Chapter 12: OLS Assumptions and Diagnostic Testing

Answers to Exercises

Brian Fogarty

Contents

Exercise 1

Re-run `model.1` from Chapter 11 Exercises.

```
library(tidyverse)
simd <- read_csv("simd2020.csv", na = "*")

simd <- simd %>%
  mutate(pct_income_deprived = Income_rate*100,
         pct_employment_deprived = Employment_rate*100,
         pct_not_participating = not_participating*100)

summary(model.1 <- lm(crime_rate ~ pct_income_deprived + pct_employment_deprived +
                        pct_not_participating, data = simd))
```

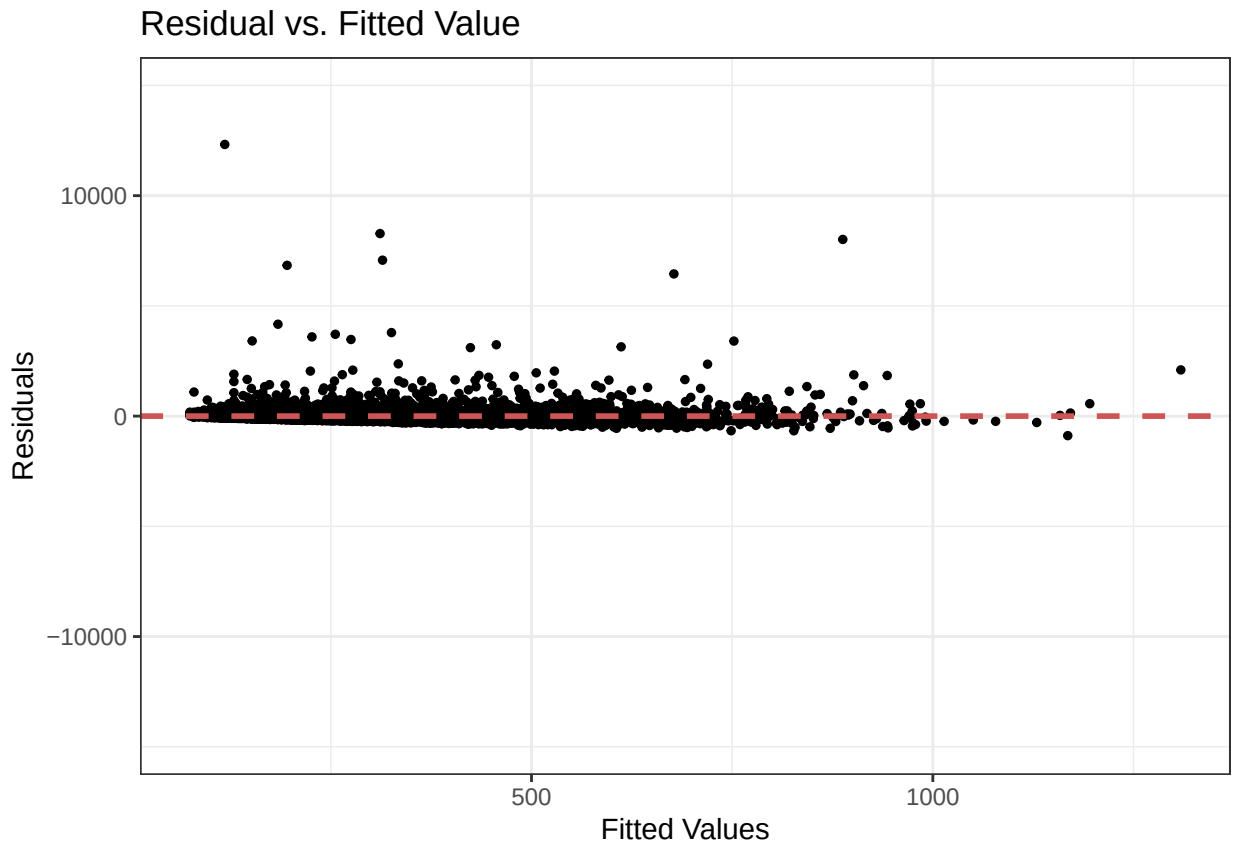
```
Call:
lm(formula = crime_rate ~ pct_income_deprived + pct_employment_deprived +
    pct_not_participating, data = simd)

Residuals:
    Min      1Q  Median      3Q     Max
 -886.4  -134.9   -59.5    45.2 12322.7

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

```
(Intercept)               84.748      8.128  10.426  < 2e-16 ***
pct_income_deprived       -5.179      1.758  -2.945  0.00324 **
pct_employment_deprived   21.746      2.382   9.128  < 2e-16 ***
pct_not_participating     19.348      1.412  13.706  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 377.6 on 6470 degrees of freedom
  (502 observations deleted due to missingness)
Multiple R-squared:  0.1592,    Adjusted R-squared:  0.1588
F-statistic: 408.4 on 3 and 6470 DF,  p-value: < 2.2e-16
```

Exercise 1.a - Functional Form

The first test is to plot the residuals and fitted values of our model.

```
library(lindia)

gg_resfitted(model.1) +
  theme_bw()
```



We see there are some massive positive residuals and thus we likely violate functional form.

Next, we'll use the `resettest()` function from the `lmtest` package for the Ramsey RESET test.

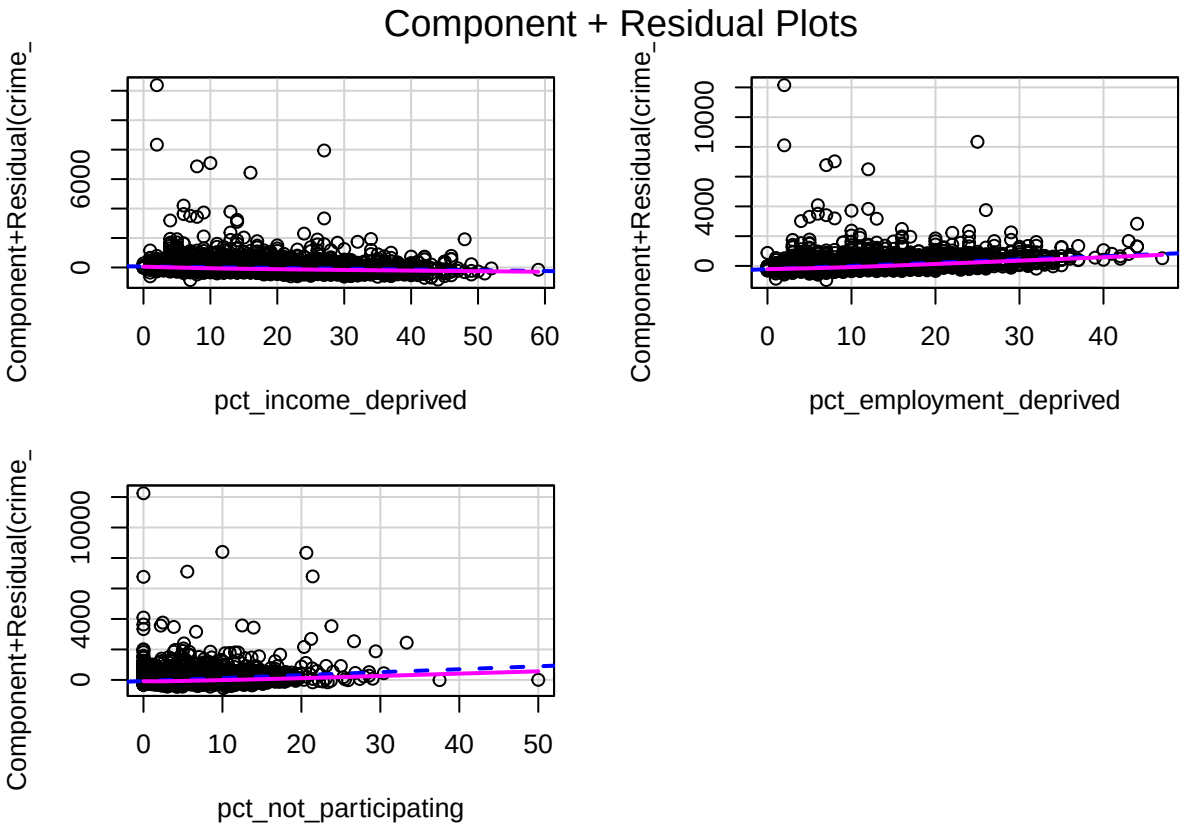
```
library(lmtest)

resettest(model.1, power = 2:3, type = "fitted")
```

```
    RESET test

data:  model.1
RESET = 21.149, df1 = 2, df2 = 6468, p-value = 7.002e-10
```

We see that $p \leq 0.05$, we reject the null, and conclude that we do indeed violate the assumption of correct functional form; as suggested by the previous plot.

We'll use a component-plus-residual plot to check for non-linearity in the predictors.

```
library(car)

crPlots(model.1)
```



It's rather difficult to see where the problems might be using these plots. If we zoom into or maximise the plots, we'll see the estimated `pct_not_participating` appears off from the hypothetical linear relationship. However, the massive positive residuals may be hiding non-linearity in the plots. Let's include all three predictors in the `boxTidwell()` function.

Since all three predictors have at least one zero, we'll add `+1` to each. We will include the option `max.iter = 100` to increase the number of iterations that are used in the MLE process.

```
boxTidwell(crime_rate ~ I(pct_income_deprived + 1) + I(pct_employment_deprived + 1) +
             I(pct_not_participating + 1), data = simd, max.iter = 100)
```

```
                                MLE of lambda Score Statistic (z) Pr(>|z|)
I(pct_income_deprived + 1)             2.6030             -3.2648 0.001095 **
I(pct_employment_deprived + 1)         1.6167              3.8482 0.000119 ***
```

```
I(pct_not_participating + 1)          1.5751             7.0108 2.37e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

iterations =  36
```

Since $p \leq 0.05$ for all three predictors, we need to transform each one based on the corresponding `MLE of lambda` value.

```
boxTidwell(crime_rate ~ I((pct_income_deprived + 1)^2.6) +
             I((pct_employment_deprived + 1)^1.62) +
             I((pct_not_participating + 1)^1.58), data = simd, max.iter=100)
```

```
                                  MLE of lambda Score Statistic (z)
I((pct_income_deprived + 1)^2.6)            1.00055             -0.0048
I((pct_employment_deprived + 1)^1.62)       0.99790             -0.0122
I((pct_not_participating + 1)^1.58)         0.99665             -0.0604
                                  Pr(>|z|)
I((pct_income_deprived + 1)^2.6)        0.9962
I((pct_employment_deprived + 1)^1.62)   0.9903
I((pct_not_participating + 1)^1.58)     0.9518

iterations =  4
```

These transformations fix the linearity problem.

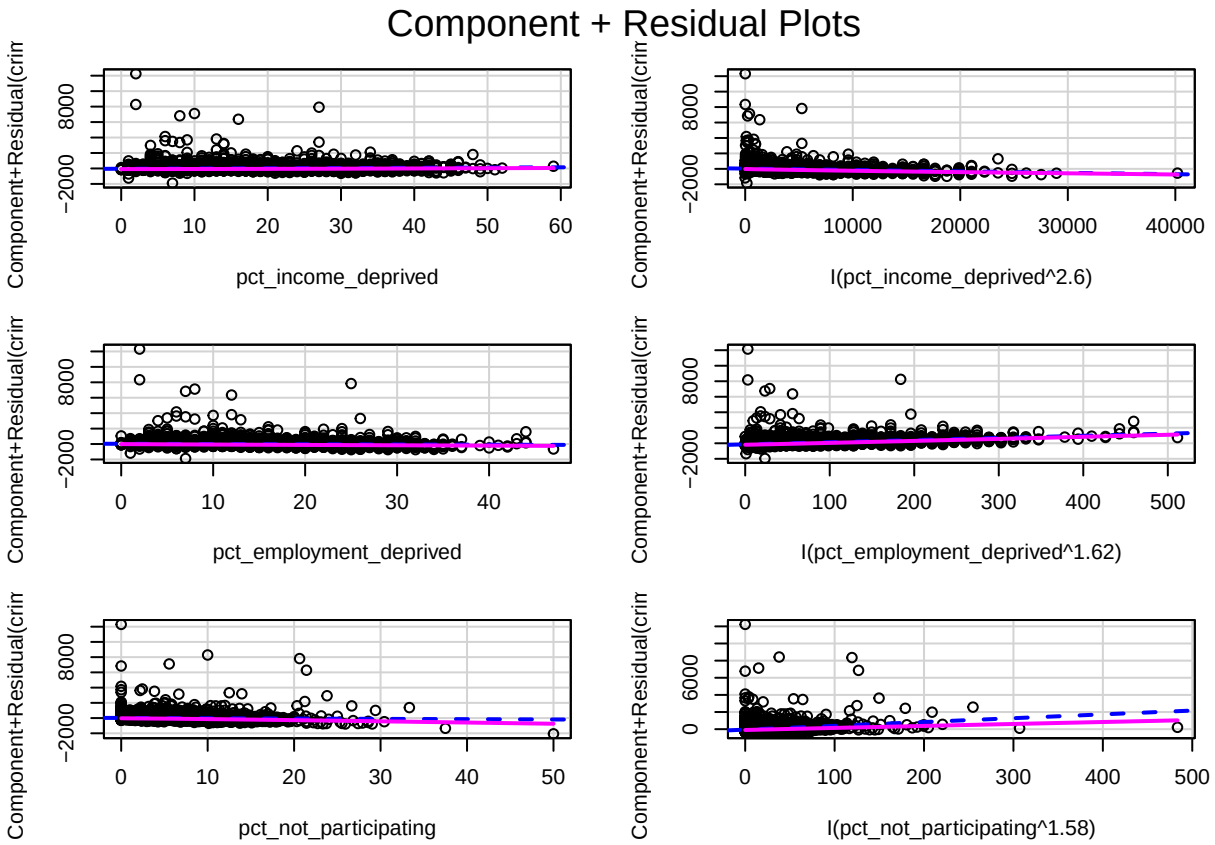
Now, let's include the transformations, using the `I()` function, in a new regression model and save the results as `model.1a`.

```
summary(model.1a <- lm(crime_rate ~ pct_income_deprived +
                         I(pct_income_deprived^2.6) + pct_employment_deprived +
                         I(pct_employment_deprived^1.62) + pct_not_participating +
                         I(pct_not_participating^1.58), data = simd))
```

```
Call:
lm(formula = crime_rate ~ pct_income_deprived + I(pct_income_deprived^2.6) +
    pct_employment_deprived + I(pct_employment_deprived^1.62) +
    pct_not_participating + I(pct_not_participating^1.58), data = simd)

Residuals:
    Min      1Q  Median      3Q     Max
-1884.3  -127.0   -63.2    40.8 12284.9

Coefficients:
                               Estimate Std. Error t value Pr(>|t|)
(Intercept)                  146.84782   14.12777  10.394  < 2e-16 ***
pct_income_deprived            3.16711    2.91009   1.088 0.276494
I(pct_income_deprived^2.6)    -0.01808    0.00666  -2.714 0.006663 **
pct_employment_deprived       -2.91473    7.00425  -0.416 0.677323
I(pct_employment_deprived^1.62)  2.77553  0.75832   3.660 0.000254 ***
pct_not_participating         -3.98634    4.13321  -0.964 0.334848
I(pct_not_participating^1.58)  4.45652    0.74485   5.983 2.31e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 376 on 6467 degrees of freedom
```

```
  (502 observations deleted due to missingness)
Multiple R-squared:  0.1669,    Adjusted R-squared:  0.1661
F-statistic: 215.9 on 6 and 6467 DF,  p-value: < 2.2e-16
```

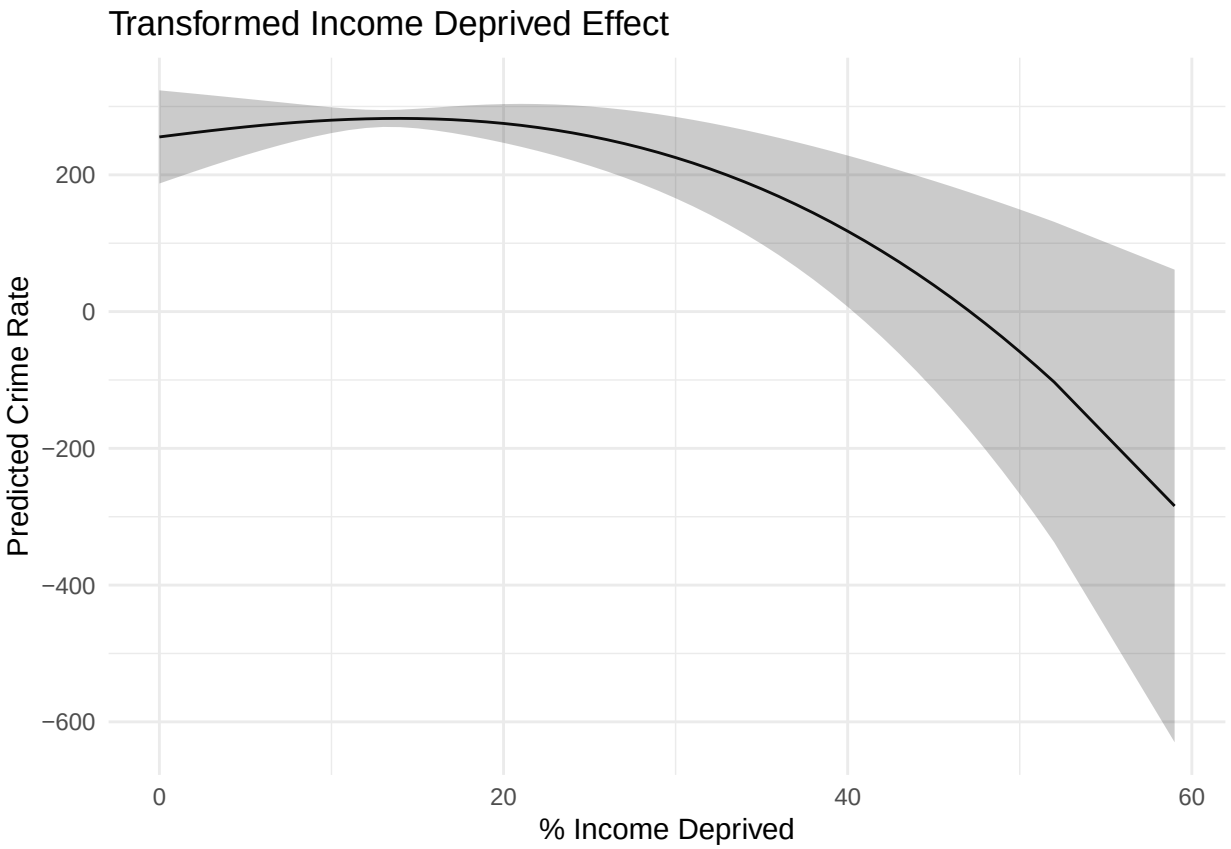
```
crPlots(model.1a)
```



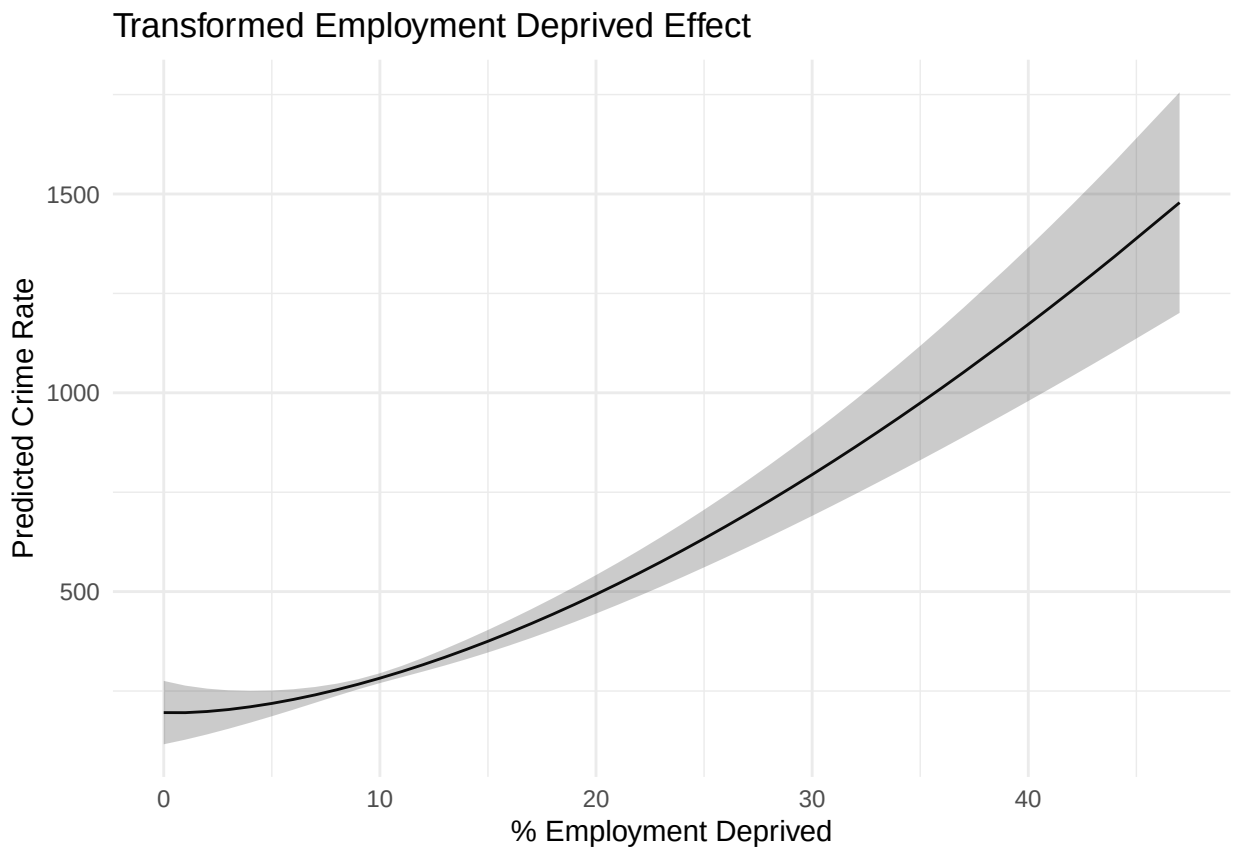
Although the Box-Tidwell test tells us everything is good, these plots don't appear to be an improvement over the original model. Again, the massive positive residuals might be the reason for the lack of clarity.

To make sense of the transformed predictors, we can plot the effects using the `ggpredict()` function from the `ggeffects` library.

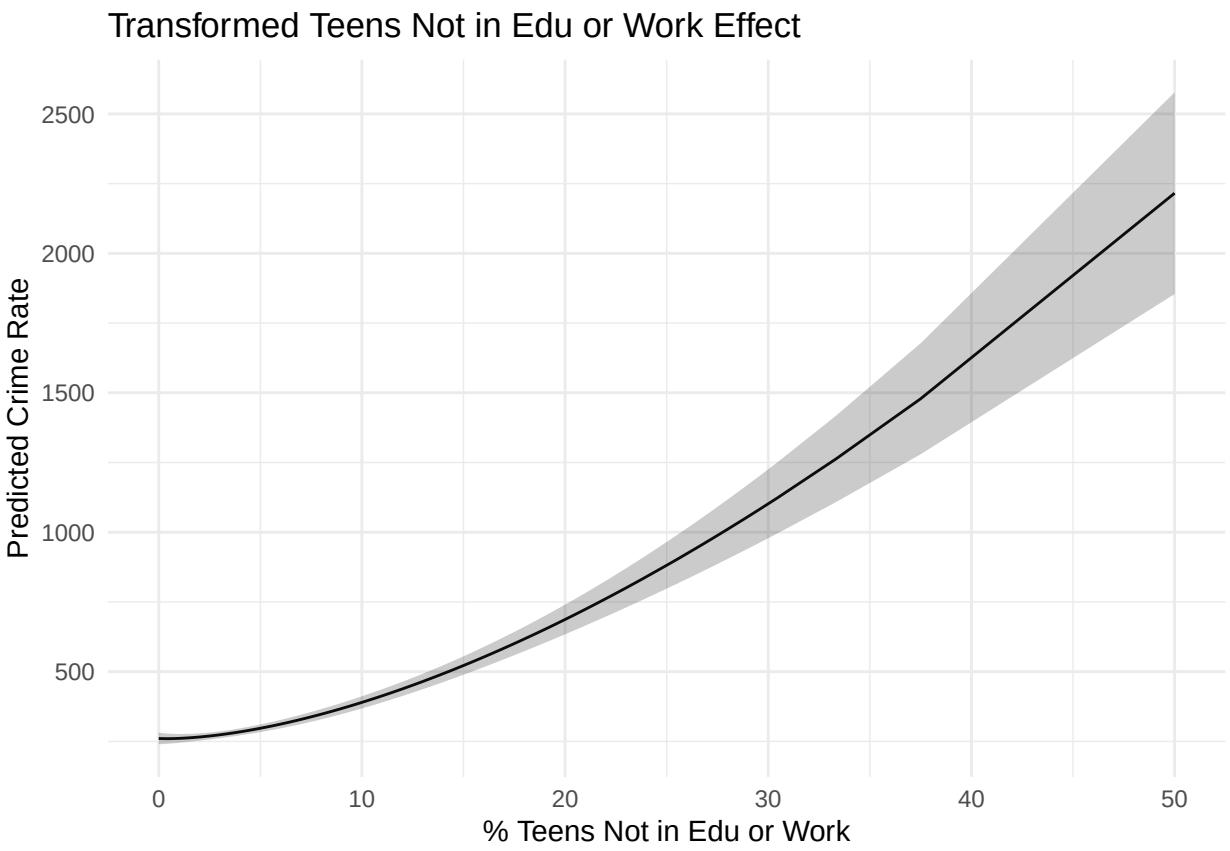
```
library(ggeffects)

ggpredict(model.1a, terms = "pct_income_deprived") %>%
ggplot(aes(x = x, y = predicted)) +
  geom_line() +
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25) +
  labs(title = "Transformed Income Deprived Effect",
       x = "% Income Deprived",
       y="Predicted Crime Rate") +
  theme_minimal()
```




```
ggpredict(model.1a, terms = "pct_employment_deprived") %>%
ggplot(aes(x = x, y = predicted)) +
  geom_line() +
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25) +
  labs(title = "Transformed Employment Deprived Effect",
       x = "% Employment Deprived",
       y="Predicted Crime Rate") +
  theme_minimal()
```



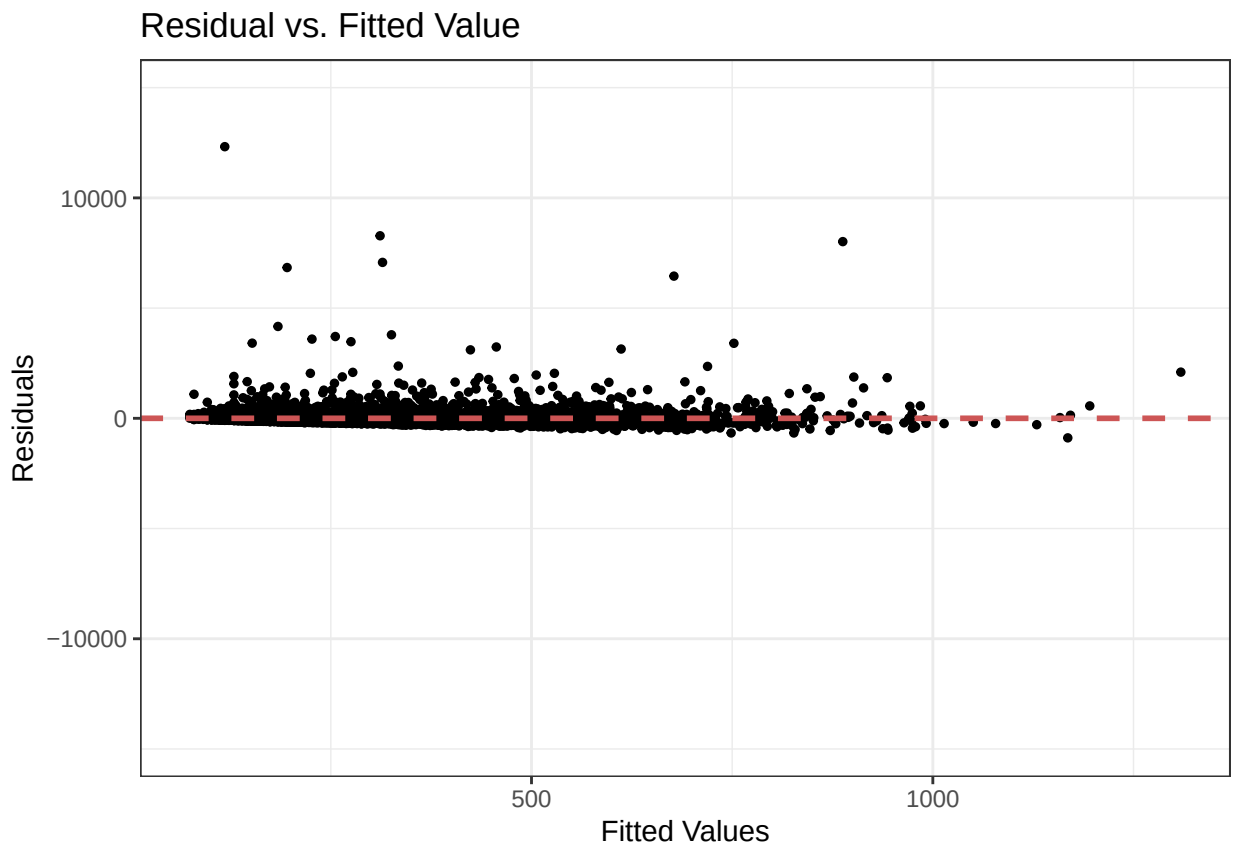

```
ggpredict(model.1a, terms = "pct_not_participating") %>%
ggplot(aes(x = x, y = predicted)) +
  geom_line() +
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25) +
  labs(title = "Transformed Teens Not in Edu or Work Effect",
       x = "% Teens Not in Edu or Work",
       y="Predicted Crime Rate") +
  theme_minimal()
```



Exercise 1.b - Heteroscedasticity

The first test is to plot the residuals and fitted values of our model.

```
gg_resfitted(model.1) +
  theme_bw()
```



It is somewhat difficult to tell if there's a problem with heteroscedasticity given the massive positive residuals.

Next, we'll use the Breusch-Pagan test for determining whether heteroscedasticity exists. We will use the `bptest()` function from the `lmtest` package.

```
bptest(model.1, studentize = FALSE)
```

```
    Breusch-Pagan test

data:  model.1
BP = 3140.7, df = 3, p-value < 2.2e-16
```

We see that the p-value is below 0.05, thus we reject the null and conclude that we have heteroscedasticity.

To correct for heteroscedasticity, we will re-run our regression model with robust standard errors using the `coeftest()` function from the `lmtest` package. We also need to load the `sandwich` package to use the `vcovHC` option.

```
library(sandwich)

coeftest(model.1, vcov = vcovHC)
```

```
t test of coefficients:

                     Estimate Std. Error t value  Pr(>|t|)
(Intercept)           84.7481     9.1764  9.2354 < 2.2e-16 ***
pct_income_deprived   -5.1785     1.9756 -2.6213   0.00878 **
```

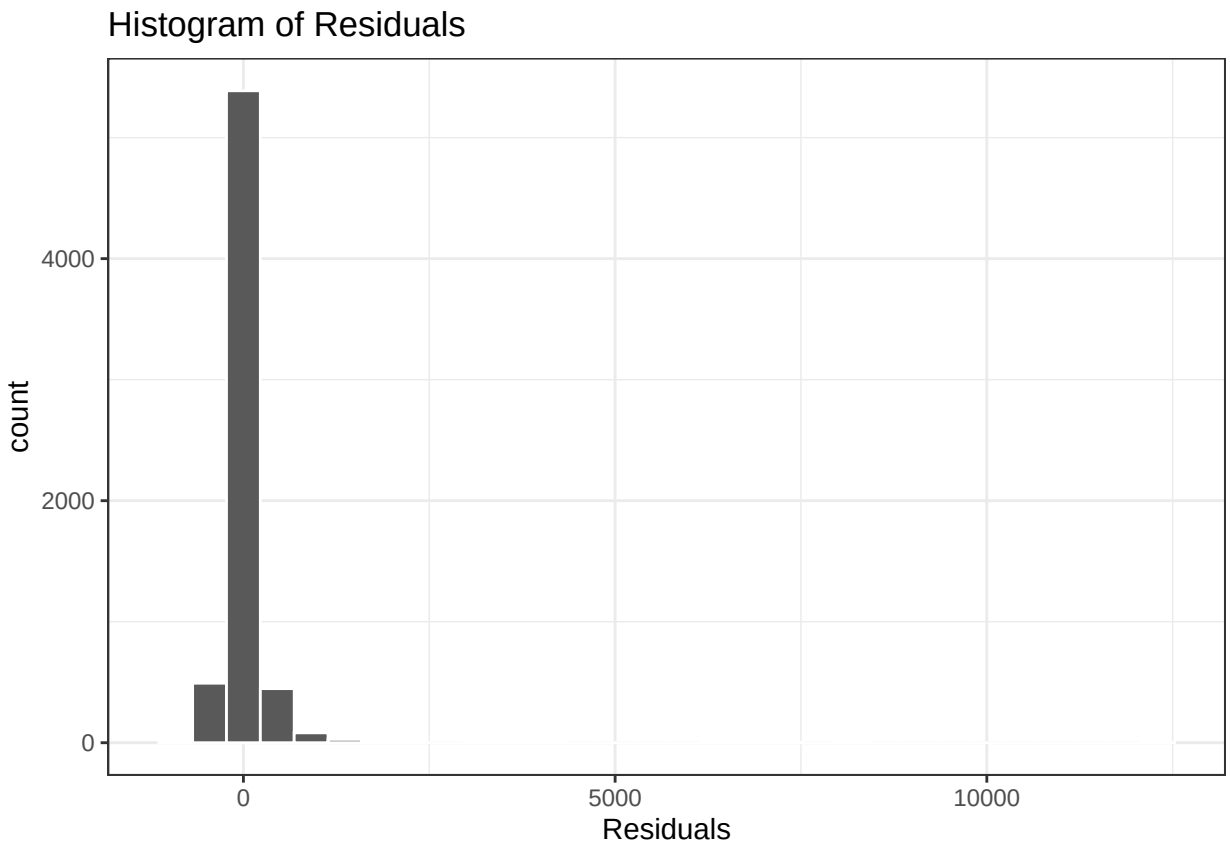
```
pct_employment_deprived  21.7463     2.7183  8.0000 1.463e-15 ***
pct_not_participating    19.3480     3.1378  6.1662 7.417e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that all three predictors are still statistically significant when using robust standard errors.

Exercise 1.c - Normality

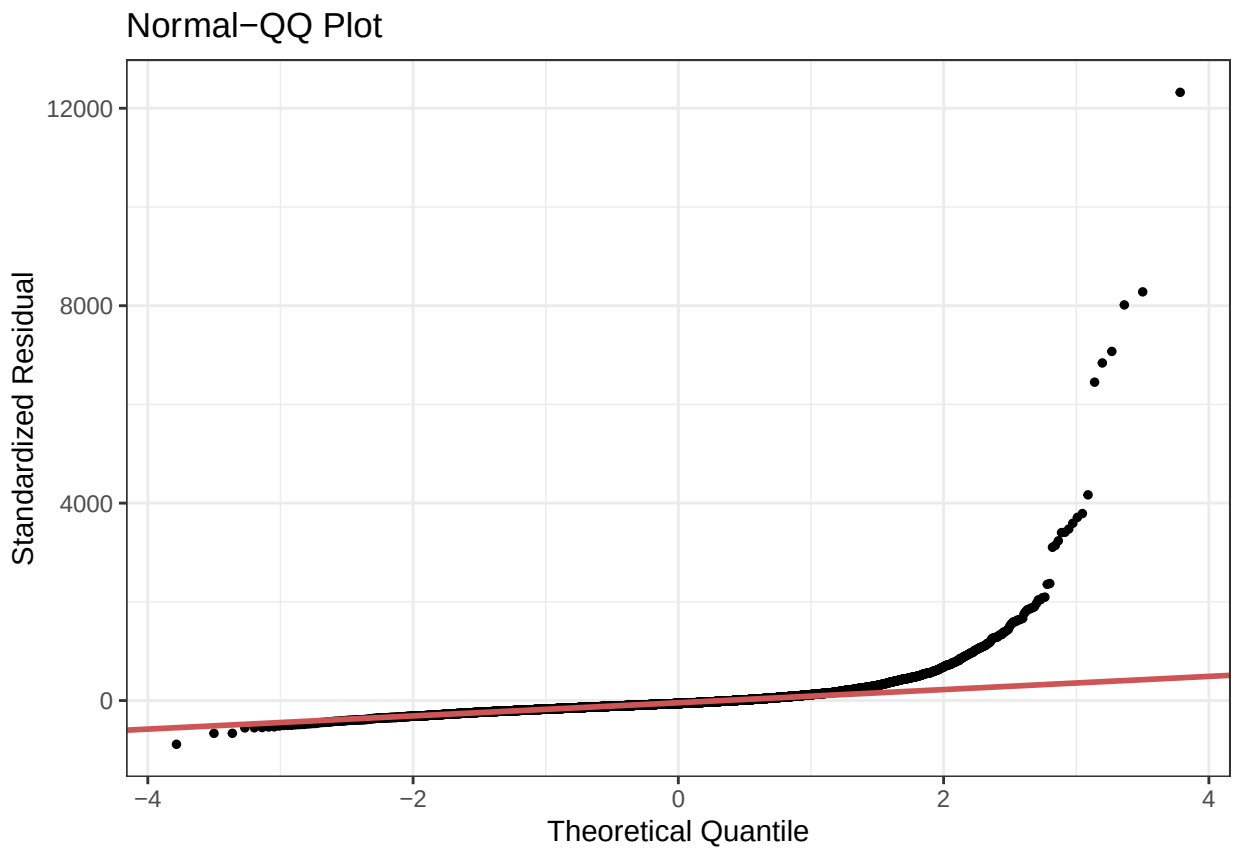
First, we will plot a histogram of the residuals for `model.1` using the `gg_rehist()` function from the `lindia` package.

```
gg_reshist(model.1) +
  theme_bw()
```



Yeah, this is essentially worthless; again, the massive positive residuals.

```
gg_qqplot(model.1) +
  theme_bw()
```



Same as above.

```
library(nortest)

ad.test(model.1$residuals)
```

```
    Anderson-Darling normality test

data:  model.1$residuals
A = 730.2, p-value < 2.2e-16
```

We see that $p \leq 0.05$, we reject the null, and thus we cannot assume our residuals are normally distributed.

We can use a Box-Cox transformation (with the `powerTransform()` function) to figure out whether and what value to transform the outcome variable.

```
summary(powerTransform(model.1))
```

```
bcPower Transformation to Normality
   Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
Y1   -0.0319       -0.03      -0.0527      -0.0111

Likelihood ratio test that transformation parameter is equal to 0
 (log transformation)
                           LRT df      pval
LR test, lambda = (0) 9.116675  1 0.0025329

Likelihood ratio test that no transformation is needed
```

```
                           LRT df       pval
LR test, lambda = (1) 12634.93  1 < 2.22e-16
```

This shows we should transform the outcome variable by raising it to -0.03. We re-run our regression model with the transformed outcome variable (using the `I()` function). We also re-run the Box-Cox transformation to test whether non-normality is fixed.

```
summary(model.1b <- lm(I(crime_rate)^(-.03) ~ pct_income_deprived +
                         pct_employment_deprived +
                         pct_not_participating, data = simd))
```

```
Call:
lm(formula = I(crime_rate)^(-0.03) ~ pct_income_deprived + pct_employment_deprived +
    pct_not_participating, data = simd)

Residuals:
      Min        1Q    Median        3Q       Max
-0.114598 -0.010305  0.000138  0.011317  0.067073

Coefficients:
                           Estimate Std. Error  t value Pr(>|t|)
(Intercept)               8.710e-01  3.739e-04 2329.304  < 2e-16 ***
pct_income_deprived      -4.110e-04  8.089e-05   -5.082 3.84e-07 ***
pct_employment_deprived -9.933e-04  1.096e-04   -9.062  < 2e-16 ***
pct_not_participating   -9.474e-04  6.494e-05  -14.589  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01737 on 6470 degrees of freedom
  (502 observations deleted due to missingness)
Multiple R-squared:  0.3765,    Adjusted R-squared:  0.3762
F-statistic:  1302 on 3 and 6470 DF,  p-value: < 2.2e-16
```

```
summary(powerTransform(model.1b))
```

```
bcPower Transformation to Normality
   Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
Y1     1.064           1         0.37        1.758

Likelihood ratio test that transformation parameter is equal to 0
 (log transformation)
                           LRT df      pval
LR test, lambda = (0) 9.116675  1 0.0025329

Likelihood ratio test that no transformation is needed
                             LRT df    pval
LR test, lambda = (1) 0.03271388  1 0.85647
```

We see in the `powerTransform()` results that we have fixed non-normality. However, the correction produces non-sensible regression coefficients and so we might not want to make this correction.

Exercise 1.d - Multicollinearity

We run a VIF test on `model.1`.

```
vif(model.1)
```

```
    pct_income_deprived pct_employment_deprived   pct_not_participating
              13.123299                13.203014                1.460745
```

LOOK AT THAT! Multicollinearity in the wild! We see that `pct_income_deprived` and `pct_employment_deprived` both have VIF values over 10. Let's take a quick look at their correlation.

```
cor.test(simd$pct_income_deprived,simd$pct_employment_deprived)
```

```
    Pearson's product-moment correlation

data:  simd$pct_income_deprived and simd$pct_employment_deprived
t = 294.64, df = 6971, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.9603320 0.9638232
sample estimates:
      cor
0.9621171
```

They are correlated at 0.96 - almost a perfect correlation!

We need to drop one of them from our model. In this case - where we have, arguably, substantively different but highly correlated measures - our choice will depend on our theoretical and substantive interests. In the next exercise, we'll drop `pct_income_deprived` from our model.

Exercise 1.e - Outliers, Leverage, and Influential Data Points

We already know that there are massive positive outliers in `model.1`.

We have 3 predictors and 6,474 observations (n) in `model.1`. We calculate our leverage cut-point as:

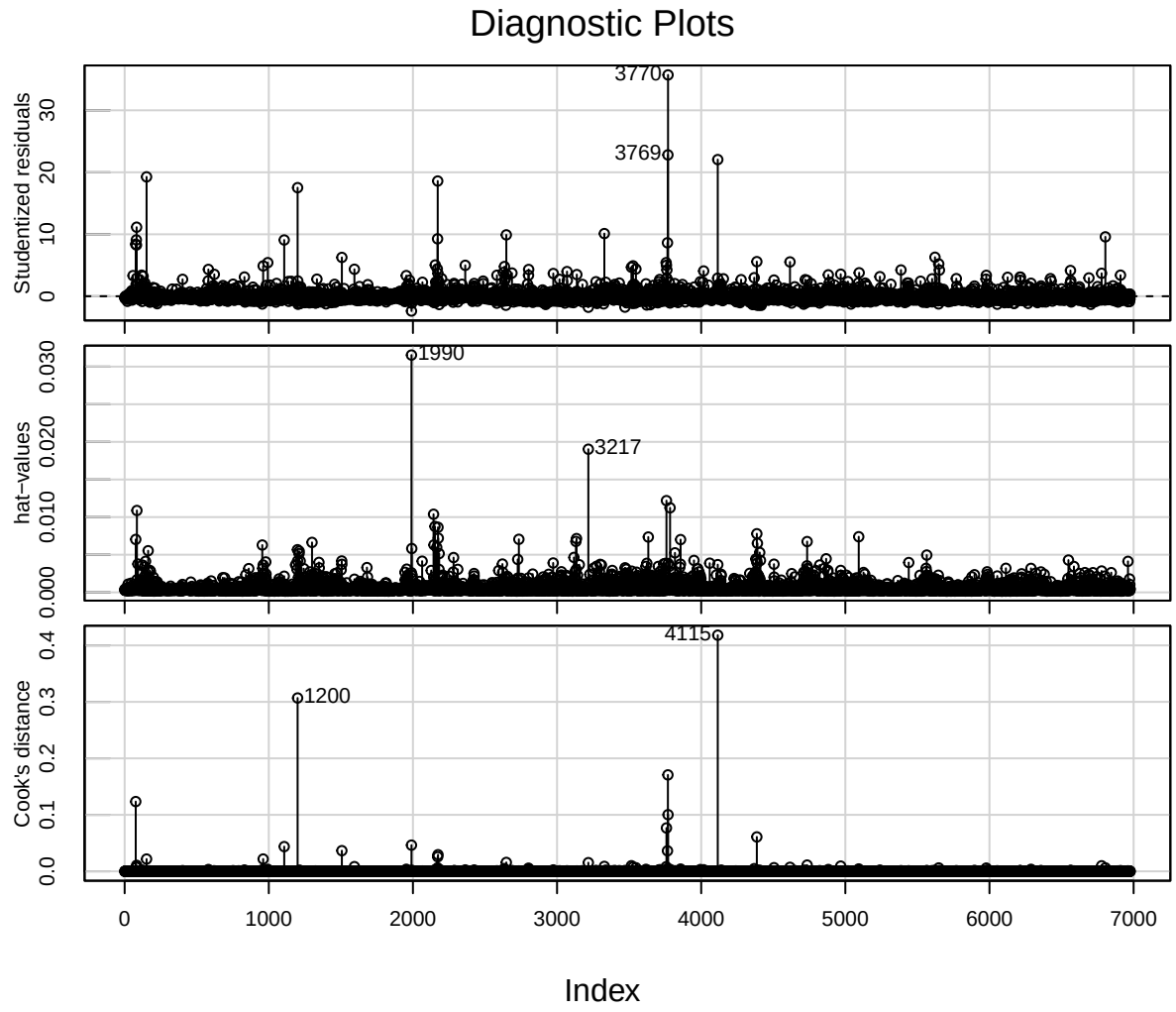
```
(2*(3+1))/6474
```

```
[1] 0.001235712
```

Thus, any data point that has a hat-value ≥ 0.0012 is considered to have high leverage.

Since we have a relatively large number of observations, we'll consider any point that has a Cook's distance greater than 1 to be influential.

```
influenceIndexPlot(model.1,
                   vars = c("Studentized","hat","Cook"))
```



We see a lot of outliers (and large outliers), points with leverage, but none of the observations are influential.

Let's take a look at observation 4115 that's flagged in the bottom plot (Cook's distance plot).

```
simd %>%
  select(Intermediate_Zone,Council_area,crime_rate,pct_income_deprived,
         pct_employment_deprived,pct_not_participating) %>%
  slice(4115)
```

```
# A tibble: 1 x 6
  Intermediate_Zone   Council_area crime_rate pct_income_depri~ pct_employment_~
  <chr>               <chr>             <dbl>             <dbl>            <dbl>
1 Inverness Central,~ Highland          8904.                27               25
# ... with 1 more variable: pct_not_participating <dbl>
```

This datazone is in Inverness (the main city in the Highlands) and has very high values across all our variables. We usually think of datazones in Glasgow and Edinburgh having such characteristics and not areas of the Highlands. We probably would want to investigate this further if we were conducting a proper study.

Let's also take a look at observation 3770 that's flagged in the top plot (Studentized residuals plot).

```
simd %>%
  select(Intermediate_Zone,Council_area,crime_rate,pct_income_deprived,
         pct_employment_deprived,pct_not_participating,Total_population) %>%
  slice(3770)
```

```
# A tibble: 1 x 7
  Intermediate_Zone Council_area crime_rate pct_income_deprived pct_employment_~
  <chr>             <chr>             <dbl>               <dbl>            <dbl>
1 City Centre South Glasgow City     12441.                   2                2
# ... with 2 more variables: pct_not_participating <dbl>,
#   Total_population <dbl>
```

This datazone has the highest crime rate of any datazone, but also has very low values on our predictors. How? The datazone is part of Glasgow City Centre - an area with many businesses, restaurants, pubs, clubs, etc., that attract visitors and, thus, crime. However, the people that actually live in the area appear to have low levels of deprivation.

Exercise 2

We'll create a new tibble for the filtered values and name it `simd1`. We need to `detach` the `car` package to use the `recode()` function from `dplyr`.

```
detach("package:car", unload = TRUE)

simd1 <- simd %>%
  filter(crime_rate <= 2000 & pct_employment_deprived <= 40 &
           pct_not_participating <= 30) %>%
  mutate(urban_fct = recode(urban, `1` = "Urban", `0` = "Rural"))

summary(model.2 <- lm(crime_rate ~ pct_employment_deprived + pct_not_participating +
       urban_fct, data = simd1))
```

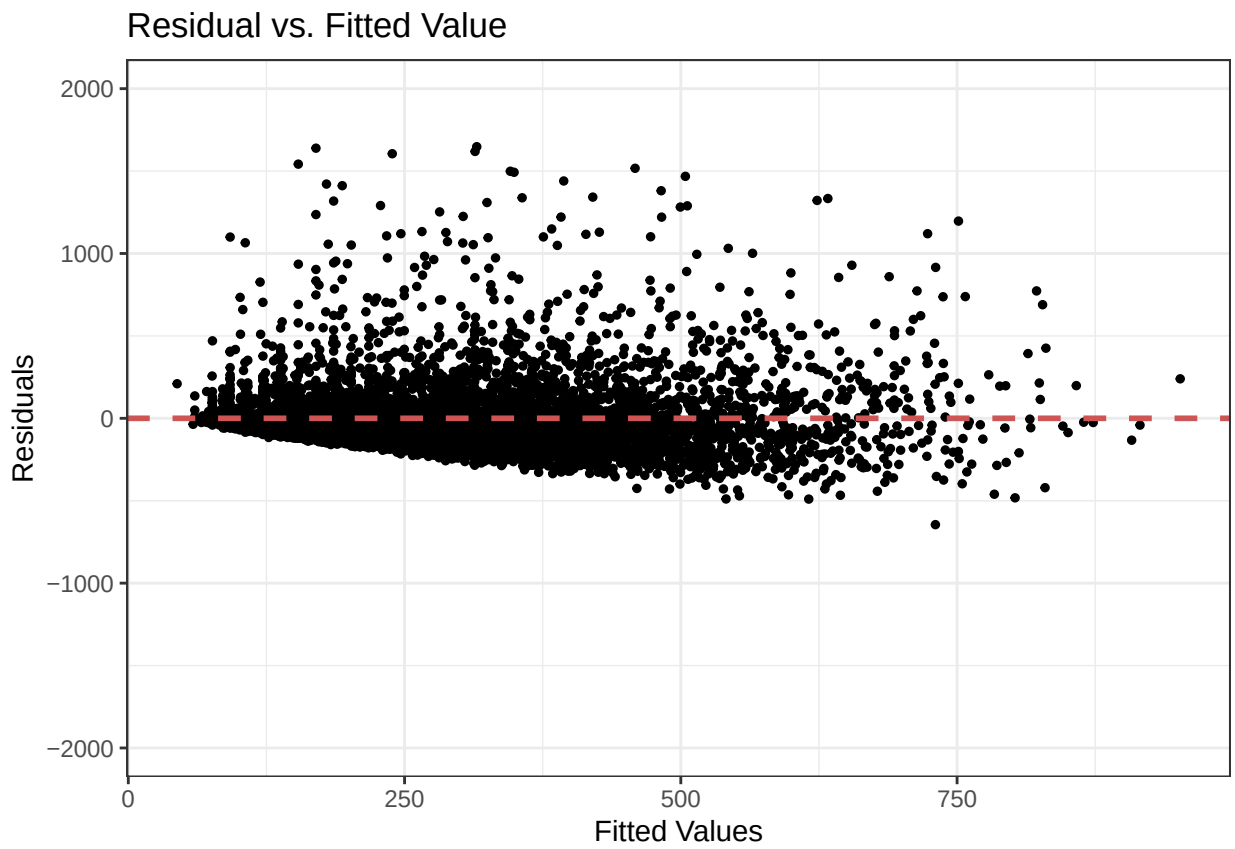
```
Call:
lm(formula = crime_rate ~ pct_employment_deprived + pct_not_participating +
    urban_fct, data = simd1)

Residuals:
    Min      1Q  Median      3Q     Max
-645.08 -113.10  -43.36   56.31 1647.73

Coefficients:
                         Estimate Std. Error t value Pr(>|t|)
(Intercept)               44.1845     5.7291   7.712 1.42e-14 ***
pct_employment_deprived   15.9909     0.4632  34.520  < 2e-16 ***
pct_not_participating     11.1436     0.8432  13.216  < 2e-16 ***
urban_fctUrban            61.6721     5.9376  10.387  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 212.7 on 6423 degrees of freedom
Multiple R-squared:  0.3327,    Adjusted R-squared:  0.3324
F-statistic:  1067 on 3 and 6423 DF,  p-value: < 2.2e-16
```

The results are roughly the same as `model.1` - the size of the coefficients are smaller (because of the filtered outcome variable) and the R^2 and adjusted R^2 values are higher.

Now let's quickly go through diagnostics.

Exercise 2.a - Functional Form

```
gg_resfitted(model.2) +
  theme_bw()
```



We see there are some large positive residuals, but it's not as extreme as in `model.1`.

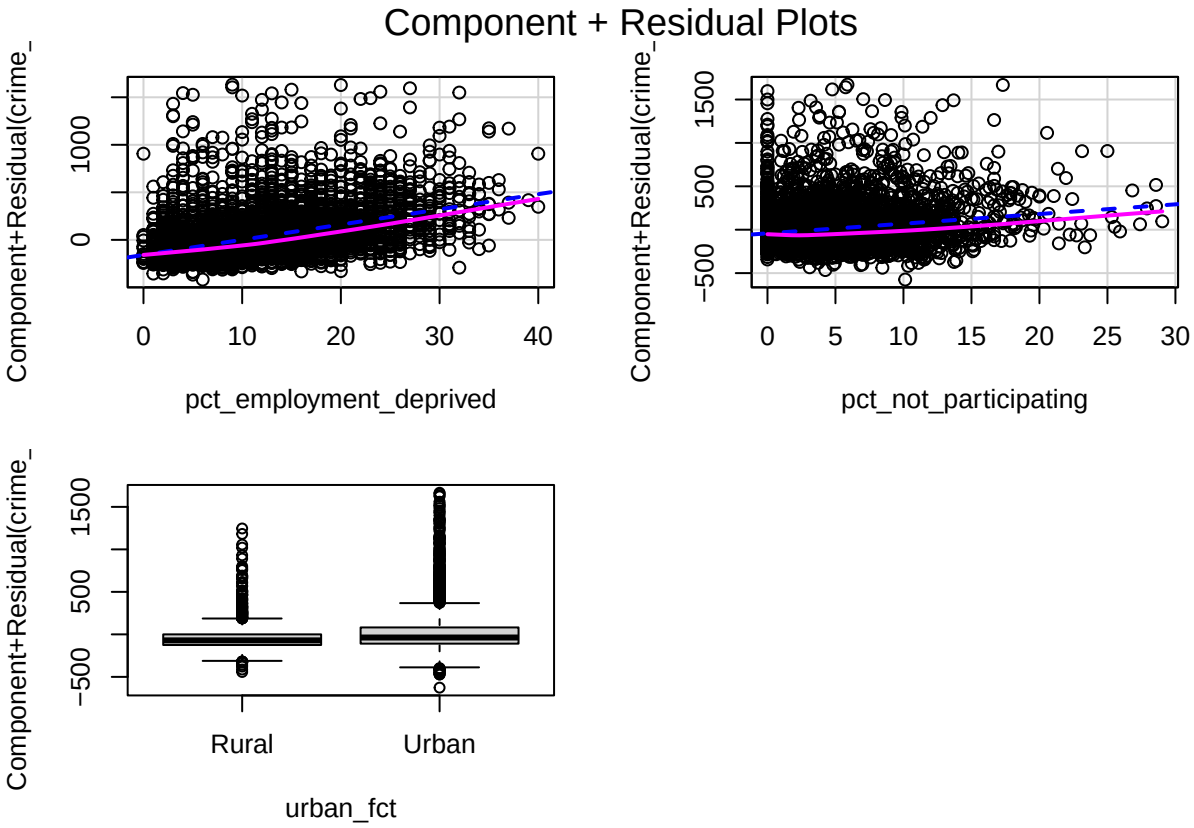
```
resettest(model.2, power = 2:3, type = "fitted")
```

```
    RESET test

data:  model.2
RESET = 11.089, df1 = 2, df2 = 6421, p-value = 1.557e-05
```

We see that $p \leq 0.05$, we reject the null, and conclude that we do indeed violate the assumption of correct functional form.

We need to re-load the `car` package for the `crPlots()` function.

```
library(car)

crPlots(model.2)
```



It's easier to see here how the estimate and theoretical linear relationships diverge for `pct_employment_deprived` and `pct_not_participating`.

```
boxTidwell(crime_rate ~ I(pct_employment_deprived + 1) + I(pct_not_participating + 1),
           ~ urban_fct, data = simd1)
```

```
                              MLE of lambda Score Statistic (z)  Pr(>|z|)
I(pct_employment_deprived + 1)        1.2207              2.4476   0.01438 *
I(pct_not_participating + 1)          1.5210              4.8996 9.604e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

iterations =  8
```

Since $p \leq 0.05$ for both predictors, we need to transform each one with the corresponding `MLE of lambda` value.

```
boxTidwell(crime_rate ~ I((pct_employment_deprived + 1)^1.22) +
            I((pct_not_participating + 1)^1.52), ~ urban_fct, data = simd1)
```

```
                                  MLE of lambda Score Statistic (z)
I((pct_employment_deprived + 1)^1.22)        1.0006              0.0097
I((pct_not_participating + 1)^1.52)          1.0007              0.0063
                                  Pr(>|z|)
I((pct_employment_deprived + 1)^1.22)   0.9923
I((pct_not_participating + 1)^1.52)     0.9950

iterations =  0
```

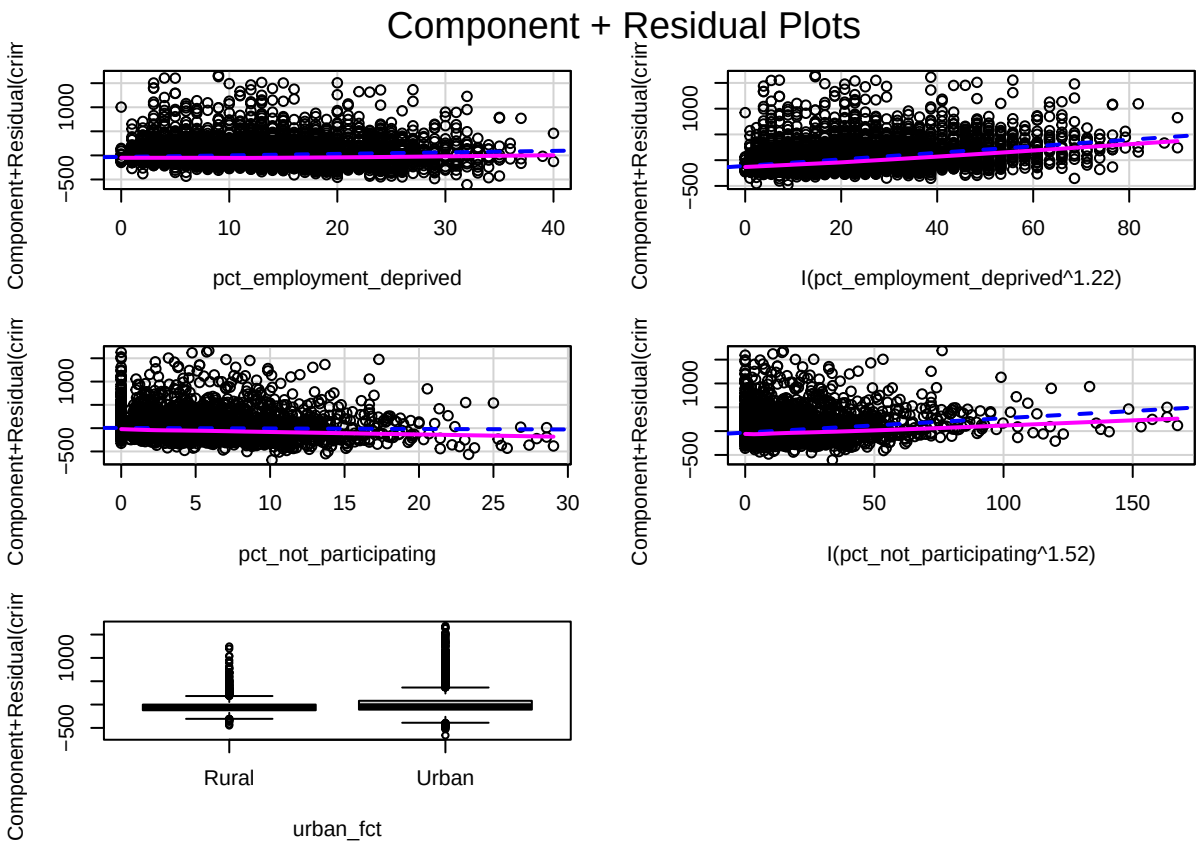
These transformations fix the linearity problem.

Now, let's include the transformations, using the `I()` function, in a new regression model and save the results as `model.2a`.

```
summary(model.2a <- lm(crime_rate ~ pct_employment_deprived +
                         I(pct_employment_deprived^1.22) +
                         pct_not_participating + I(pct_not_participating^1.52) +
                         urban_fct, data = simd1))
```

```
Call:
lm(formula = crime_rate ~ pct_employment_deprived + I(pct_employment_deprived^1.22) +
    pct_not_participating + I(pct_not_participating^1.52) + urban_fct,
    data = simd1)

Residuals:
    Min      1Q  Median      3Q     Max
-677.53 -112.21  -45.52   55.96 1667.34

Coefficients:
                               Estimate Std. Error t value Pr(>|t|)
(Intercept)                     77.1961     9.8160   7.864 4.32e-15 ***
pct_employment_deprived          3.1137     5.0208   0.620  0.53517
I(pct_employment_deprived^1.22)  6.3479     2.3545   2.696  0.00703 **
pct_not_participating           -1.0716     2.9597  -0.362  0.71730
I(pct_not_participating^1.52)    3.0198     0.6931   4.357 1.34e-05 ***
urban_fctUrban                  59.5208     5.9432  10.015  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 212.1 on 6421 degrees of freedom
Multiple R-squared:  0.3363,    Adjusted R-squared:  0.3358
F-statistic: 650.8 on 5 and 6421 DF,  p-value: < 2.2e-16
```

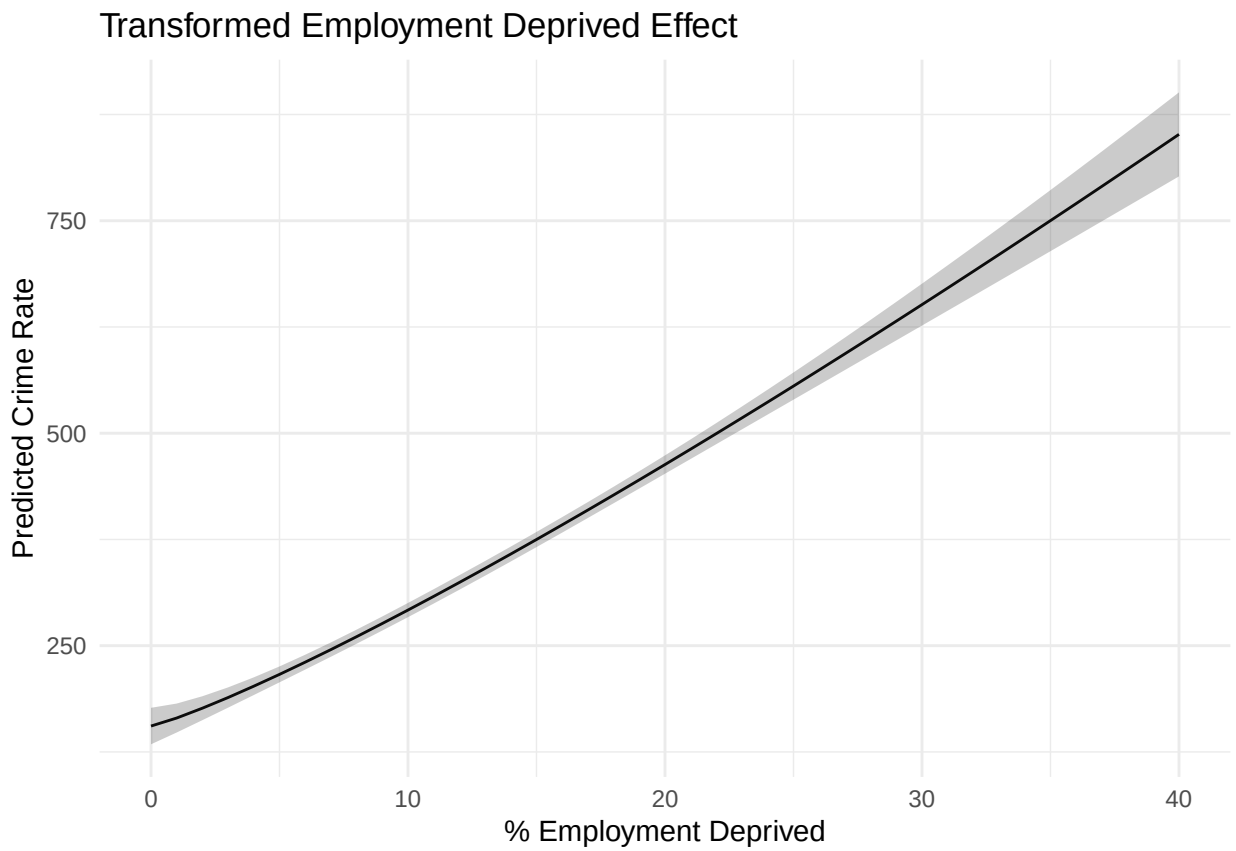
```
crPlots(model.2a)
```



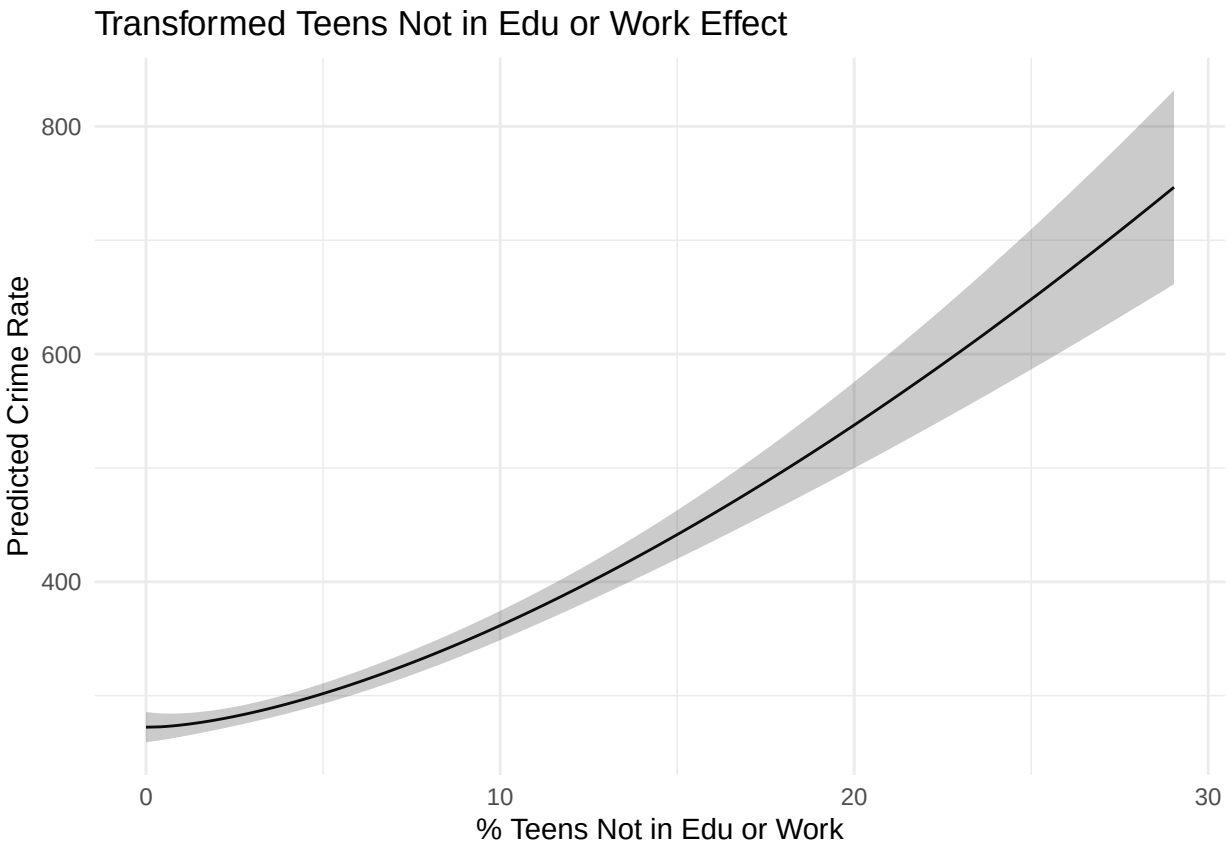
Although the Box-Tidwell test tells us everything is good, these plots don't really appear to be an improvement over the original model - just like in `model.1`.

To understand the transformed predictors, we can plot the effects using the `ggpredict()` function from the `ggeffects` library.

```
ggpredict(model.2a, terms = "pct_employment_deprived") %>%
ggplot(aes(x = x, y = predicted)) +
  geom_line() +
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25) +
  labs(title = "Transformed Employment Deprived Effect",
       x = "% Employment Deprived",
       y="Predicted Crime Rate") +
  theme_minimal()
```

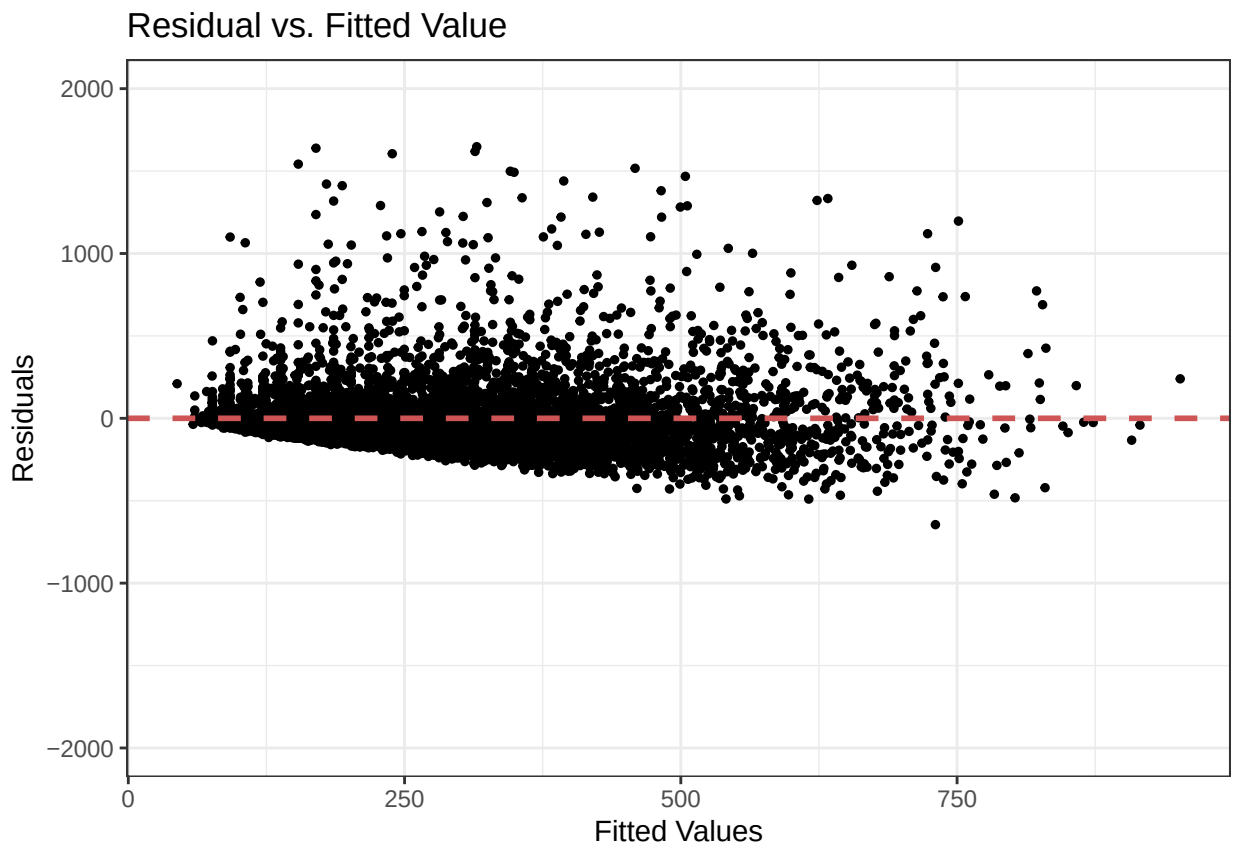



```
ggpredict(model.2a, terms = "pct_not_participating") %>%
ggplot(aes(x = x, y = predicted)) +
  geom_line() +
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25) +
  labs(title = "Transformed Teens Not in Edu or Work Effect",
       x = "% Teens Not in Edu or Work",
       y="Predicted Crime Rate") +
  theme_minimal()
```



Exercise 2.b - Heteroscedasticity

```
gg_resfitted(model.2) +
  theme_bw()
```



It is fairly clear that we have a problem with heteroscedasticity.

```
bptest(model.2, studentize = FALSE)
```

```
    Breusch-Pagan test

data:  model.2
BP = 911.07, df = 3, p-value < 2.2e-16
```

We see that the p-value is below 0.05, thus we reject the null and conclude that we have heteroscedasticity.

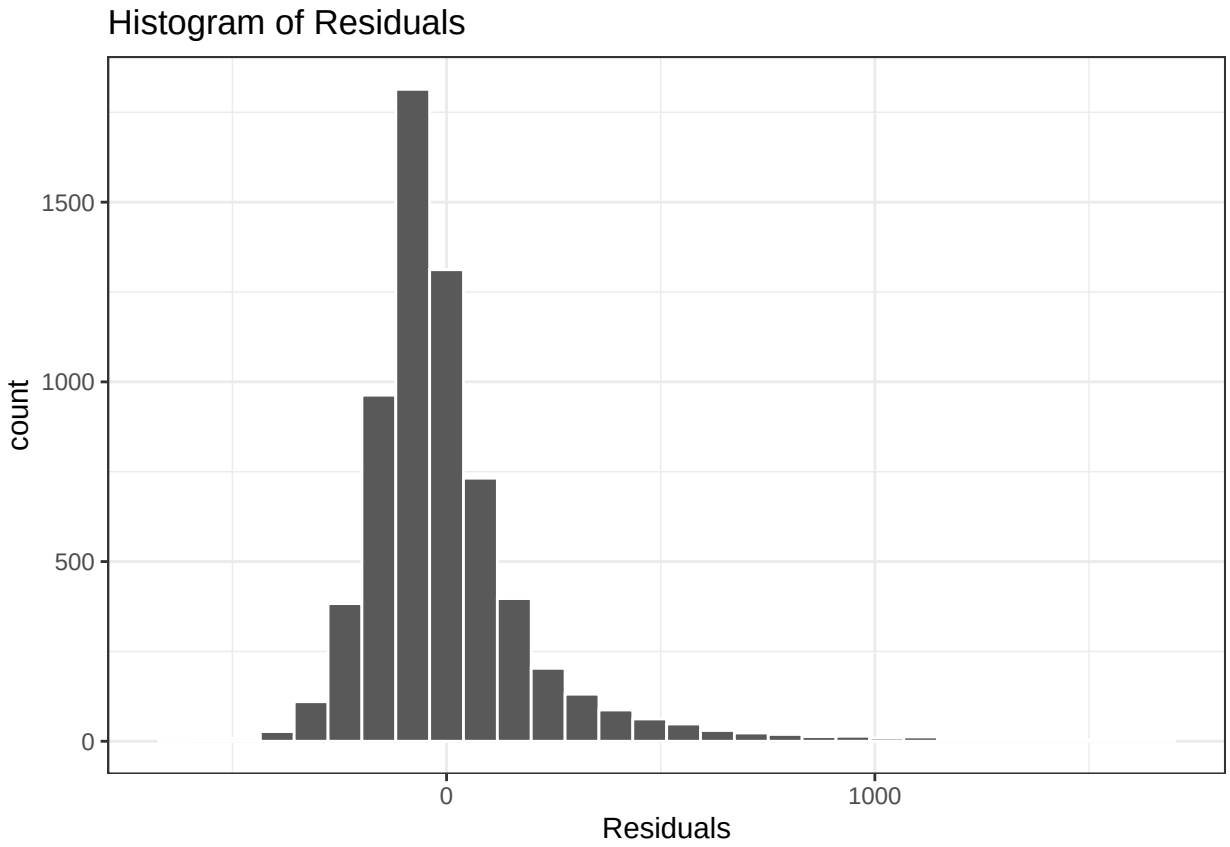
```
coeftest(model.2, vcov = vcovHC)
```

```
t test of coefficients:

                          Estimate Std. Error t value  Pr(>|t|)
(Intercept)               44.18454    4.64488  9.5125 < 2.2e-16 ***
pct_employment_deprived 15.99089    0.56188 28.4597 < 2.2e-16 ***
pct_not_participating   11.14360    1.09045 10.2193 < 2.2e-16 ***
urban_fctUrban          61.67209    4.93723 12.4912 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that all three predictors are still statistically significant when using robust standard errors.

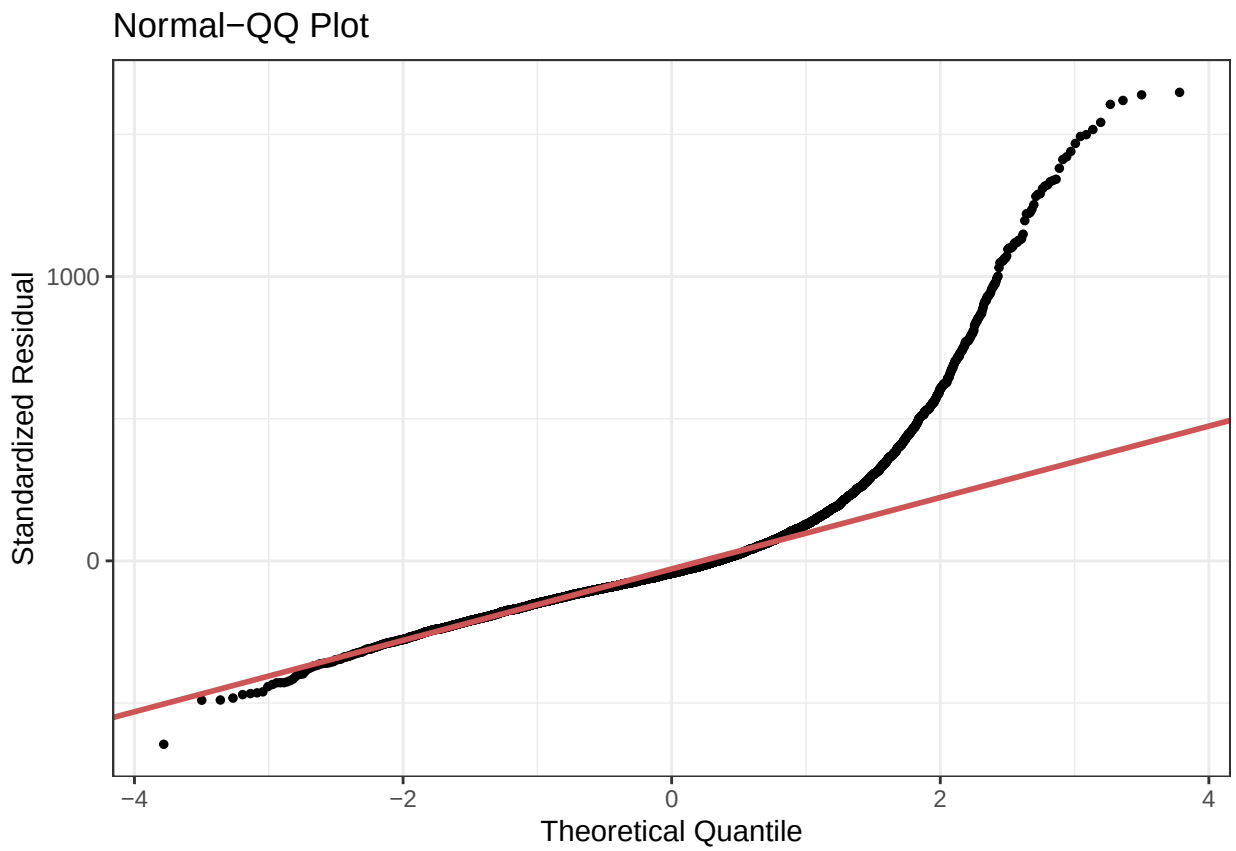
Exercise 2.c - Normality

```
gg_reshist(model.2) +
  theme_bw()
```



Although we filtered the extreme points in our variables, this histogram still shows a skewed distribution.

```
gg_qqplot(model.2) +
  theme_bw()
```



Same as above.

```
ad.test(model.2$residuals)
```

```
    Anderson-Darling normality test

data:  model.2$residuals
A = 298.4, p-value < 2.2e-16
```

We see that $p \leq 0.05$, we reject the null, and thus we cannot assume our residuals are normally distributed.

```
summary(powerTransform(model.2))
```

```
bcPower Transformation to Normality
   Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
Y1    0.0552        0.06       0.0317       0.0787

Likelihood ratio test that transformation parameter is equal to 0
 (log transformation)
                           LRT df      pval
LR test, lambda = (0) 21.06959  1 4.429e-06

Likelihood ratio test that no transformation is needed
                           LRT df       pval
LR test, lambda = (1) 6050.282  1 < 2.22e-16
```

This shows we should transform the outcome variable by raising it to 0.06. We re-run our regression model and make the transformation using the `I()` function. We also re-run the Box-Cox transformation to test

whether non-normality is fixed.

```
summary(model.2b <- lm(I(crime_rate)^(.06) ~ pct_employment_deprived +
                         pct_not_participating +
                         urban_fct, data = simd1))
```

```
Call:
lm(formula = I(crime_rate)^(0.06) ~ pct_employment_deprived +
    pct_not_participating + urban_fct, data = simd1)

Residuals:
      Min        1Q    Median        3Q       Max
-0.202765 -0.036204 -0.001359  0.033399  0.223318

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              1.3055340  0.0014371  908.45   <2e-16 ***
pct_employment_deprived 0.0048381  0.0001162   41.64   <2e-16 ***
pct_not_participating   0.0027684  0.0002115   13.09   <2e-16 ***
urban_fctUrban          0.0201431  0.0014894   13.52   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.05335 on 6423 degrees of freedom
Multiple R-squared:  0.4047,    Adjusted R-squared:  0.4045
F-statistic:  1456 on 3 and 6423 DF,  p-value: < 2.2e-16
```

```
summary(powerTransform(model.2b))
```

```
bcPower Transformation to Normality
   Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
Y1    0.9201           1        0.528       1.3121

Likelihood ratio test that transformation parameter is equal to 0
 (log transformation)
                            LRT df      pval
LR test, lambda = (0) 21.06959  1 4.429e-06

Likelihood ratio test that no transformation is needed
                            LRT df    pval
LR test, lambda = (1) 0.159723  1 0.68941
```

We see in the `powerTransform()` results that we have fixed non-normality. Again, though, the correction produces non-sensible regression coefficients and so we might not want to make this correction.

Exercise 2.d - Multicollinearity

```
vif(model.2)
```

```
pct_employment_deprived   pct_not_participating               urban_fct
               1.514111                1.497793                1.045149
```

This time the VIF values are not near 10 and thus we don't have multicollinearity.

Exercise 2.e - Outliers, Leverage, and Influential Data Points

Even with filtering the variables, we know there are large positive outliers in `model.2`.

We have 3 predictors and 6,423 observations (n) in `model.2`. We calculate our leverage cut-point as:

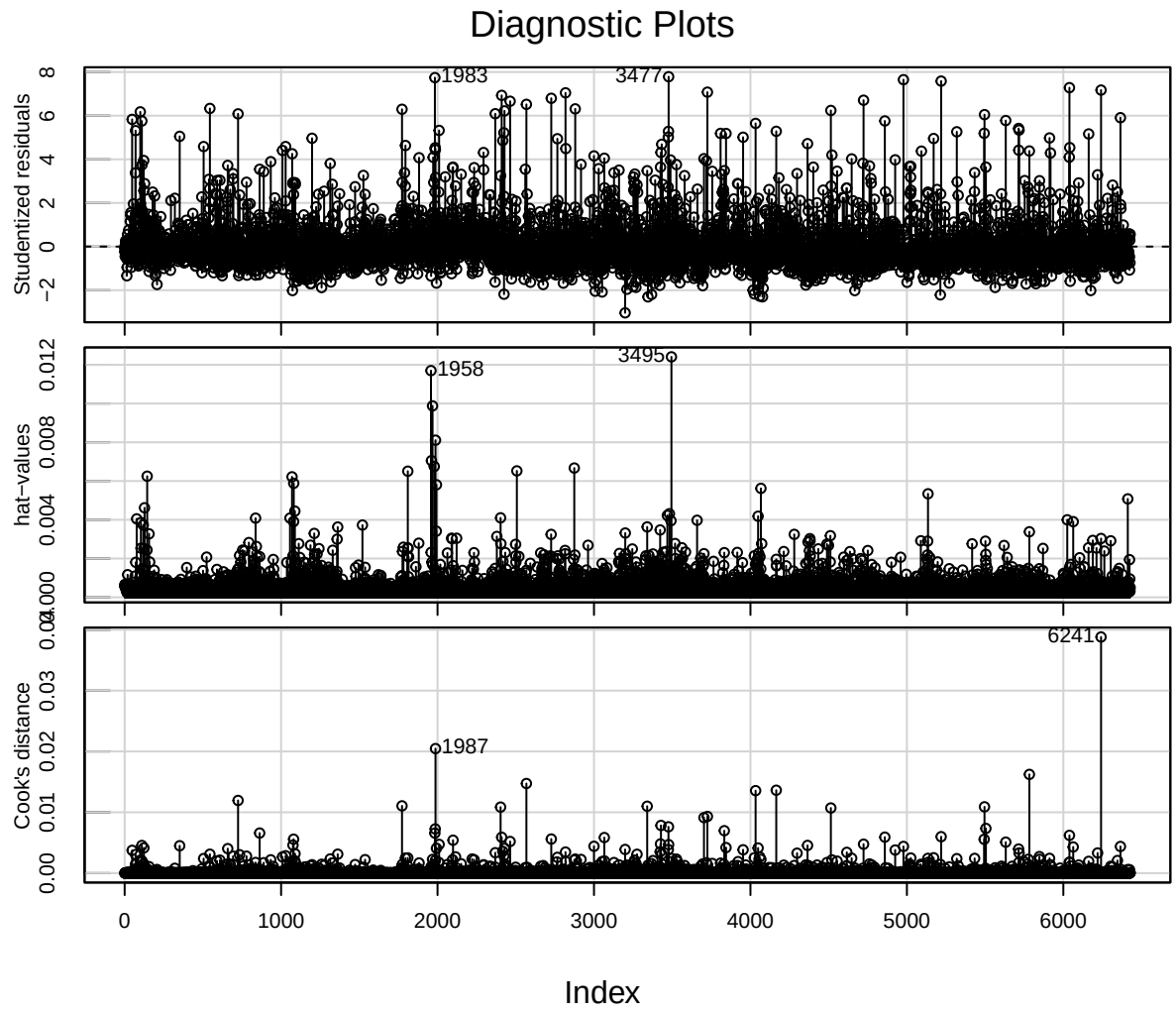
```
(2*(3+1))/6423
```

```
[1] 0.001245524
```

Thus, any data point that has a hat-value ≥ 0.0012 is considered to have high leverage.

Since we have a relatively large number of observations, we'll consider any point that has a Cook's distance greater than 1 to be influential.

```
influenceIndexPlot(model.2,
                   vars = c("Studentized","hat","Cook"))
```



As with `model.1`, we see a lot of outliers (and large outliers), points with leverage, but none of the observations are influential.

Let's take a look at observation 6241 that's flagged in the bottom plot (Cook's distance plot).

```
simd1 %>%
  select(Intermediate_Zone,Council_area,crime_rate,pct_employment_deprived,
         pct_not_participating,urban_fct) %>%
  slice(6241)
```

```
# A tibble: 1 x 6
  Intermediate_Zone Council_area crime_rate pct_employment_de~ pct_not_particip~
  <chr>             <chr>             <dbl>              <dbl>             <dbl>
1 Dedridge East     West Lothian      1976.                 10              17.3
# ... with 1 more variable: urban_fct <chr>
```

This datazone is in West Lothian - west of Edinburgh and east of Glasgow - and part of Livingston. The datazone has a very high crime rate and percentage of teens not in education, training, or work, and a slightly above average percentage of employment deprivation.

Let's also take a look at observation 3477 that's flagged in the top plot (Studentized residuals plot).

```
simd1 %>%
  select(Intermediate_Zone,Council_area,crime_rate,pct_employment_deprived,
         pct_not_participating,urban_fct) %>%
  slice(3477)
```

```
# A tibble: 1 x 6
  Intermediate_Zone Council_area crime_rate pct_employment_de~ pct_not_particip~
  <chr>             <chr>             <dbl>              <dbl>             <dbl>
1 City Centre West  Glasgow City      1963.                  9              5.88
# ... with 1 more variable: urban_fct <chr>
```

This datazone has a very high crime rate, slightly below average percentage of employment deprivation, and slightly above average percentage of teens not in education, training, or work. As discussed in `model.1`, this is datazone is in Glasgow City Centre, which seems to have some quirky data values.