EDA_FA4_KHAFAJI

Exploring the Mortality by Latitude dataset

First, let's load the dataset.

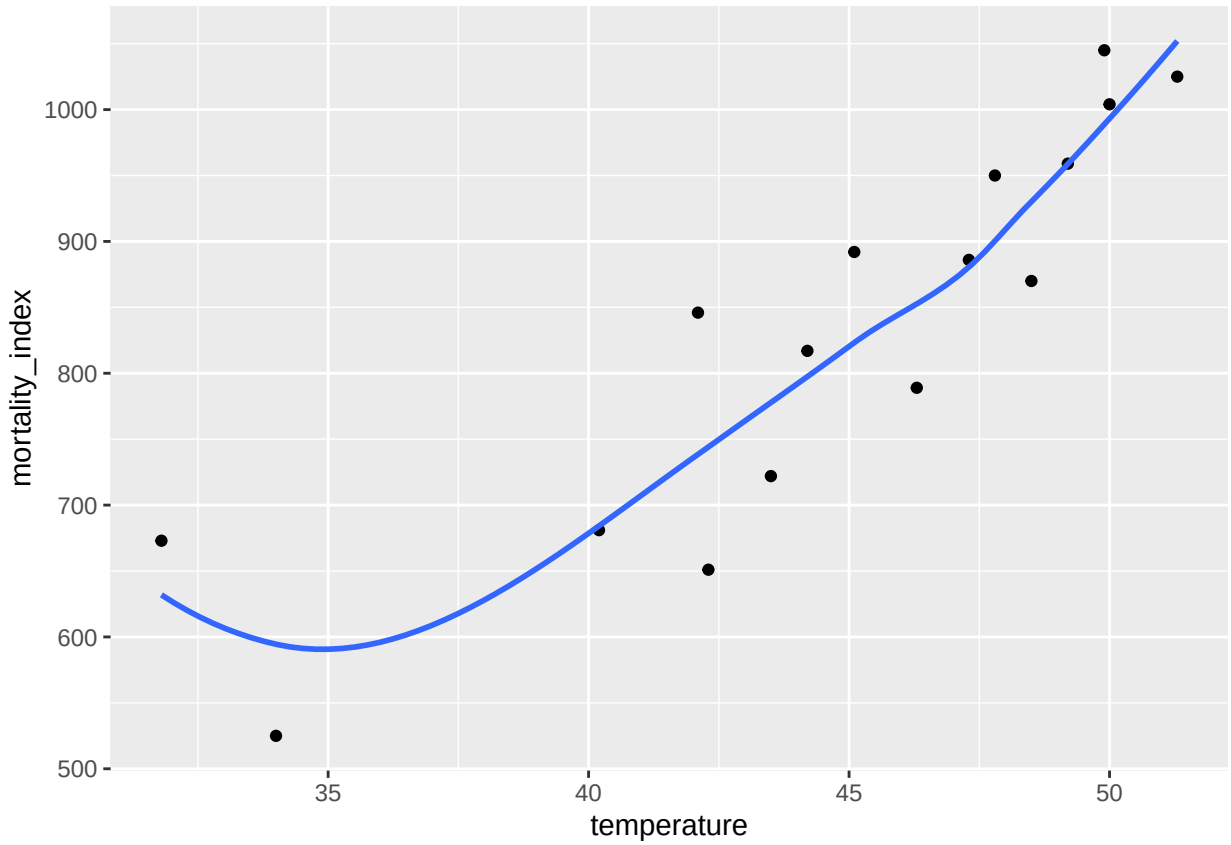
```
mbl <- read.csv("mortality_by_latitude.csv", sep=",")
mbl
```

```
##    latitude mortality_index temperature
## 1        50            1025        51.3
## 2        51            1045        49.9
## 3        52            1004        50.0
## 4        53             959        49.2
## 5        54             870        48.5
## 6        55             950        47.8
## 7        56             886        47.3
## 8        57             892        45.1
## 9        58             789        46.3
## 10       59             846        42.1
## 11       60             817        44.2
## 12       61             722        43.5
## 13       62             651        42.3
## 14       63             681        40.2
## 15       69             673        31.8
## 16       70             525        34.0
```

1 Make a plot of the mortality index against mean average temperature.

```
mbl %>% ggplot(aes(y = mortality_index, x=temperature)) +
  geom_point() +
  geom_smooth(method=loess
              , se=FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

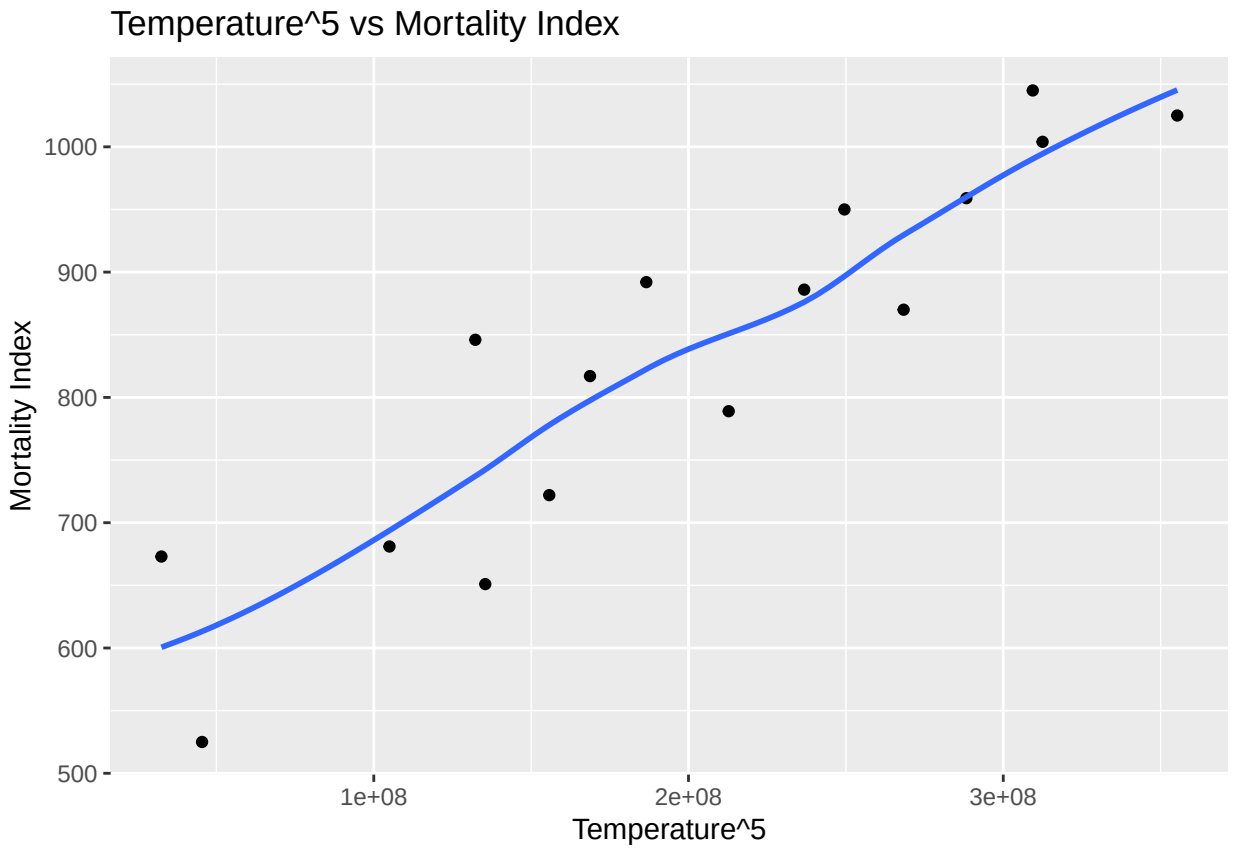


From here, we can conclude that the data is hollow up. Let's find a transformation that can help with that.

```
mbl <- mbl %>% mutate(temperature_5thPower = temperature^5)

mbl %>% ggplot(aes(y = mortality_index, x=temperature_5thPower)) +
  geom_point() +
  geom_smooth(method=loess
              , se=FALSE)+
  labs(title="Temperature^5 vs Mortality Index", x="Temperature^5", y="Mortality Index")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

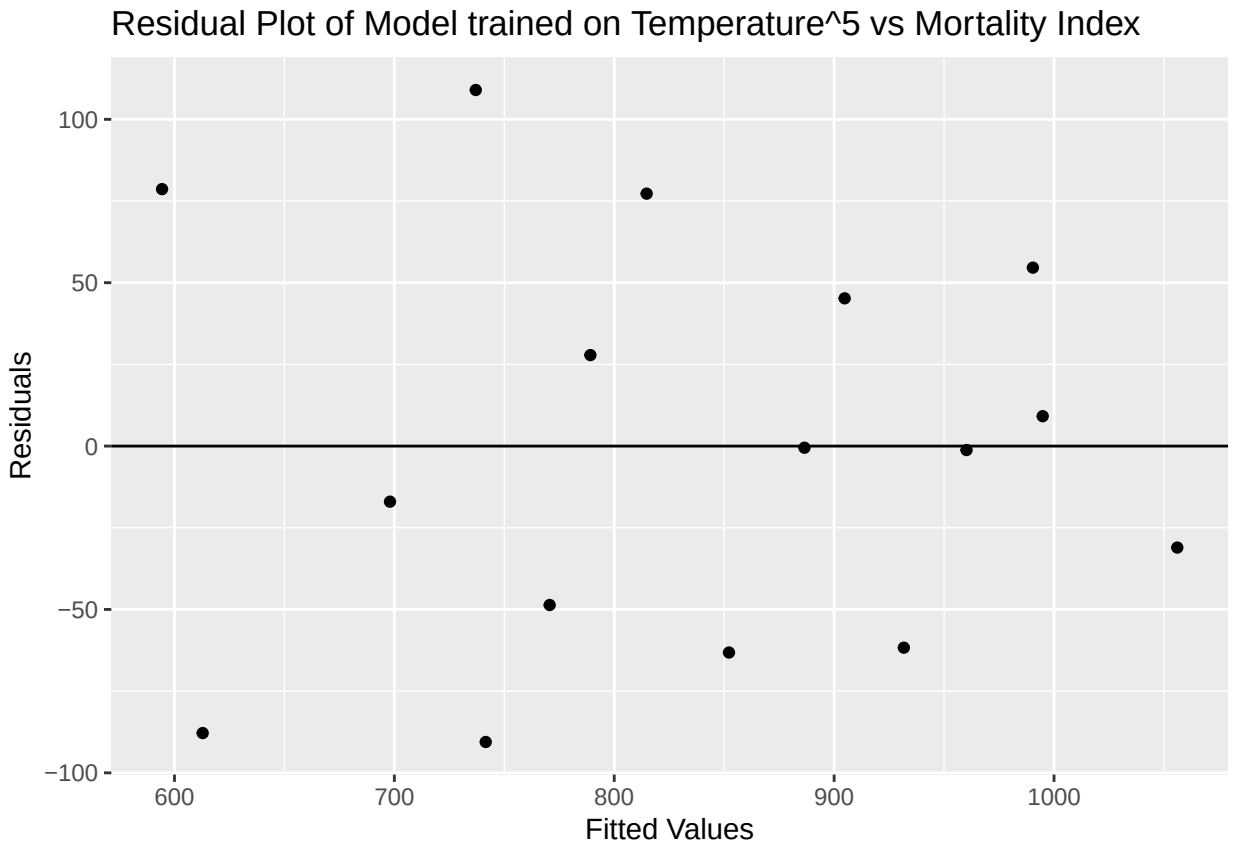


We can see that

let's then make a plot of the residuals to check for any remaining patterns.

```
model <- lm(mortality_index ~ temperature_5thPower, data=mbl)

model %>% ggplot(aes(x=.fitted, y=.resid)) +
  geom_point()+
  geom_hline(yintercept=0)+
  labs(title="Residual Plot of Model trained on Temperature^5 vs Mortality Index", x="Fitted Values", y=
```



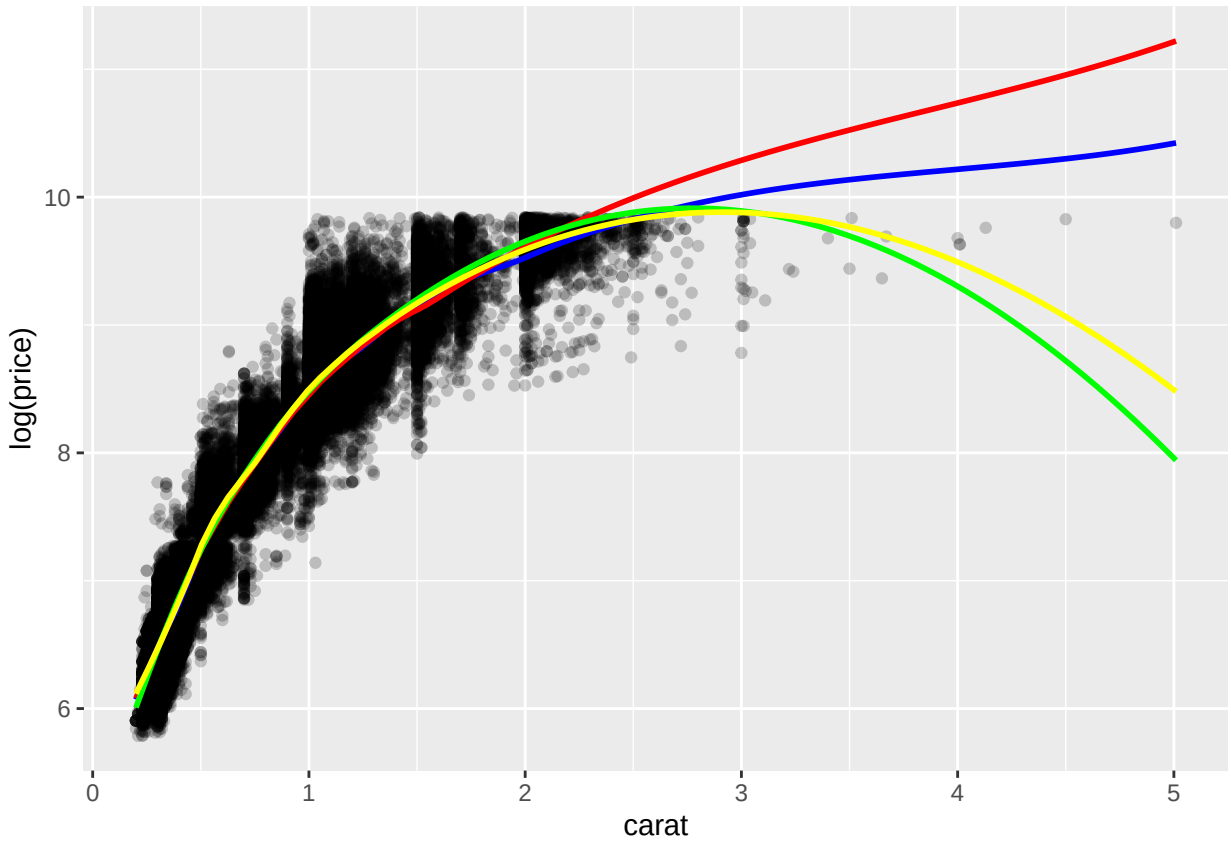
We can see that The model is not that accurate with low temperatures, but increases in accuracy as temperature increases.

Exploring the Diamonds Dataset

2. Using the same subset of the diamonds dataset, make a plot of log price as a function of carat with a loess smoother. Try several values for the span and degree arguments and comment briefly about your choice.

```
diamonds %>% ggplot(aes(x=carat, y=log(price)))+
  geom_point(alpha=0.2)+
  geom_smooth(method = "loess", se = FALSE, span = 0.3, method.args = list(degree = 1), color = "blue")
  geom_smooth(method = "loess", se = FALSE, span = 0.5, method.args = list(degree = 1), color = "red") +
  geom_smooth(method = "loess", se = FALSE, span = 0.7, method.args = list(degree = 2), color = "green"
  geom_smooth(method = "loess", se = FALSE, span = 0.5, method.args = list(degree = 2), color = "yellow
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



The loess fit with a span of 0.5 and degree = 2 fit the data better than the other spans and degrees

3. Compare the fit of the loess smoother to the fit of the polynomial + step function regression using a plot of the residuals in the two models. Which one is more faithful to the data?

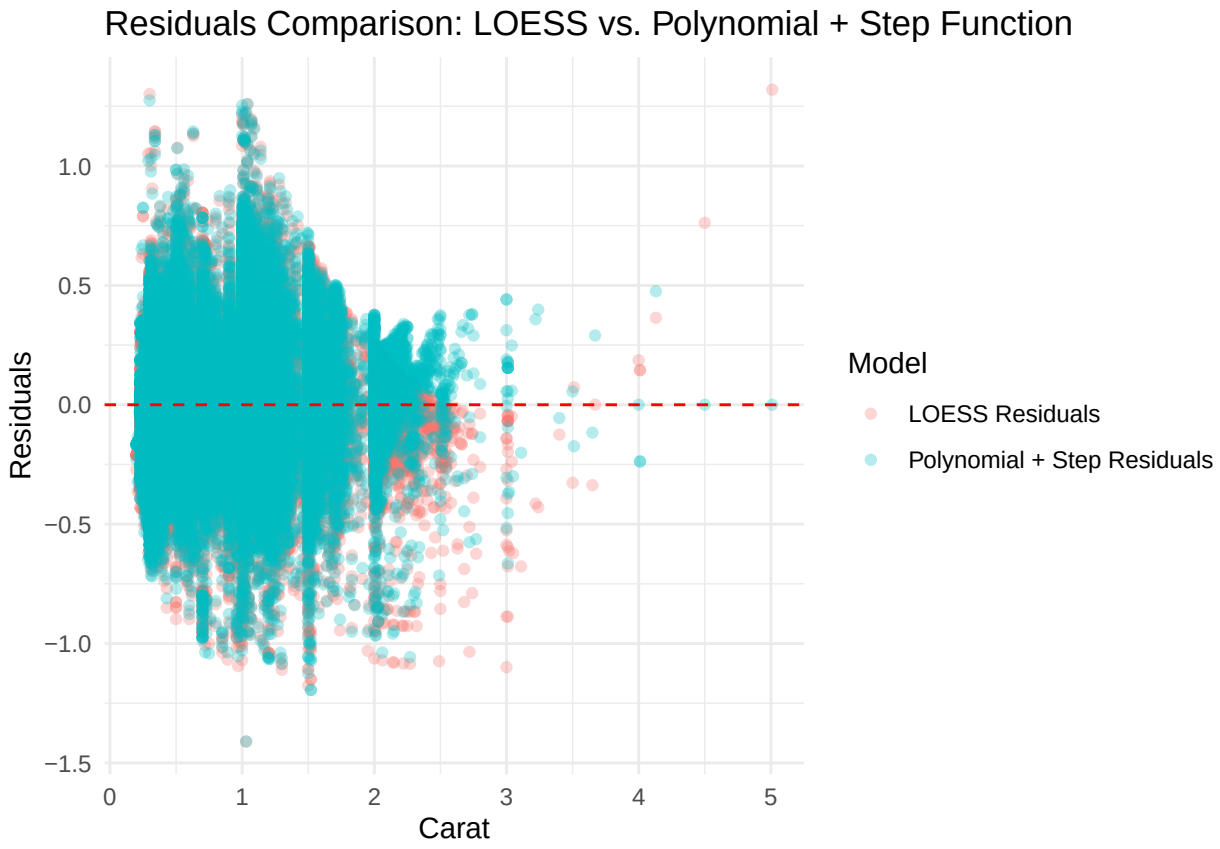
```
diamonds <- diamonds %>% mutate(log_price = log(price))

loess_fit <- loess(log_price ~ carat, data = diamonds, span = 0.5, degree = 2)
diamonds <- diamonds %>%
  mutate(loess_resid = residuals(loess_fit))


diamonds <- diamonds %>%
  mutate(carat_bin = cut(carat, breaks = seq(0, 10, by = 0.25)))
poly_step_fit <- lm(log_price ~ poly(carat, degree = 2) + carat_bin, data = diamonds)
diamonds <- diamonds %>%
  mutate(poly_step_resid = residuals(poly_step_fit))

ggplot(diamonds) +
  geom_point(aes(x = carat, y = loess_resid, color = "LOESS Residuals"), alpha = 0.3) +
  geom_point(aes(x = carat, y = poly_step_resid, color = "Polynomial + Step Residuals"), alpha = 0.3) +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals Comparison: LOESS vs. Polynomial + Step Function",
       x = "Carat",
       y = "Residuals",
```

```
       color = "Model") +
  theme_minimal()
```



As we can see, although they have almost the same distribution, the Polynomial + Step regression has a smaller absolute value of the residuals. Therefore, Polynomial + Step regression is the better model for log price as a function of carat.