172
21    7 103.53182
22  310        NA
23   40  95.96709
24  600        NA
25  512  86.25134
26  458  86.83932
27  492        NA
28  272  76.98465
29   52  88.30690
30  528  96.93394
⋮   ⋮   ⋮
612  20 104.40881
613 158  83.39205
614 106        NA
615 134  95.29368
616 525  75.10063
617  89  94.62706
618 583  91.12693
619  79  90.11597
620 613  79.55434
621 307  81.06783
622 602  80.25929
623  45 102.60113
624 397        NA
625 507  84.51045
626 359  71.13255
627 421  84.62355
628 465  85.54076
629 226  79.08505
630 438 101.94549
631 347 101.56291
632 296  86.87314
633 542        NA
634 259  83.60478
635  98  82.42091
636 235  78.81061
637 126  89.68589
638 475        NA
639 402 111.21483
640 552        NA
641 264  89.81081
```

```
#If df prints num of rows same as length(unique), then all IDs are
unique, which is good.
partA_IV
length(unique(partA_IV$ID))
```

	ID	IV
1	478	4.958839
2	444	4.462000
3	500	3.864393
4	293	2.828281
5	392	5.988746
6	357	5.064720
7	511	3.742016
8	381	4.525937
9	177	5.830791
10	216	5.362566
11	345	4.145341
12	395	6.293983
13	307	3.967386
14	435	3.401253
15	420	NA
16	49	4.797480
17	298	NA
18	636	5.882720
19	302	5.464069
20	472	6.116958
21	331	4.932155
22	327	4.369092
23	532	5.422489
24	626	5.180817
25	317	5.594304
26	476	4.523304
27	388	5.335175
28	180	3.288556
29	73	6.093512
30	15	4.071437
⋮	⋮	⋮
612	25	5.699614
613	518	5.292556
614	413	5.165470
615	445	3.851572
616	279	NA
617	367	6.199381
618	179	NA
619	38	NA
620	356	3.434099
621	390	6.043801
622	495	NA
623	281	5.264373
624	370	4.799091
625	98	3.816197
626	166	4.930279
627	41	5.993219
628	571	4.564770
629	284	4.726344

```
630  55       NA
631 212 6.097972
632 143 4.719072
633 418 5.087994
634 155 3.483674
635 559 4.152491
636 324 3.765104
637 382 5.927676
638 355       NA
639 412 4.594120
640 137 5.441122
641 312 6.480558
```

```
[1] 641
```

```
#If df prints num of rows same as length(unique), then all IDs are
unique, which is good.
partA_DV
length(unique(partA_DV$ID))
```

```
    ID  DV
1   123  94.76442
2   217  74.29114
3   246  88.81691
4   485  95.00834
5   487  91.32801
6   489  93.33611
7   529  90.18231
8   136 104.48009
9   255  88.02639
10  348  98.03509
11  192  93.41010
12  234 106.24397
13   53 104.65715
14  543  83.32863
15  604  84.25496
16  452  89.18605
17  284  95.41087
18  632  88.40596
19  470 114.04317
20  526  91.98172
21    7 103.53182
22  310        NA
23   40  95.96709
24  600        NA
25  512  86.25134
26  458  86.83932
27  492        NA
28  272  76.98465
29   52  88.30690
```

```
30  528  96.93394
⋮   ⋮   ⋮
612  20 104.40881
613 158  83.39205
614 106        NA
615 134  95.29368
616 525  75.10063
617  89  94.62706
618 583  91.12693
619  79  90.11597
620 613  79.55434
621 307  81.06783
622 602  80.25929
623  45 102.60113
624 397        NA
625 507  84.51045
626 359  71.13255
627 421  84.62355
628 465  85.54076
629 226  79.08505
630 438 101.94549
631 347 101.56291
632 296  86.87314
633 542        NA
634 259  83.60478
635  98  82.42091
636 235  78.81061
637 126  89.68589
638 475        NA
639 402 111.21483
640 552        NA
641 264  89.81081

[1] 641
```

```
partA <- merge(partA_IV, partA_DV, by = "ID")

#If df prints num of rows same as length(unique), then all IDs are
unique, which is good.
partA
length(unique(partA$ID))
```

```
    ID  IV       DV
1    1  3.004821  75.88475
2    2  5.510845  95.51800
3    3  3.492998  78.42695
4    4  5.483592  96.88385
5    5  5.169892        NA
6    6  4.235609  87.29528
7    7  6.280710 103.53182
```

8	8	5.185404	89.23098
9	9	5.719443	NA
10	10	4.034938	85.06978
11	11	4.917672	94.50610
12	12	3.464703	80.30297
13	13	6.339799	100.78565
14	14	6.386308	102.24648
15	15	4.071437	83.38679
16	16	3.891094	81.24049
17	17	NA	84.97899
18	18	4.750984	86.73947
19	19	3.787786	83.45217
20	20	6.213811	104.40881
21	21	5.269975	91.41682
22	22	4.762756	87.90819
23	23	NA	100.44489
24	24	4.004267	NA
25	25	5.699614	97.85908
26	26	NA	88.46603
27	27	4.781962	91.30148
28	28	5.796401	97.97070
29	29	3.758305	80.11396
30	30	4.456300	87.85744
⋮	⋮	⋮	⋮
612	612	4.838639	83.29365
613	613	3.804236	79.55434
614	614	5.176290	90.87047
615	615	4.104794	81.31605
616	616	4.532052	88.98090
617	617	5.813414	93.40338
618	618	5.837407	98.04301
619	619	NA	97.39008
620	620	4.809159	93.50235
621	621	3.417515	NA
622	622	6.633071	104.45610
623	623	5.438367	94.37462
624	624	6.708384	100.00226
625	625	4.771578	91.52980
626	626	5.180817	92.22548
627	627	6.776982	105.10097
628	628	4.467790	87.26828
629	629	4.378895	88.58807
630	630	5.600912	98.64919
631	631	4.310808	84.87648
632	632	4.330794	88.40596
633	633	6.314961	NA
634	634	4.277221	85.77395
635	635	5.715283	101.13068
636	636	5.882720	100.12748

```
637 637 4.474357  85.10011
638 638 4.303195  86.98506
639 639 3.985306  80.38932
640 640 5.831264  99.24615
641 641 4.521000  89.04763
```

```
[1] 641
```

```r
str(partA)
```

```
'data.frame':   641 obs. of  3 variables:
 $ ID: int  1 2 3 4 5 6 7 8 9 10 ...
 $ IV: num  3 5.51 3.49 5.48 5.17 ...
 $ DV: num  75.9 95.5 78.4 96.9 NA ...
```

```r
any(is.na(partA[,1]))
any(is.nan(partA[,1]))
any(is.null(partA[,1]))
```

```
[1] FALSE
```

```
[1] FALSE
```

```
[1] FALSE
```

```r
any(is.na(partA[,2]))
any(is.nan(partA[,2]))
any(is.null(partA[,2]))
```

```
[1] TRUE
```

```
[1] FALSE
```

```
[1] FALSE
```

```r
any(is.na(partA[,3]))
any(is.nan(partA[,3]))
any(is.null(partA[,3]))
```

```
[1] TRUE
```

```
[1] FALSE
```

```
[1] FALSE
```

```r
sum(is.na(partA_IV$IV))
sum(is.na(partA_DV$DV))
```

```
[1] 88
```

```
[1] 76
```

```r
sum(is.na(partA$IV) & is.na(partA$DV)) #IV and DV are both na
sum(!(is.na(partA$IV)) & is.na(partA$DV)) #Means DV is na and IV is
not
sum(is.na(partA$IV) & (!is.na(partA$DV))) #Means IV is na and DV is
not
```

```
[1] 7
```

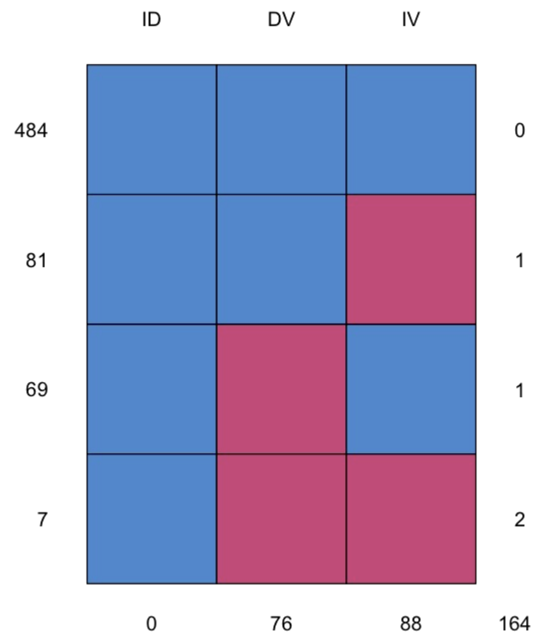
```
[1] 69
```

```
[1] 81
```

```r
#install.packages('mice')

partAFix <- partA
library(mice)
md.pattern(partAFix)
```

```

Attaching package: ‘mice’


The following object is masked from ‘package:stats’:

    filter


The following objects are masked from ‘package:base’:

    cbind, rbind
```

```
    ID DV IV    
484 1   1  1   0
81  1   1  0   1
69  1   0  1   1
7   1   0  0   2
    0  76 88 164
```




```
partANoNA <- partAFix[!is.na(partAFix$IV) | !is.na(partAFix$DV),]
partANoNA

    ID  IV       DV
1    1  3.004821  75.88475
2    2  5.510845  95.51800
3    3  3.492998  78.42695
4    4  5.483592  96.88385
5    5  5.169892        NA
6    6  4.235609  87.29528
7    7  6.280710 103.53182
8    8  5.185404  89.23098
9    9  5.719443        NA
10  10  4.034938  85.06978
11  11  4.917672  94.50610
12  12  3.464703  80.30297
13  13  6.339799 100.78565
14  14  6.386308 102.24648
15  15  4.071437  83.38679
16  16  3.891094  81.24049
17  17        NA  84.97899
18  18  4.750984  86.73947
19  19  3.787786  83.45217
20  20  6.213811 104.40881
```

```
21   21  5.269975  91.41682
22   22  4.762756  87.90819
23   23        NA 100.44489
24   24  4.004267        NA
25   25  5.699614  97.85908
26   26        NA  88.46603
27   27  4.781962  91.30148
28   28  5.796401  97.97070
29   29  3.758305  80.11396
30   30  4.456300  87.85744
⋮    ⋮   ⋮         ⋮
612 612 4.838639  83.29365
613 613 3.804236  79.55434
614 614 5.176290  90.87047
615 615 4.104794  81.31605
616 616 4.532052  88.98090
617 617 5.813414  93.40338
618 618 5.837407  98.04301
619 619       NA  97.39008
620 620 4.809159  93.50235
621 621 3.417515        NA
622 622 6.633071 104.45610
623 623 5.438367  94.37462
624 624 6.708384 100.00226
625 625 4.771578  91.52980
626 626 5.180817  92.22548
627 627 6.776982 105.10097
628 628 4.467790  87.26828
629 629 4.378895  88.58807
630 630 5.600912  98.64919
631 631 4.310808  84.87648
632 632 4.330794  88.40596
633 633 6.314961        NA
634 634 4.277221  85.77395
635 635 5.715283 101.13068
636 636 5.882720 100.12748
637 637 4.474357  85.10011
638 638 4.303195  86.98506
639 639 3.985306  80.38932
640 640 5.831264  99.24615
641 641 4.521000  89.04763
```

```r
imputedSet <- mice(partANoNA, method = "norm.boot", printFlag = TRUE)
```

```
 iter imp variable
  1   1  IV  DV
  1   2  IV  DV
  1   3  IV  DV
  1   4  IV  DV
```

```
  1   5  IV  DV
  2   1  IV  DV
  2   2  IV  DV
  2   3  IV  DV
  2   4  IV  DV
  2   5  IV  DV
  3   1  IV  DV
  3   2  IV  DV
  3   3  IV  DV
  3   4  IV  DV
  3   5  IV  DV
  4   1  IV  DV
  4   2  IV  DV
  4   3  IV  DV
  4   4  IV  DV
  4   5  IV  DV
  5   1  IV  DV
  5   2  IV  DV
  5   3  IV  DV
  5   4  IV  DV
  5   5  IV  DV
```

```r
partAFinal <- complete(imputedSet)
partAFinal
```

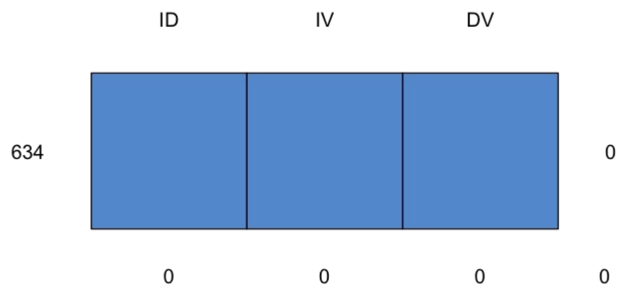
	ID	IV	DV
1	1	3.004821	75.88475
2	2	5.510845	95.51800
3	3	3.492998	78.42695
4	4	5.483592	96.88385
5	5	5.169892	94.60257
6	6	4.235609	87.29528
7	7	6.280710	103.53182
8	8	5.185404	89.23098
9	9	5.719443	94.85699
10	10	4.034938	85.06978
11	11	4.917672	94.50610
12	12	3.464703	80.30297
13	13	6.339799	100.78565
14	14	6.386308	102.24648
15	15	4.071437	83.38679
16	16	3.891094	81.24049
17	17	4.726524	84.97899
18	18	4.750984	86.73947
19	19	3.787786	83.45217
20	20	6.213811	104.40881
21	21	5.269975	91.41682
22	22	4.762756	87.90819
23	23	6.359214	100.44489
24	24	4.004267	84.73408

```
25  25  5.699614  97.85908
26  26  4.282892  88.46603
27  27  4.781962  91.30148
28  28  5.796401  97.97070
29  29  3.758305  80.11396
30  30  4.456300  87.85744
⋮   ⋮   ⋮         ⋮
605 612 4.838639  83.29365
606 613 3.804236  79.55434
607 614 5.176290  90.87047
608 615 4.104794  81.31605
609 616 4.532052  88.98090
610 617 5.813414  93.40338
611 618 5.837407  98.04301
612 619 4.998529  97.39008
613 620 4.809159  93.50235
614 621 3.417515  80.28067
615 622 6.633071 104.45610
616 623 5.438367  94.37462
617 624 6.708384 100.00226
618 625 4.771578  91.52980
619 626 5.180817  92.22548
620 627 6.776982 105.10097
621 628 4.467790  87.26828
622 629 4.378895  88.58807
623 630 5.600912  98.64919
624 631 4.310808  84.87648
625 632 4.330794  88.40596
626 633 6.314961 100.73420
627 634 4.277221  85.77395
628 635 5.715283 101.13068
629 636 5.882720 100.12748
630 637 4.474357  85.10011
631 638 4.303195  86.98506
632 639 3.985306  80.38932
633 640 5.831264  99.24615
634 641 4.521000  89.04763
```

```r
md.pattern(partAFinal)
```

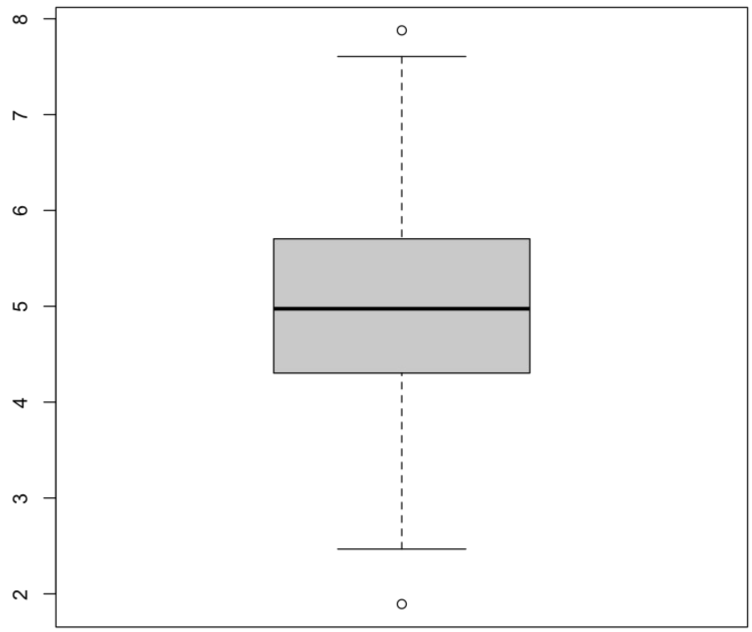
```
 /\     /\
{  `---'  }
{  O   O  }
==>  V <==  No need for mice. This data set is completely observed.
 \  \|/  /
  `-----'
```

```
    ID IV DV
634 1  1  1  0
    0  0  0  0
```



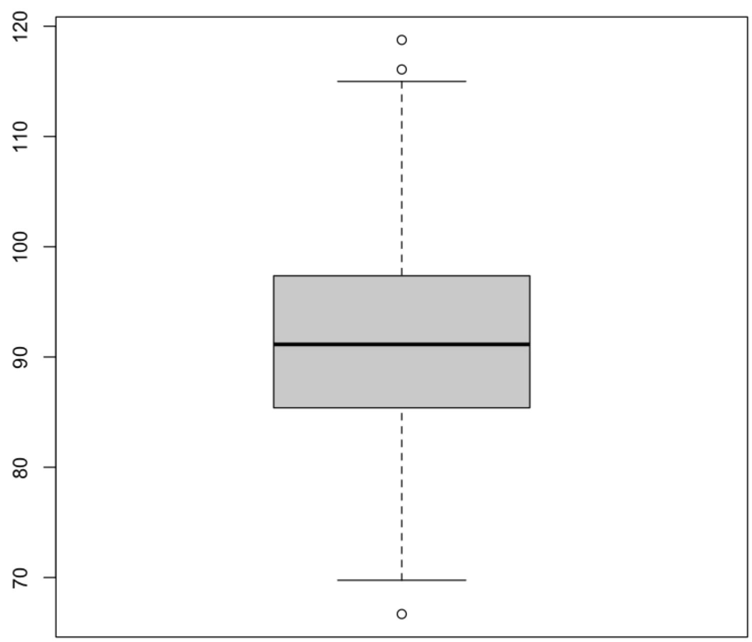

```
IVStats <- boxplot(partAFinal$IV, data = partAFinal)$stats
rownames(IVStats)<-c("Min","First Quartile","Median","Third
Quartile","Maximum")
colnames(IVStats) <- c("IV Stats")
IVStats
```

```
               IV Stats
Min            2.467449
First Quartile 4.303195
Median         4.974665
Third Quartile 5.703970
Maximum        7.606001
```

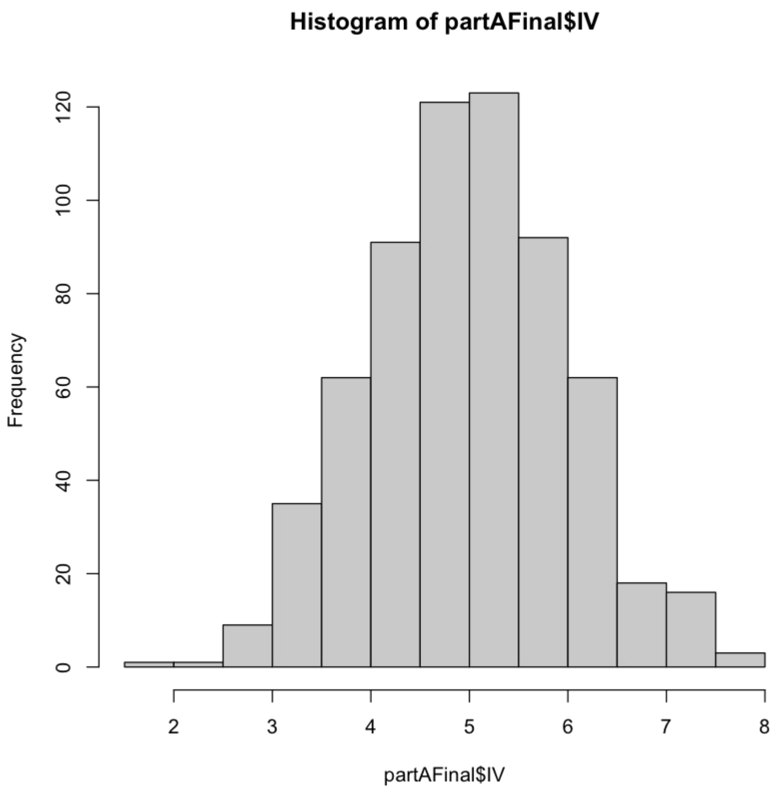



```r
DVStats <- boxplot(partAFinal$DV, data = partAFinal)$stats
rownames(DVStats)<-c("Min","First Quartile","Median","Third
Quartile","Maximum")
colnames(DVStats) <- c("DV Stats")
DVStats
```

```
                 DV Stats
Min              69.75385
First Quartile   85.39072
Median           91.14344
Third Quartile   97.36469
Maximum         114.99198
```




```
IVFreq <- hist(partAFinal$IV, data = partAFinal)

Warning message in plot.window(xlim, ylim, "", ...):
“"data" is not a graphical parameter”
Warning message in title(main = main, sub = sub, xlab = xlab, ylab =
ylab, ...):
“"data" is not a graphical parameter”
Warning message in axis(1, ...):
“"data" is not a graphical parameter”
Warning message in axis(2, at = yt, ...):
“"data" is not a graphical parameter”
```

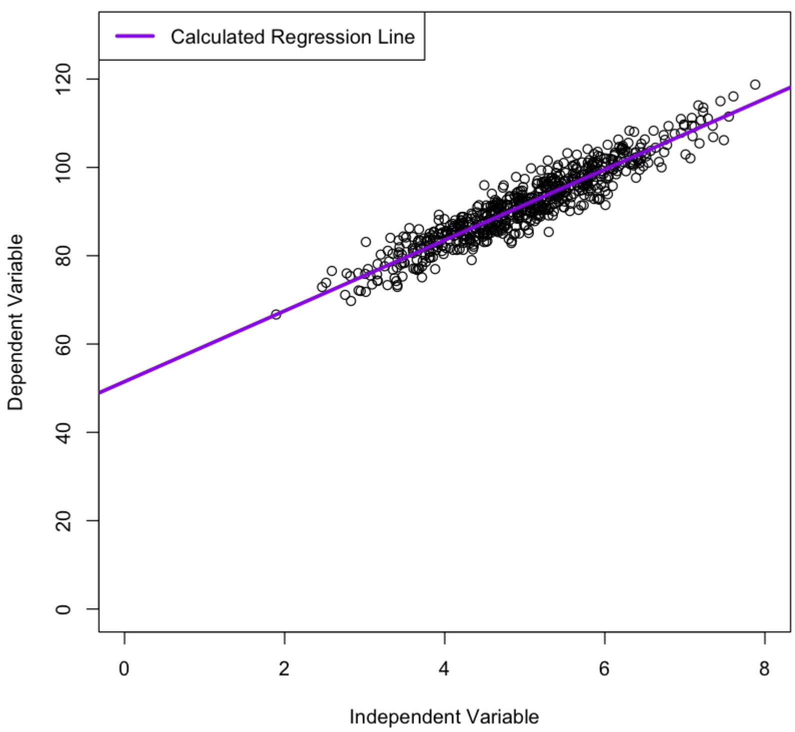



```r
#After completing the dataset, move onto calculating the bivariate
statistics.
partAModel <- lm(DV ~ IV, data=partAFinal)
summary(partAModel)
```

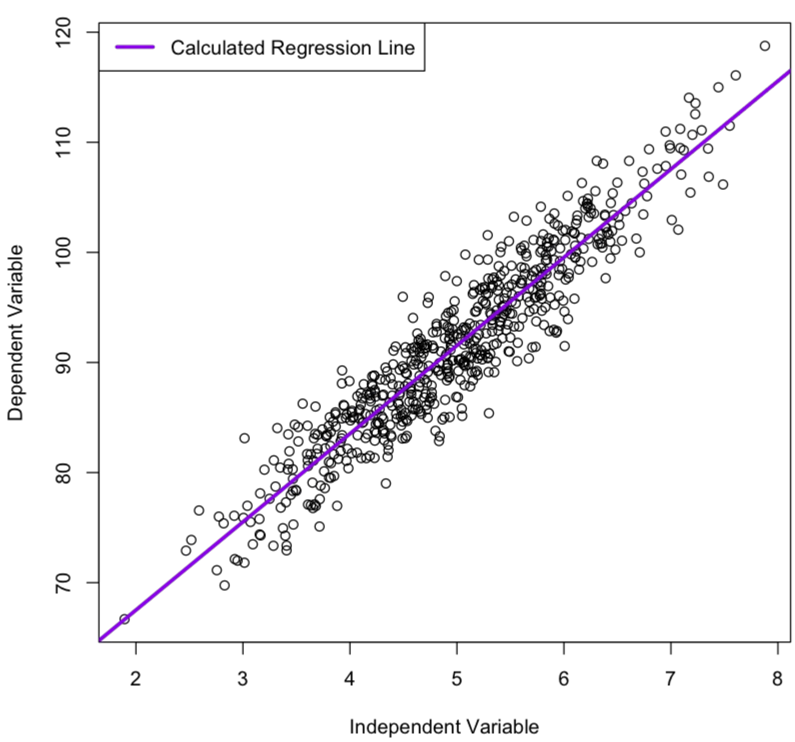
```
Call:
lm(formula = DV ~ IV, data = partAFinal)

Residuals:
    Min      1Q  Median      3Q     Max 
-8.5455 -2.1755  0.0453  2.1630  8.4793 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  51.4987     0.5967   86.31   <2e-16 ***
IV            8.0087     0.1172   68.32   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.943 on 632 degrees of freedom
Multiple R-squared:  0.8807,	Adjusted R-squared:  0.8805 
F-statistic:  4667 on 1 and 632 DF,  p-value: < 2.2e-16
```

```
#install.packages('knitir')
library(knitr)

kable(anova(partAModel), caption = "PART A ANOVA TABLE")
lm(DV ~ IV, data=partAFinal)



Table: PART A ANOVA TABLE

|          |  Df|    Sum Sq|      Mean Sq|  F value| Pr(>F)|
|:---------|---:|---------:|------------:|--------:|------:|
|IV        |   1| 40419.525| 40419.525156| 4667.282|      0|
|Residuals | 632|  5473.237|     8.660185|       NA|     NA|


Call:
lm(formula = DV ~ IV, data = partAFinal)

Coefficients:
(Intercept)           IV
     51.499        8.009

plot(partAFinal$DV ~ partAFinal$IV, main="DV against IV (zoomed out)",
xlab = "Independent Variable",xlim = c(0,8) ,ylim = c(0,130), ylab =
"Dependent Variable", pch = 21)
legend('topleft', legend = "Calculated Regression Line", lwd = 3, col
= "purple")
abline(partAModel, col = "purple", lwd = 3)

plot(partAFinal$DV ~ partAFinal$IV, main="DV against IV (zoomed in)",
xlab = "Independent Variable", ylab = "Dependent Variable", pch = 21)
abline(partAModel, col = "purple", lwd = 3)
legend('topleft', legend = "Calculated Regression Line", lwd = 3, col
= "purple")
```

DV against IV (zoomed out)



DV against IV (zoomed in)



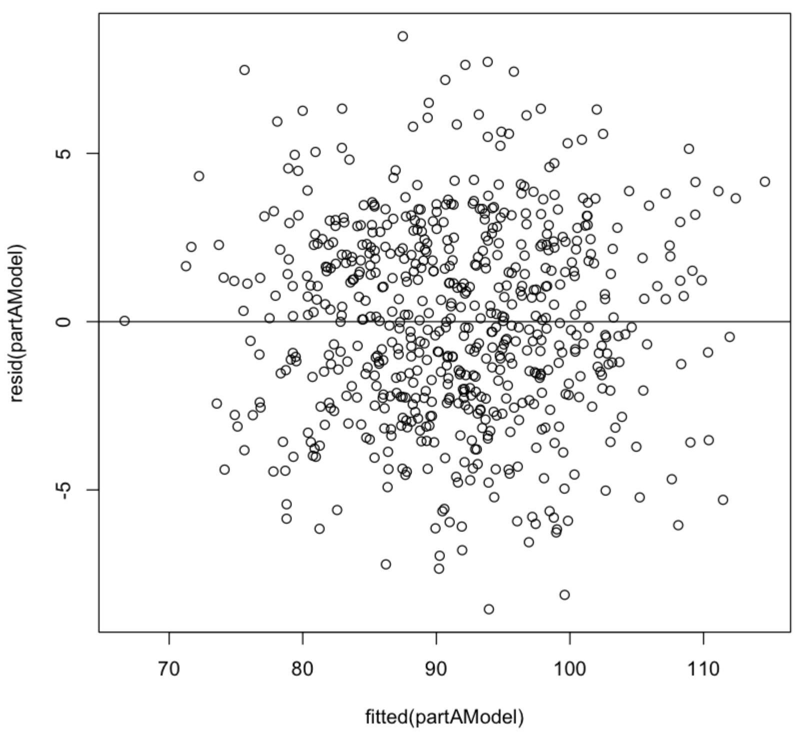
```r
confint(partAModel, level = 0.90)
confint(partAModel, level = 0.95)
confint(partAModel, level = 0.99)
```

```
                  5 %      95 %
(Intercept) 50.51589 52.481600
IV           7.81562  8.201832
```

```
                 2.5 %     97.5 %
(Intercept) 50.327077 52.67042
IV           7.778522  8.23893
```

```
                 0.5 %     99.5 %
(Intercept) 49.957206 53.04029
IV           7.705852  8.31160
```

```r
#plot box plot of each variable (maybe a histogram)
#plot residual against predicted
#Done

plot(fitted(partAModel), resid(partAModel), abline(0,0))
```




```r
#There is no obvious pattern, showing that there is no transformation
needed for part A.
```

```r
partBStart <- read.csv("378631_partB.csv", header = TRUE)
partBStart
```

	ID	x	y
1	1	3.953044	841.16920
2	2	3.371096	291.48824
3	3	2.830730	121.15881
4	4	3.617563	135.17602
5	5	2.757039	62.32664
6	6	2.632387	61.95896
7	7	3.372841	341.26199
8	8	2.566523	134.28516
9	9	2.846186	117.93309
10	10	3.698594	204.32728
11	11	3.606219	467.49029
12	12	2.811418	96.08189
13	13	2.908577	243.90988
14	14	3.126886	150.73344
15	15	3.012225	126.41770
16	16	2.789821	57.80047
17	17	3.879373	554.31502
18	18	2.942675	143.24687
19	19	3.742891	137.87775
20	20	2.922001	194.47741
21	21	3.927682	329.81609
22	22	3.640303	160.95731
23	23	3.164244	105.40885
24	24	2.556463	75.42010
25	25	3.775686	287.76219
26	26	3.171728	169.19542
27	27	2.913990	82.17532
28	28	3.365548	205.02964
29	29	2.990624	182.96686
30	30	2.653300	95.76600
⋮	⋮	⋮	⋮
420	420	2.840384	100.78389
421	421	3.402311	177.90133
422	422	2.551892	195.05385
423	423	3.625051	176.19481
424	424	3.163110	101.37462
425	425	2.816197	202.97836
426	426	2.951745	100.33448
427	427	2.538393	72.49762
428	428	2.836178	122.13170
429	429	2.905612	207.64019
430	430	3.573616	200.28828
431	431	2.562395	74.38383
432	432	3.043727	215.32288
433	433	3.533629	143.21095
434	434	2.745954	199.56952

```
435 435 3.591877 303.52652
436 436 3.450903 307.62896
437 437 3.255505 214.35423
438 438 3.656920 446.20926
439 439 3.793343 580.14026
440 440 2.548607  53.54303
441 441 3.935034 319.33627
442 442 3.608533 388.10683
443 443 3.172032 182.41110
444 444 3.407523 270.68846
445 445 2.892556 109.13102
446 446 3.600378 225.15345
447 447 3.130015 111.30049
448 448 3.210704  92.17601
449 449 3.177238 225.64685
```

```
#Checking if data is clean and unique
any(is.na(partBStart))
any(is.null(partBStart))
any(is.nan(partBStart[,1]))
any(is.nan(partBStart[,2]))
any(is.nan(partBStart[,3]))

length(unique(partBStart[,1]))
length(unique(partBStart[,2]))
length(unique(partBStart[,3]))
```

```
[1] FALSE

[1] FALSE

[1] FALSE

[1] FALSE

[1] FALSE

[1] 449

[1] 449

[1] 449
```

```
#Here is the r value, b1 and b0.
partBModel <- lm(y ~ x, data=partBStart)
summary(partBModel)
```

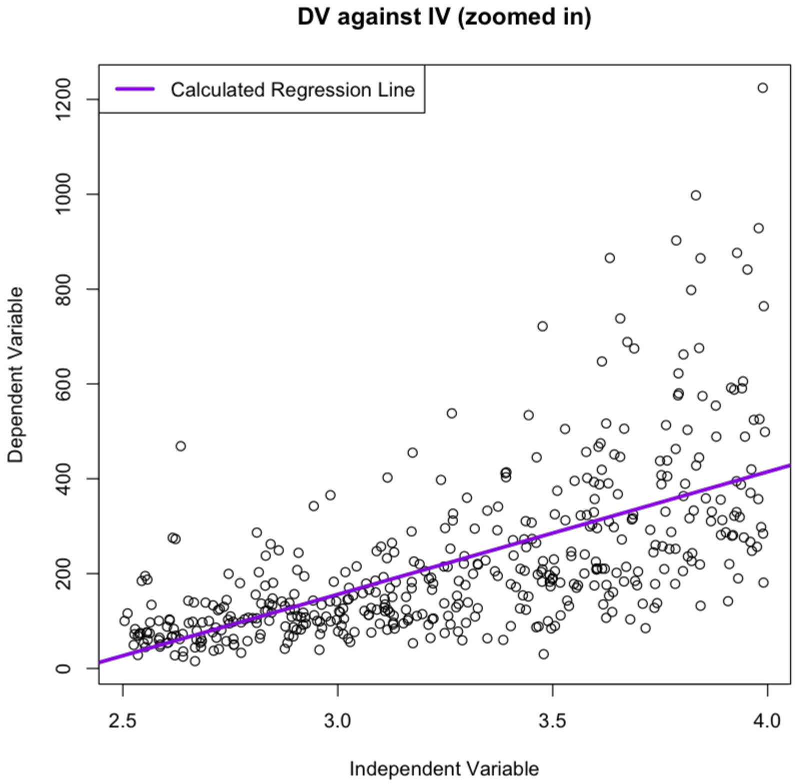
```
Call:
lm(formula = y ~ x, data = partBStart)

Residuals:
```

```
    Min      1Q  Median      3Q     Max 
-256.46  -83.00  -16.96   46.30  812.36 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -619.61      49.13  -12.61   <2e-16 ***
x             258.64      14.93   17.33   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 136.6 on 447 degrees of freedom
Multiple R-squared:  0.4018,	Adjusted R-squared:  0.4005 
F-statistic: 300.2 on 1 and 447 DF,  p-value: < 2.2e-16
```

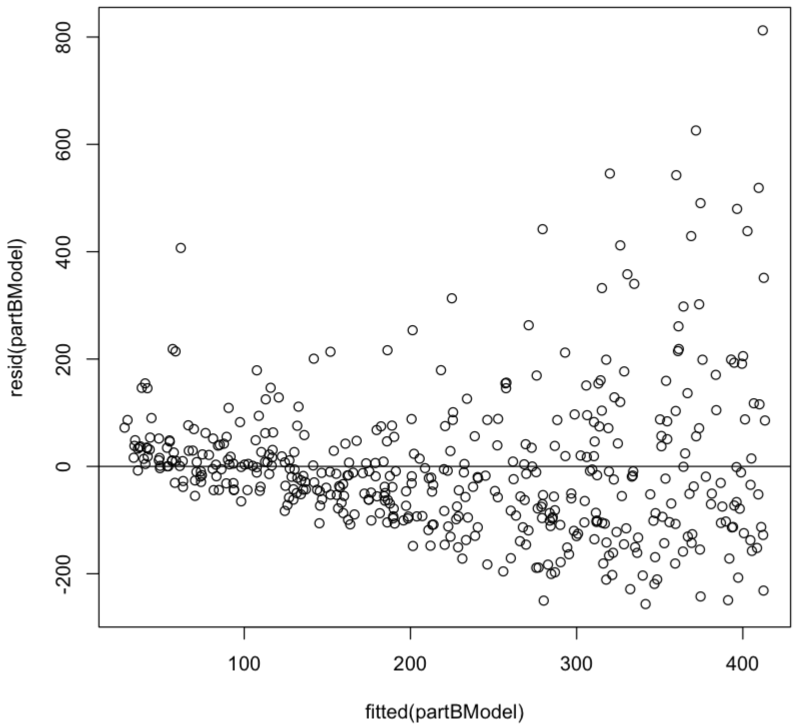
```
kable(anova(partBModel), caption = "PART B ANOVA TABLE (no
transformation no bin)")
lm(y ~ x, data=partBStart)
```

```
Table: PART B ANOVA TABLE (no transformation no bin)

|          |  Df|  Sum Sq|    Mean Sq|  F value| Pr(>F)|
|:---------|---:|-------:|----------:|--------:|------:|
|x         |   1| 5603044| 5603043.72| 300.2391|      0|
|Residuals | 447| 8341888|   18661.94|       NA|     NA|


Call:
lm(formula = y ~ x, data = partBStart)

Coefficients:
(Intercept)            x  
     -619.6        258.6  
```

```
plot(partBStart$y ~ partBStart$x, main="DV against IV (zoomed in)",
xlab = "Independent Variable", ylab = "Dependent Variable", pch = 21)
abline(partBModel, col = "purple", lwd = 3)
legend('topleft', legend = "Calculated Regression Line", lwd = 3, col
= "purple")
```



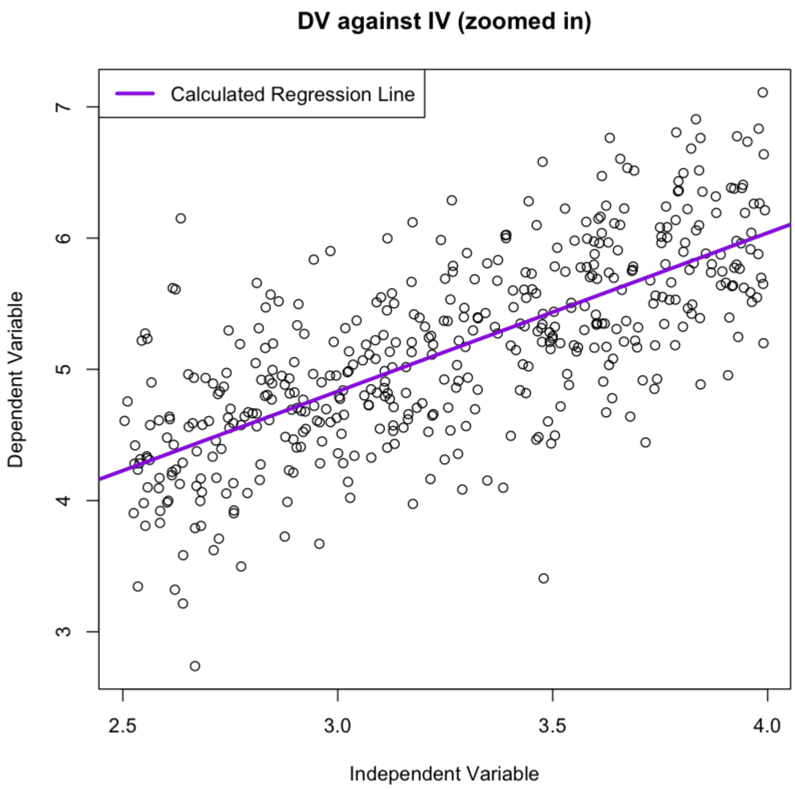

```
plot(fitted(partBModel), resid(partBModel), abline(0,0))
```



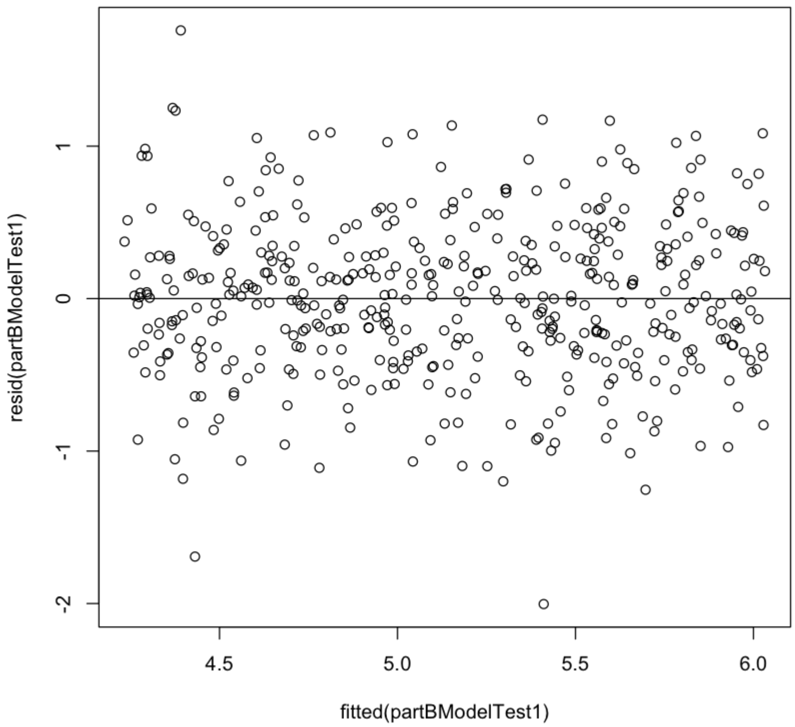

```
#Ask TA's for part B, what order should the binning, LOF, and
transformations be placed in, or we supposed to do it twice.
#Transformation, summary table, lack of fit,
#bin after best transformation, then lack of fit

#And do we try different transformations and perform LOF multiple
times too?
#Also ask what value to bin by.

#First test is to try th natrual log of the dependent variable.
partBTest1 <- partBStart
partBTest1$y <- log(partBTest1$y)
partBModelTest1 <- lm(y ~ x, data=partBTest1)
summary(partBModelTest1)


Call:
lm(formula = y ~ x, data = partBTest1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.00352 -0.32814 -0.00075  0.31585  1.75915

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept)  1.20942    0.18488   6.542 1.67e-10 ***
x            1.20761    0.05617  21.499  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.5141 on 447 degrees of freedom
Multiple R-squared:  0.5084,	Adjusted R-squared:  0.5073
F-statistic: 462.2 on 1 and 447 DF,  p-value: < 2.2e-16
```

```
plot(partBTest1$y ~ partBTest1$x, main="DV against IV (zoomed in)",
xlab = "Independent Variable", ylab = "Dependent Variable", pch = 21)
abline(partBModelTest1, col = "purple", lwd = 3)
legend('topleft', legend = "Calculated Regression Line", lwd = 3, col
= "purple")
```




```
plot(fitted(partBModelTest1), resid(partBModelTest1), abline(0,0))
```




```
#Second test is to try th natrual log of the independent variable
(nothing on dependent) (bad)
partBTest2 <- partBStart
partBTest2$x <- log(partBTest2$x)
partBModelTest2 <- lm(y ~ x, data=partBTest2)
summary(partBModelTest2)


Call:
lm(formula = y ~ x, data = partBTest2)

Residuals:
    Min      1Q  Median      3Q     Max
-253.80  -87.17  -18.19   46.01  828.72

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -734.45      57.39  -12.80   <2e-16 ***
x             816.87      48.58   16.81   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 138.2 on 447 degrees of freedom
Multiple R-squared:  0.3874,     Adjusted R-squared:  0.386
F-statistic: 282.7 on 1 and 447 DF,  p-value: < 2.2e-16
```

```r
#Third test is to try squaring the independent variable (nothing on
dependent) (bad)
partBTest3 <- partBStart
partBTest3$x <- (partBTest3$x)^2
partBModelTest3 <- lm(y ~ x, data=partBTest3)
summary(partBModelTest3)
```

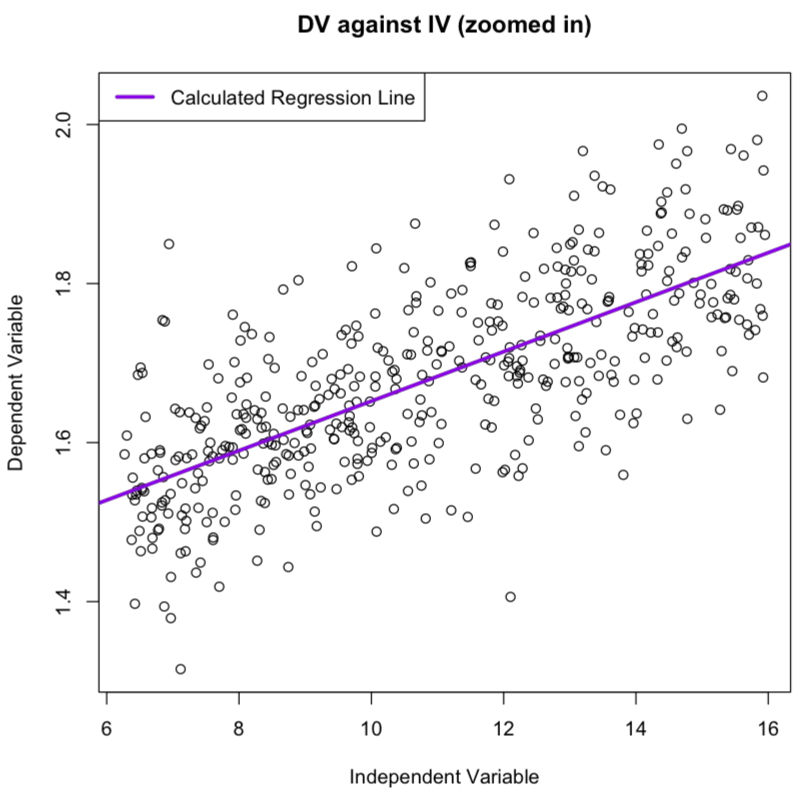
```

Call:
lm(formula = y ~ x, data = partBTest3)

Residuals:
    Min      1Q  Median      3Q     Max 
-260.95  -82.44  -13.31   43.45  795.90 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -211.390     25.353  -8.338 9.44e-16 ***
x             40.219      2.265  17.756  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 135.3 on 447 degrees of freedom
Multiple R-squared:  0.4136,	Adjusted R-squared:  0.4123 
F-statistic: 315.3 on 1 and 447 DF,  p-value: < 2.2e-16
```

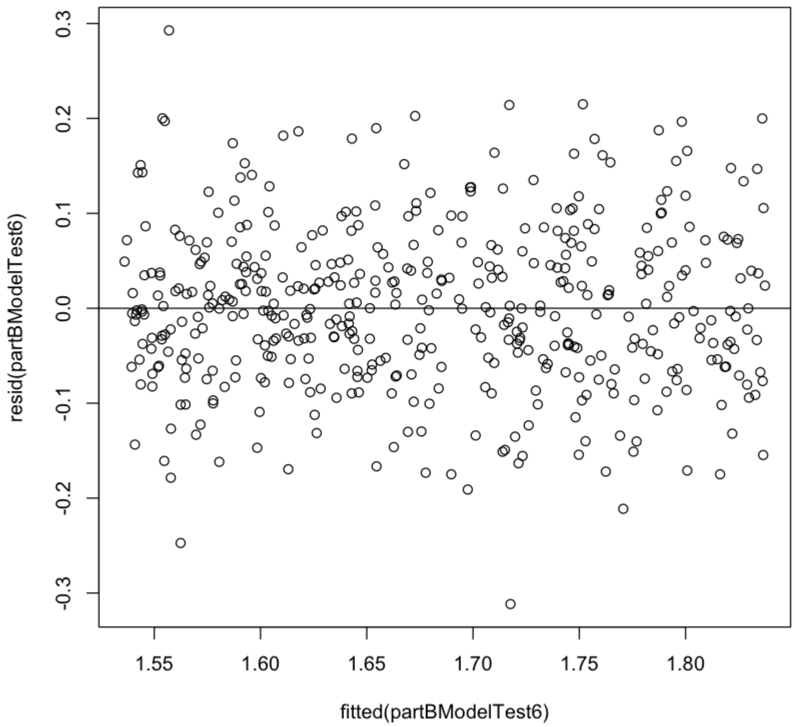
```r
#Fourth test is to try making the sqrt transformation better on the
dependent variable ()
partBTest4 <- partBStart
partBTest4$y <- sqrt(partBTest4$y)
partBModelTest4 <- lm(y ~ x, data=partBTest4)
summary(partBModelTest4)
```

```

Call:
lm(formula = y ~ x, data = partBTest4)

Residuals:
     Min       1Q   Median       3Q      Max 
-10.3526  -2.4799  -0.3552   1.8743  14.8699 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -13.3351     1.3600  -9.806   <2e-16 ***
x             8.3878     0.4132  20.300   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.782 on 447 degrees of freedom
Multiple R-squared:  0.4797,	Adjusted R-squared:  0.4785 
F-statistic: 412.1 on 1 and 447 DF,  p-value: < 2.2e-16
```

```
#Fifth test is to try making the sqrt transformation better on the
dependent variable ()
partBTest5 <- partBStart
partBTest5$y <- sqrt(partBTest5$y)
partBModelTest5 <- lm(y ~ x, data=partBTest5)
summary(partBModelTest5)


Call:
lm(formula = y ~ x, data = partBTest5)

Residuals:
     Min       1Q   Median       3Q      Max
-10.3526  -2.4799  -0.3552   1.8743  14.8699

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -13.3351     1.3600  -9.806   <2e-16 ***
x             8.3878     0.4132  20.300   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.782 on 447 degrees of freedom
Multiple R-squared:  0.4797,	Adjusted R-squared:  0.4785
F-statistic: 412.1 on 1 and 447 DF,  p-value: < 2.2e-16

#Try to find some other method of finding a transformation, other than
that I check the others mentioned
#in the textbook and then choose the best one.

#Sixth test is to try combing the ln on the dependent with another
transformation on the x. #r^2 0.5084 should be beat
partBTest6 <- partBStart
partBTest6$y <- exp(0.1)^log(partBTest6$y)
partBTest6$x <- exp(2)^log(partBTest6$x)
partBModelTest6 <- lm(y ~ x, data=partBTest6)
summary(partBModelTest6)


Call:
lm(formula = y ~ x, data = partBTest6)

Residuals:
      Min        1Q    Median        3Q       Max
-0.311664 -0.055052 -0.002726  0.049161  0.292755

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.340881   0.016147   83.04   <2e-16 ***
x           0.031127   0.001443   21.58   <2e-16 ***
---
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.08614 on 447 degrees of freedom
Multiple R-squared:  0.5102,	Adjusted R-squared:  0.5091 
F-statistic: 465.6 on 1 and 447 DF,  p-value: < 2.2e-16

plot(partBTest6$y ~ partBTest6$x, main="DV against IV (zoomed in)",
xlab = "Independent Variable", ylab = "Dependent Variable", pch = 21)
abline(partBModelTest6, col = "purple", lwd = 3)
legend('topleft', legend = "Calculated Regression Line", lwd = 3, col
= "purple")
plot(fitted(partBModelTest6), resid(partBModelTest6), abline(0,0))
```



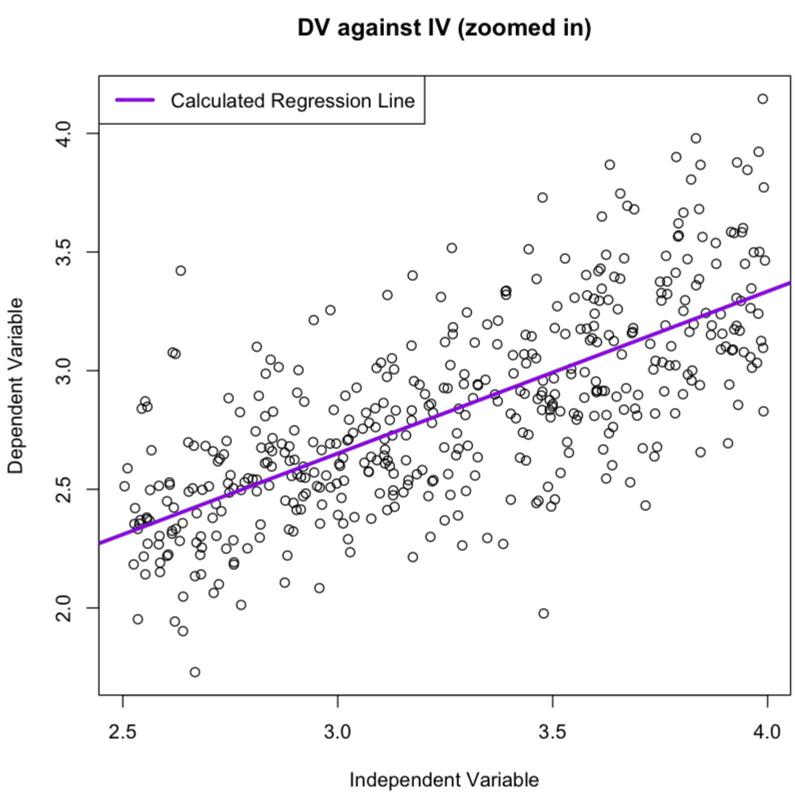



```
partBTest6
```

	ID	x	y
1	1	15.626556	1.961049
2	2	11.364287	1.763852
3	3	8.013033	1.615606
4	4	13.086761	1.633390
5	5	7.601262	1.511706
6	6	6.929462	1.510812
7	7	11.376058	1.791879
8	8	6.587043	1.632311
9	9	8.100773	1.611252
10	10	13.679595	1.702286
11	11	13.004818	1.849172
12	12	7.904073	1.578571
13	13	8.459822	1.732698
14	14	9.777415	1.651281
15	15	9.073497	1.622485
16	16	7.783101	1.500352
17	17	15.049534	1.880943
18	18	8.659335	1.642890
19	19	14.009235	1.636626
20	20	8.538091	1.693897
21	21	15.426686	1.785777

```
22  22  13.251806 1.662153
23  23  10.012437 1.593264
24  24   6.535504 1.540809
25  25  14.255801 1.761584
26  26  10.059861 1.670471
27  27   8.491340 1.554083
28  28  11.326910 1.702871
29  29   8.943832 1.683594
30  30   7.039999 1.578051
⋮   ⋮   ⋮         ⋮
420 420  8.067782 1.586131
421 421 11.575718 1.678873
422 422  6.512153 1.694398
423 423 13.140993 1.677256
424 424 10.005267 1.587058
425 425  7.930968 1.701159
426 426  8.712799 1.585423
427 427  6.443439 1.534731
428 428  8.043908 1.616899
429 429  8.442580 1.705027
430 430 12.770730 1.698891
431 431  6.565867 1.538678
432 432  9.264274 1.711232
433 433 12.486534 1.642849
434 434  7.540264 1.698280
435 435 12.901583 1.771004
436 436 11.908729 1.773384
437 437 10.598311 1.710461
438 438 13.373060 1.840576
439 439 14.389454 1.889528
440 440  6.495399 1.488916
441 441 15.484495 1.780020
442 442 13.021509 1.815077
443 443 10.061787 1.683081
444 444 11.611215 1.750842
445 445  8.366879 1.598803
446 446 12.962724 1.718889
447 447  9.796992 1.601953
448 448 10.308618 1.572033
449 449 10.094842 1.719265

#Sixth test is to try combing the ln on the dependent with another
transformation on the x. #r^2 0.5084 should be beat
partBTest7 <- partBStart
partBTest7$y <- (partBTest7$y)^(1/5)
partBTest7$x <- (partBTest7$x)
partBModelTest7 <- lm(y ~ x, data=partBTest7)
summary(partBModelTest7)

plot(partBTest7$y ~ partBTest7$x, main="DV against IV (zoomed in)",
```

```
xlab = "Independent Variable", ylab = "Dependent Variable", pch = 21)
abline(partBModelTest7, col = "purple", lwd = 3)
legend('topleft', legend = "Calculated Regression Line", lwd = 3, col
= "purple")
#plot(fitted(partBModelTest6), resid(partBModelTest6), abline(0,0))


Call:
lm(formula = y ~ x, data = partBTest7)

Residuals:
     Min       1Q   Median       3Q      Max
-1.00195 -0.19090 -0.01977  0.16620  1.01956

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.6019     0.1053   5.715 2.01e-08 ***
x             0.6832     0.0320  21.349  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2929 on 447 degrees of freedom
Multiple R-squared:  0.5049,    Adjusted R-squared:  0.5038
F-statistic: 455.8 on 1 and 447 DF,  p-value: < 2.2e-16
```



```r
#Will proceed with test 6 since I am at wits end.

transformedPartB <- partBTest6

groups <- cut(transformedPartB$x,breaks=c(-
Inf,seq(min(transformedPartB$x)+0.1, max(transformedPartB$x)-
0.1,by=0.1),Inf))
table(groups)
```

```
groups
(-Inf,6.37] (6.37,6.47] (6.47,6.57] (6.57,6.67] (6.67,6.77]
(6.77,6.87]
          2           7           9           2           5
10
(6.87,6.97] (6.97,7.07] (7.07,7.17] (7.17,7.27] (7.27,7.37]
(7.37,7.47]
          6           3           6           6           4
6
(7.47,7.57] (7.57,7.67] (7.67,7.77] (7.77,7.87] (7.87,7.97]
(7.97,8.07]
          7           4           4           3           8
6
(8.07,8.17] (8.17,8.27] (8.27,8.37] (8.37,8.47] (8.47,8.57]
(8.57,8.67]
          5           2           8           8           7
2
(8.67,8.77] (8.77,8.87] (8.87,8.97] (8.97,9.07] (9.07,9.17]
(9.17,9.27]
          5           3           4           6           8
4
(9.27,9.37] (9.37,9.47] (9.47,9.57] (9.57,9.67] (9.67,9.77]
(9.77,9.87]
          1           5           6           5           8
8
(9.87,9.97] (9.97,10.1] (10.1,10.2] (10.2,10.3] (10.3,10.4]
(10.4,10.5]
          1           7           4           3           6
2
(10.5,10.6] (10.6,10.7] (10.7,10.8] (10.8,10.9]   (10.9,11]
(11,11.1]
          5           5           7           4           3
6
(11.1,11.2] (11.2,11.3] (11.3,11.4] (11.4,11.5] (11.5,11.6]
(11.6,11.7]
          0           3           2           3           3
5
(11.7,11.8] (11.8,11.9]   (11.9,12]   (12,12.1] (12.1,12.2]
(12.2,12.3]
          3           7           2           6           5
8
```

```
(12.3,12.4] (12.4,12.5] (12.5,12.6] (12.6,12.7] (12.7,12.8]
(12.8,12.9]
          4           4           4           4           1
6
  (12.9,13]   (13,13.1] (13.1,13.2] (13.2,13.3] (13.3,13.4]
(13.4,13.5]
          6          10           5           7           3
5
(13.5,13.6] (13.6,13.7] (13.7,13.8] (13.8,13.9]   (13.9,14]
(14,14.1]
          3           6           2           1           3
4
(14.1,14.2] (14.2,14.3] (14.3,14.4] (14.4,14.5] (14.5,14.6]
(14.6,14.7]
          6           4           4           4           5
5
(14.7,14.8] (14.8,14.9]   (14.9,15]   (15,15.1] (15.1,15.2]
(15.2,15.3]
          5           3           2           3           3
1
(15.3,15.4] (15.4,15.5] (15.5,15.6] (15.6,15.7] (15.7,15.8] (15.8,
Inf]
          5           6           5           2           5
10

x <- ave(transformedPartB$x, groups)
binned_partB <- data.frame(x=x, y=transformedPartB$y)
binned_partB


#Testing if bin worked, which it did.
options(digits = 22)
binned_partB$x[1]
sum(binned_partB$x == 15.603831133408823106379)
options(digits = 7)

    x         y
1   15.603831 1.961049
2   11.345598 1.763852
3    8.038317 1.615606
4   13.122549 1.633390
5    7.604706 1.511706
6    6.927471 1.510812
7   11.401560 1.791879
8    6.578855 1.632311
9    8.105054 1.611252
10  13.719521 1.702286
11  13.017603 1.849172
12   7.924418 1.578571
13   8.421249 1.732698
```

```
14    9.795836 1.651281
15    9.116063 1.622485
16    7.808865 1.500352
17   15.028568 1.880943
18    8.663540 1.642890
19   14.005922 1.636626
20    8.525755 1.693897
21   15.418681 1.785777
22   13.213038 1.662153
23   10.021985 1.593264
24    6.522941 1.540809
25   14.212834 1.761584
26   10.021985 1.670471
27    8.525755 1.554083
28   11.345598 1.702871
29    8.907857 1.683594
30    7.015006 1.578051
⋮   ⋮         ⋮
420  8.038317 1.586131
421 11.608546 1.678873
422  6.522941 1.694398
423 13.122549 1.677256
424 10.021985 1.587058
425  7.924418 1.701159
426  8.726951 1.585423
427  6.414915 1.534731
428  8.038317 1.616899
429  8.421249 1.705027
430 12.806752 1.698891
431  6.522941 1.538678
432  9.221666 1.711232
433 12.527483 1.642849
434  7.527761 1.698280
435 12.934256 1.771004
436 11.913418 1.773384
437 10.629577 1.710461
438 13.405646 1.840576
439 14.402633 1.889528
440  6.522941 1.488916
441 15.524861 1.780020
442 13.017603 1.815077
443 10.021985 1.683081
444 11.608546 1.750842
445  8.315194 1.598803
446 12.934256 1.718889
447  9.795836 1.601953
448 10.340512 1.572033
449 10.096104 1.719265

[1] 15.6038311334088231O638
```

```
[1] 0
```

```r
#install.packages('remotes')
library(remotes)
#install_github("cran/alr3")
library(alr3)
```

```
Loading required package: car

Loading required package: carData
```

```r
fit_b <- lm(y ~ x, data = binned_partB)
pureErrorAnova(fit_b)
```

```
               Df  Sum Sq    Mean Sq     F value     Pr(>F)
x               1 3.4549212 3.454921157 455.9396415 1.344063e-65
Residuals     447 3.3165053 0.007419475          NA           NA
 Lack of fit   93 0.6340404 0.006817638   0.8997113 7.263212e-01
 Pure Error   354 2.6824649 0.007577584          NA           NA
```

```r
groups <- cut(transformedPartB$x,breaks=c(-
Inf,seq(min(transformedPartB$x)+0.05, max(transformedPartB$x)-
0.05,by=0.05),Inf))
table(groups)
x <- ave(transformedPartB$x, groups)
binned_partB <- data.frame(x=x, y=transformedPartB$y)
fit_b <- lm(y ~ x, data = binned_partB)
summary(fit_b)
pureErrorAnova(fit_b)
```

```
groups
  (-Inf,6.32]   (6.32,6.37]   (6.37,6.42]   (6.42,6.47]   (6.47,6.52]
            2             0             3             4             4
  (6.52,6.57]   (6.57,6.62]   (6.62,6.67]   (6.67,6.72]   (6.72,6.77]
            5             2             0             5             0
  (6.77,6.82]   (6.82,6.87]   (6.87,6.92]   (6.92,6.97]   (6.97,7.02]
            5             5             2             4             1
  (7.02,7.07]   (7.07,7.12]   (7.12,7.17]   (7.17,7.22]   (7.22,7.27]
            2             4             2             5             1
  (7.27,7.32]   (7.32,7.37]   (7.37,7.42]   (7.42,7.47]   (7.47,7.52]
            1             3             4             2             3
  (7.52,7.57]   (7.57,7.62]   (7.62,7.67]   (7.67,7.72]   (7.72,7.77]
            4             4             0             3             1
  (7.77,7.82]   (7.82,7.87]   (7.87,7.92]   (7.92,7.97]   (7.97,8.02]
            2             1             4             4             2
  (8.02,8.07]   (8.07,8.12]   (8.12,8.17]   (8.17,8.22]   (8.22,8.27]
            4             5             0             1             1
  (8.27,8.32]   (8.32,8.37]   (8.37,8.42]   (8.42,8.47]   (8.47,8.52]
            4             4             4             4             2
  (8.52,8.57]   (8.57,8.62]   (8.62,8.67]   (8.67,8.72]   (8.72,8.77]
```

5	0	2	2	3
(8.77,8.82]	(8.82,8.87]	(8.87,8.92]	(8.92,8.97]	(8.97,9.02]
2	1	3	1	4
(9.02,9.07]	(9.07,9.12]	(9.12,9.17]	(9.17,9.22]	(9.22,9.27]
2	4	4	2	2
(9.27,9.32]	(9.32,9.37]	(9.37,9.42]	(9.42,9.47]	(9.47,9.52]
0	1	2	3	2
(9.52,9.57]	(9.57,9.62]	(9.62,9.67]	(9.67,9.72]	(9.72,9.77]
4	2	3	6	2
(9.77,9.82]	(9.82,9.87]	(9.87,9.92]	(9.92,9.97]	(9.97,10.02]
7	1	0	1	4
(10.02,10.07]	(10.07,10.12]	(10.12,10.17]	(10.17,10.22]	(10.22,10.27]
3	3	1	2	1
(10.27,10.32]	(10.32,10.37]	(10.37,10.42]	(10.42,10.47]	(10.47,10.52]
2	4	2	0	1
(10.52,10.57]	(10.57,10.62]	(10.62,10.67]	(10.67,10.72]	(10.72,10.77]
4	2	3	2	5
(10.77,10.82]	(10.82,10.87]	(10.87,10.92]	(10.92,10.97]	(10.97,11.02]
0	4	3	0	3
(11.02,11.07]	(11.07,11.12]	(11.12,11.17]	(11.17,11.22]	(11.22,11.27]
3	0	0	3	0
(11.27,11.32]	(11.32,11.37]	(11.37,11.42]	(11.42,11.47]	(11.47,11.52]
0	2	2	1	3
(11.52,11.57]	(11.57,11.62]	(11.62,11.67]	(11.67,11.72]	(11.72,11.77]
0	3	2	0	3
(11.77,11.82]	(11.82,11.87]	(11.87,11.92]	(11.92,11.97]	(11.97,12.02]
4	3	2	0	5
(12.02,12.07]	(12.07,12.12]	(12.12,12.17]	(12.17,12.22]	(12.22,12.27]
1	5	0	4	4
(12.27,12.32]	(12.32,12.37]	(12.37,12.42]	(12.42,12.47]	(12.47,12.52]
3	1	2	2	2
(12.52,12.57]	(12.57,12.62]	(12.62,12.67]	(12.67,12.72]	(12.72,12.77]
2	2	2	1	0
(12.77,12.82]	(12.82,12.87]	(12.87,12.92]	(12.92,12.97]	(12.97,13.02]
5	1	1	5	5
(13.02,13.07]	(13.07,13.12]	(13.12,13.17]	(13.17,13.22]	(13.22,13.27]
5	2	3	4	3
(13.27,13.32]	(13.32,13.37]	(13.37,13.42]	(13.42,13.47]	(13.47,13.52]
2	1	3	2	2
(13.52,13.57]	(13.57,13.62]	(13.62,13.67]	(13.67,13.72]	(13.72,13.77]
1	4	2	1	1
(13.77,13.82]	(13.82,13.87]	(13.87,13.92]	(13.92,13.97]	(13.97,14.02]
1	0	1	2	3
(14.02,14.07]	(14.07,14.12]	(14.12,14.17]	(14.17,14.22]	(14.22,14.27]
1	3	3	2	2
(14.27,14.32]	(14.32,14.37]	(14.37,14.42]	(14.42,14.47]	(14.47,14.52]
0	4	3	1	3
(14.52,14.57]	(14.57,14.62]	(14.62,14.67]	(14.67,14.72]	(14.72,14.77]
2	3	2	2	3

```
(14.77,14.82] (14.82,14.87] (14.87,14.92] (14.92,14.97] (14.97,15.02]
            3             0             1             1             1
(15.02,15.07] (15.07,15.12] (15.12,15.17] (15.17,15.22] (15.22,15.27]
            2             0             3             0             1
(15.27,15.32] (15.32,15.37] (15.37,15.42] (15.42,15.47] (15.47,15.52]
            2             3             2             4             2
(15.52,15.57] (15.57,15.62] (15.62,15.67] (15.67,15.72] (15.72,15.77]
            3             1             1             4             1
(15.77,15.82] (15.82,15.87]  (15.87, Inf]
            1             3             6


Call:
lm(formula = y ~ x, data = binned_partB)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.311182 -0.054744 -0.002761  0.049268  0.292453 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 1.340884   0.016147   83.04   <2e-16 ***
x           0.031126   0.001443   21.58   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.08614 on 447 degrees of freedom
Multiple R-squared:  0.5102,	Adjusted R-squared:  0.5091 
F-statistic: 465.5 on 1 and 447 DF,  p-value: < 2.2e-16

             Df  Sum Sq   Mean Sq     F value     Pr(>F)
x             1 3.454506 3.454506273 448.3577139 3.819340e-60
Residuals   447 3.316920 0.007420403          NA           NA
 Lack of fit 166 1.151872 0.006938985   0.9006055 7.703236e-01
 Pure Error  281 2.165049 0.007704799          NA           NA

#Ask if data set is bad, also r^2 value was low, struggling to find a transformation that would help.
#p value for lack of fit is low, is that ok?

#Given the nature of the dataset after the transformation, opted for a bin of smaller intervals. Upon testing,
#it also increased the p-value, meaning the variation as a result of the bin was reduced.

#Getting the correlation coefficient from model6, the model I decided on to use. it is positive since we can see
#on the scatterplot that the linear relationship is positive.
sqrt(0.5102)

[1] 0.7142829
```

```r
confint(fit_b, level = 0.90)
confint(fit_b, level = 0.95)
confint(fit_b, level = 0.99)
```

```
                  5 %        95 %
(Intercept) 1.31426885 1.36749903
x           0.02874859 0.03350422
```

```
                  2.5 %      97.5 %
(Intercept) 1.30914991 1.37261797
x           0.02829126 0.03396155
```

```
                  0.5 %      99.5 %
(Intercept) 1.29911296 1.38265492
x           0.02739455 0.03485826
```