# Math 463 HW6

Ian McConachie

First, we need to import the data from the internet into our environment:

```r
myfile <- "https://pages.uoregon.edu/dlevin/DATA/infmort.txt"
all_infmort <- read.csv(myfile, header=T)
# Removing the datapoints without mortality statistics from the dataframe
infmort = all_infmort[-c(24,91,86,83),]
# Changing categorical variables into numeric variables for future use
infmort$oil <- ifelse(infmort$oil == 'oil exports', 1, 0)
infmort$region <- as.numeric(factor(infmort$region))
```

We can then make a model that includes has infant mortality as the response variable and all other variables as predictors. Generating an analysis of variance table will then tell us which variables we should include as predictors to model the response. Specifically, we can do an F test on the inclusion of each variable to see if including it causes there to be a reduction in error that is statistically significant under the null hypothesis that the error displays an F distribution and the variable should not be part of the model.

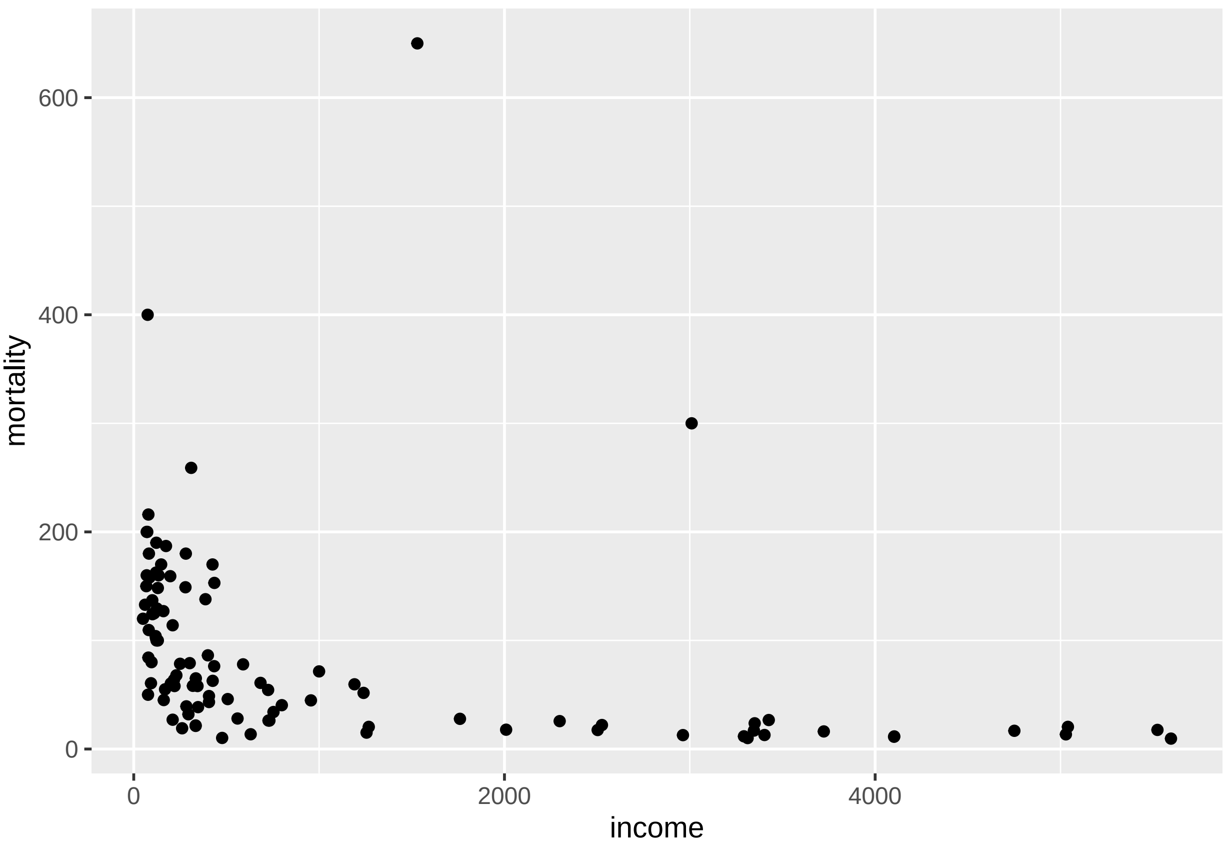
```r
diff_anova <- aov(mortality~income*oil + income*region, data=infmort)
summary(diff_anova)
```

```
##               Df Sum Sq Mean Sq F value   Pr(>F)    
## income         1  90086   90086  17.245 7.17e-05 ***
## oil            1  56902   56902  10.893  0.00136 ** 
## region         1  51834   51834   9.923  0.00218 ** 
## income:oil     1 108147  108147  20.703 1.59e-05 ***
## income:region  1  21267   21267   4.071  0.04644 *  
## Residuals     95 496258    5224                     
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

To decide which variables should be included, we have to set some arbitrary line for which p-values lower than that are considered a small enough probability to reject the null hypothesis and include the variable in the model. We can call this value alpha and set it at the de-facto standard alpha of 0.05. With this level of test, we see that all 5 variables (main effects and interaction terms) result in rejecting the null hypothesis for these F tests. This implies that all 5 variables should be included in a modeling the response.

Income and mortality are the only numeric variable in this dataset, so we can see if there are any transformations we can do to these numeric values that will result in a better fit. To get an idea of what these transformations may be we can graph the income data against the response with the R code below:

```r
ggplot(infmort, aes(x=income, y=mortality)) + geom_point()
```



By looking at the distribution of the points in the x dimension in the above graph we can get a good idea of how the data is distributed in the income variable. One thing we can notice is that the points are spread across multiple magnitudes and most of them are concentrated at lower values. This suggests that a logarithmic transformation on the income variable could improve the fit. We see a very similar thing with the data in the y-dimension which suggests that our fit may also benefit from a logarithmic transformation being applied to the mortality variable.

To see if these transformations are reasonable, we can compare EPLD values of the model with the transformations and without the transformations using LOO cross validation. We must make sure to adjust the log-transformed responses by their Jacobian to allow direct comparison between the two ELPD values.

```
# Generating the models
m_fit <- stan_glm(mortality~income*oil + income*region, data=infmort, refresh=0)
log_m_fit <- stan_glm(log(mortality)~log(income)*oil + log(income)*region, data=infmort, refresh=0)
# Examining both with LOO cross validation
mloo = loo(m_fit)
log_mloo = loo(log_m_fit)
# Adjusting the log-transformed LOO analysis with the Jacobian
log_mloo_wJ <- log_mloo
log_mloo_wJ$pointwise[,1] <- log_mloo_wJ$pointwise[,1] - log(infmort$mortality)
sum(log_mloo_wJ$pointwise[,1])
```

```
## [1] -500.1601
```

```
# Comparing the two LOO analyses
loo_compare(mloo, log_mloo_wJ)
```

```
##           elpd_diff se_diff
## log_m_fit   0.0       0.0
## m_fit     -90.5      18.8
```

The results of the above LOO analysis will be different every time due to stochastic processes used in the stan_glm function in R; however, every time I have ran the code, I have found that log_m_fit has a higher ELPD indicating that the log-transformed model has a higher predictive capability than the model that has not been transformed. This tells us that our transformations improved the model's fit.

Now that we have our model all figured out, we can compute the effect size of income, the only numeric variable in the data frame that we can do this procedure with. We start by calculating the 10% and 90% quantile values in the income variable.

```
summary(log(infmort$income))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.912   4.868   5.811   6.034   7.083   8.630
```

```
quantile(log(infmort$income),c(0.10,0.90))
```

```
##      10%      90%
## 4.394449 8.116716
```

Then, we can generate predicted values for these two quantile values and find the mean of their difference to find the effect size of the income variable on the response of infant mortality.

```
newmort_lowd <- infmort
newmort_hid <- infmort
newmort_lowd$income <- 4.394449
newmort_hid$income <- 8.116716
pvl <- predict(log_m_fit,newdata=newmort_lowd,type="response")
pvh <- predict(log_m_fit,newdata=newmort_hid,type="response")
mean(pvl-pvh)
```

```
## [1] 0.2221258
```

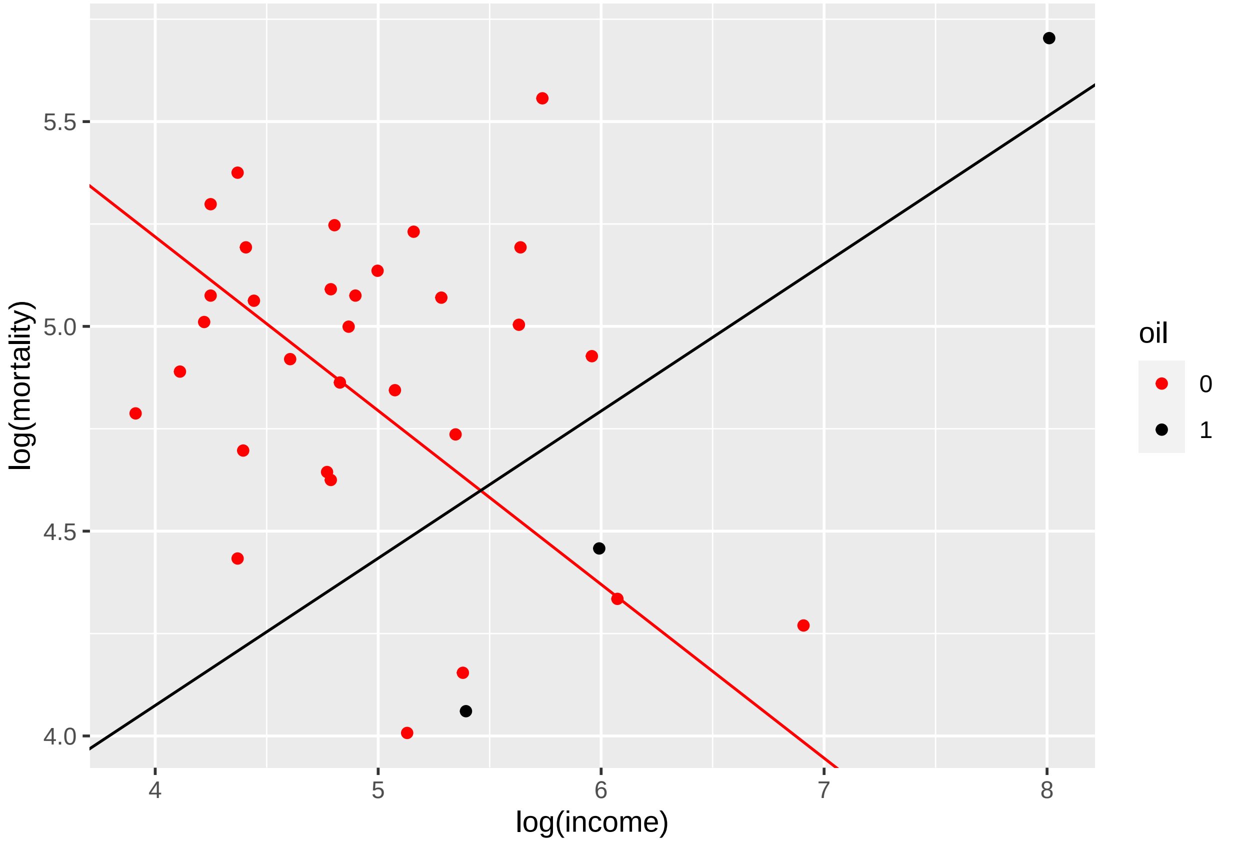
With the method we used to calculate the effect size above, the number we ended up with is relatively small, meaning that income does not have a very strong effect on mortality. However, it is not small enough to conclude that income has no effect on infant mortality. Without other effect sizes to compare it to or another model to compare with, there isn't much more analysis we can do on this singular statistic.

Now we can graph expected response (log of mortality rates) as a function of the log income predictor. To address the other two predictors in the model, we can make 4 graphs (one for each region) that each have 2 lines (one for each status of oil production.) Usually we can use ggeffects to make such a plot, but I kept running into a strange error with that function, so I just did it manually with the R code below.

```
library(dplyr)
# Reformatting oil and regionb variables for ggplot graphing
infmort_alt <- infmort
infmort_alt$oil <- as.factor(infmort$oil); infmort_alt$region <- as.factor(infmort$region)
# Calculating slope and intercept for each line
cf = coef(log_m_fit)
```

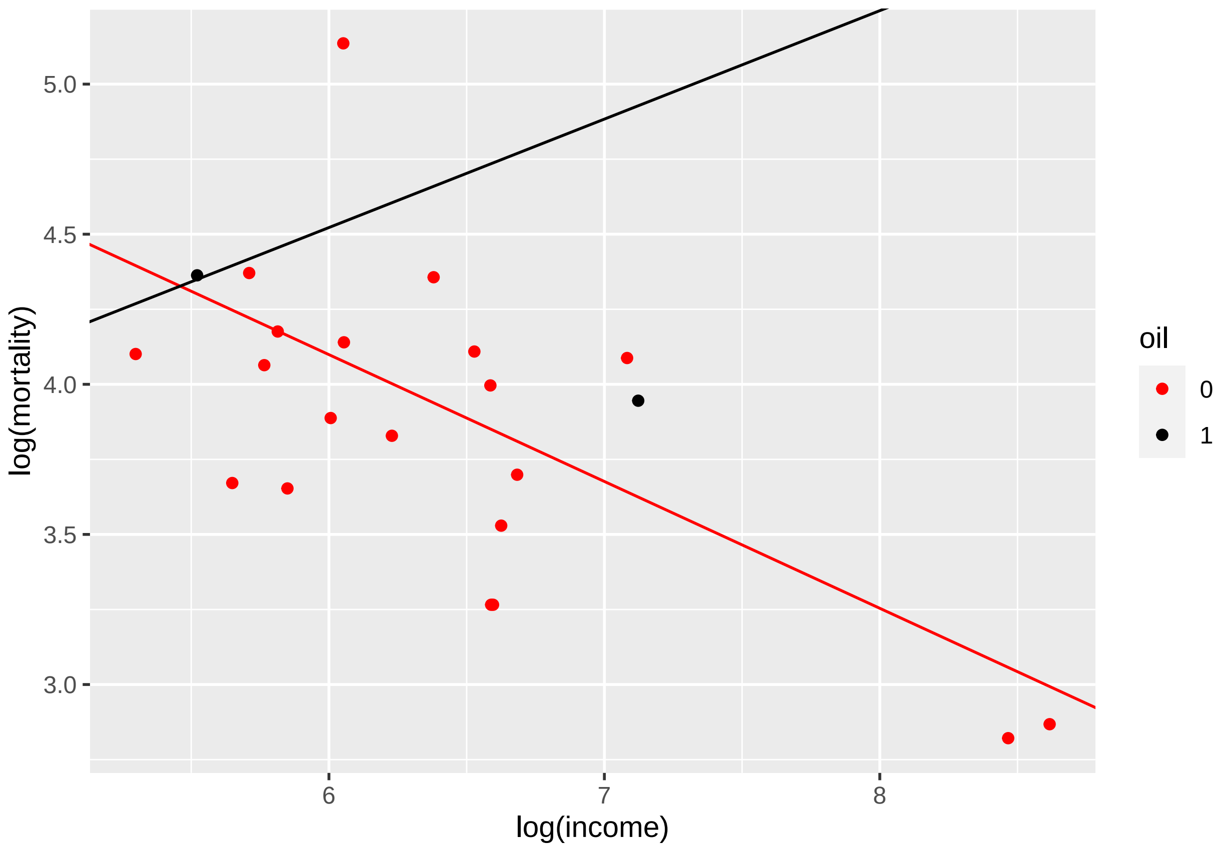
```
# Region 1
slope1 = cf[2] + (cf[6]*1); inter1 = cf[1] + (cf[4]*1);
slope2 = cf[2] + (cf[6]*1) + cf[5]; inter2 = cf[1] + (cf[4]*1) + cf[3];

infmort1 <- filter(infmort_alt, infmort_alt$region=='1')
ggplot(infmort1, aes(x=log(income), y=log(mortality), color=oil)) +
  scale_color_manual(values = c("red", "black")) +
  geom_point() + geom_abline(slope=slope1, intercept=inter1, colour='red') +
  geom_abline(slope = slope2, intercept = inter2)
```



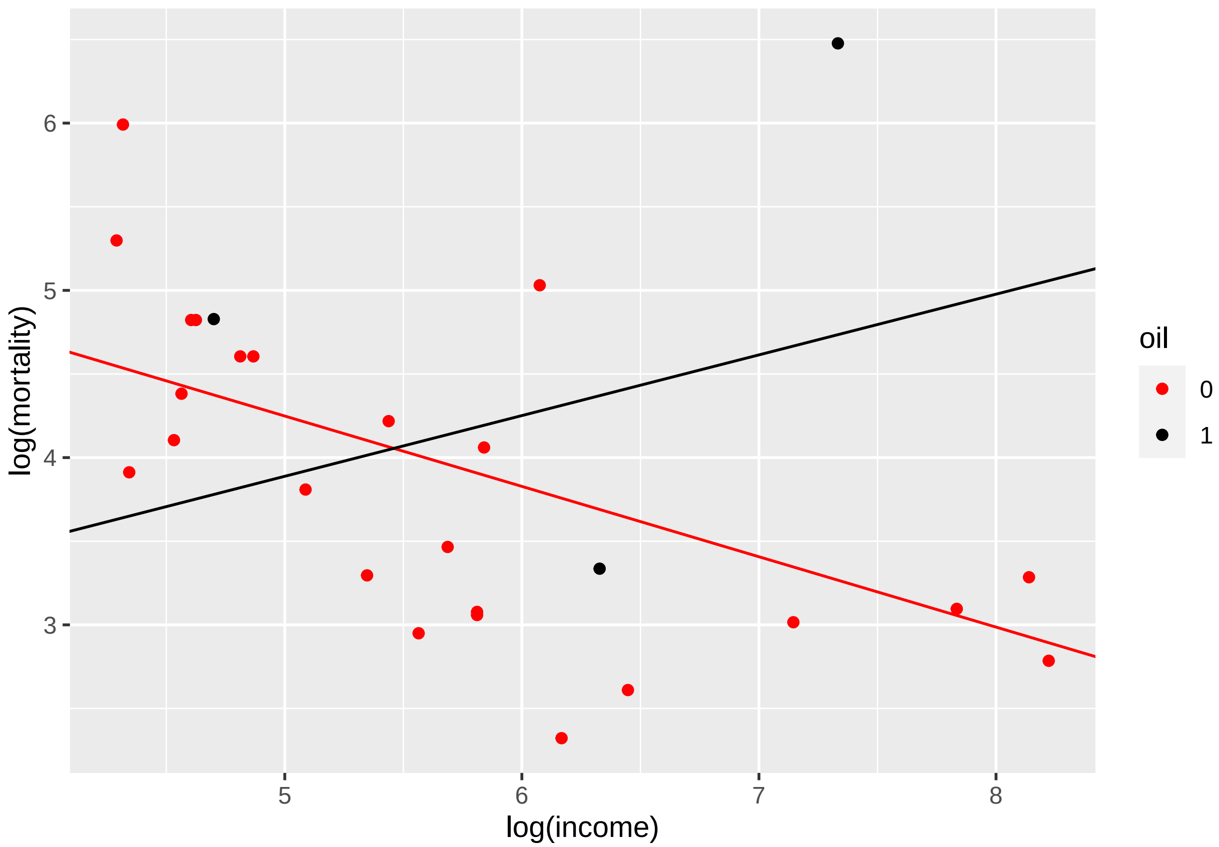
```
#Region 2
slope3 = cf[2] + (cf[6]*2); inter3 = cf[1] + (cf[4]*2);
slope4 = cf[2] + (cf[6]*2) + cf[5]; inter4 = cf[1] + (cf[4]*2) + cf[3];

infmort2 <- filter(infmort_alt, infmort_alt$region=='2')
ggplot(infmort2, aes(x=log(income), y=log(mortality), color=oil)) +
  scale_color_manual(values = c("red", "black")) +
  geom_point() + geom_abline(slope=slope3, intercept=inter3, colour='red') +
  geom_abline(slope = slope4, intercept = inter4)
```



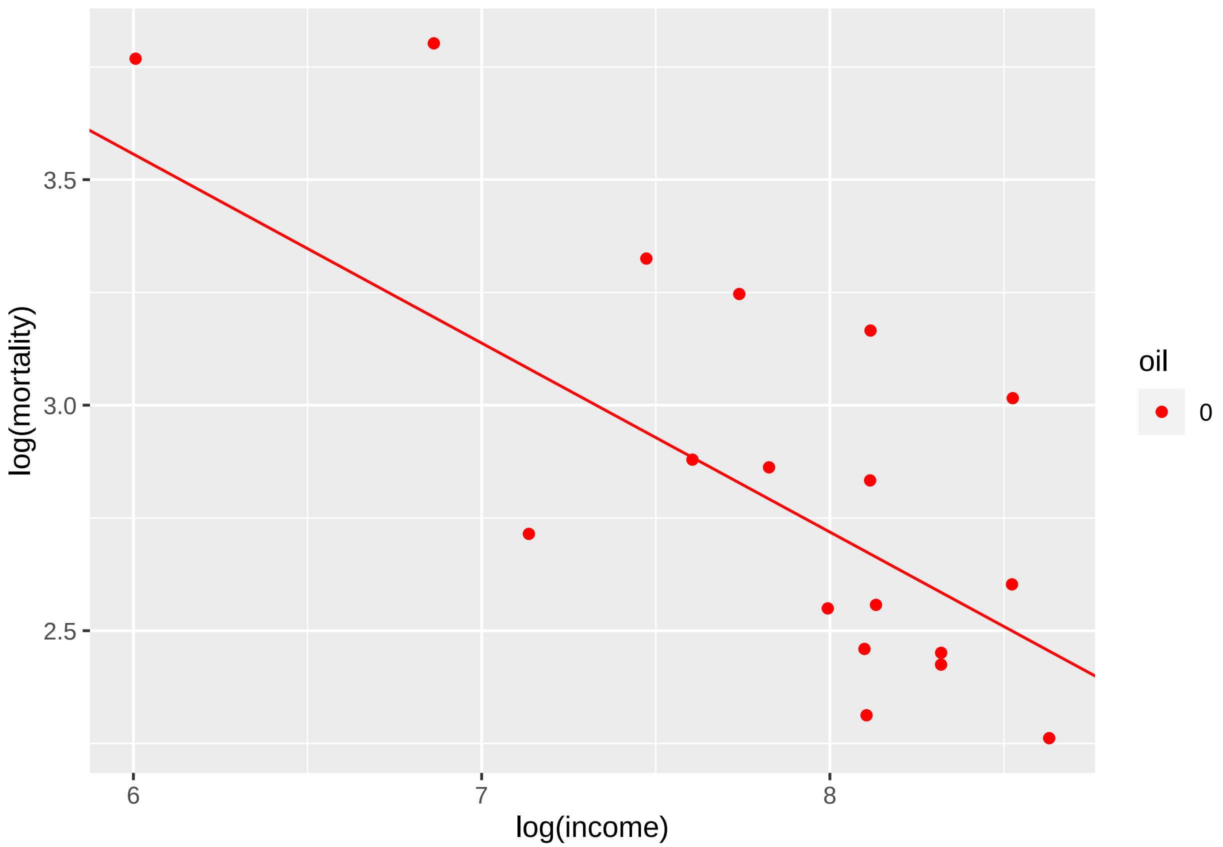
```r
# Region 3
slope5 = cf[2] + (cf[6]*3); inter5 = cf[1] + (cf[4]*3);
slope6 = cf[2] + (cf[6]*3) + cf[5]; inter6 = cf[1] + (cf[4]*3) + cf[3];

infmort3 <- filter(infmort_alt, infmort_alt$region=='3')
ggplot(infmort3, aes(x=log(income), y=log(mortality), color=oil)) +
  scale_color_manual(values = c("red", "black")) +
  geom_point() + geom_abline(slope=slope5, intercept=inter5, colour='red') +
  geom_abline(slope = slope6, intercept = inter6)
```



```
# Region 4
slope7 = cf[2] + (cf[6]*4); inter7 = cf[1] + (cf[4]*4);
slope8 = cf[2] + (cf[6]*4) + cf[5]; inter8 = cf[1] + (cf[4]*4) + cf[3];

infmort4 <- filter(infmort_alt, infmort_alt$region=='4')
ggplot(infmort4, aes(x=log(income), y=log(mortality), color=oil)) +
  scale_color_manual(values = c("red")) +
  geom_point() + geom_abline(slope=slope7, intercept=inter7, colour='red') +
  geom_abline(slope = slope8, intercept = inter8)
```



Something interesting that stands out in these plots in that in non-oil producing countries, infant mortality has a negative relationship with income (as we would expect), but in oil producing countries across the regions, we see a positive relationship. If we take this as the absolute truth, then the higher the per capita income of an oil-producing country, the higher the infant mortality rate is.

One thing we can note about this strange trend we're seeing is that there are far less oil exporting data points than non-oil exporting data points. This means a few anomalous data points in the oil category would have disproportionate sway on the overall trend we observe in the expected infant mortality. These anomalous points could possibly be the high income oil data points in regions 1 (Africa) and 3 (Asia). These countries could have a large wealth disparity, deficiencies in their healthcare system, or some sort of bias in the way this data was collected. Without more information, we cannot know for sure what is causing this trend to show up in the data.