Formative Assessment 4

PATAYON, SPIKE LEE-ROY V

2025-03-02

Instruction

1. Using the Mortality by Latitude data Download Mortality by Latitude data, make a plot of mortality index against mean average temperature. Is it hollow up or hollow down? Try to identify a transformation of one of the variables that will straighten out the relationship, and make a plot of the residuals to check for any remaining patterns.

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.

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?

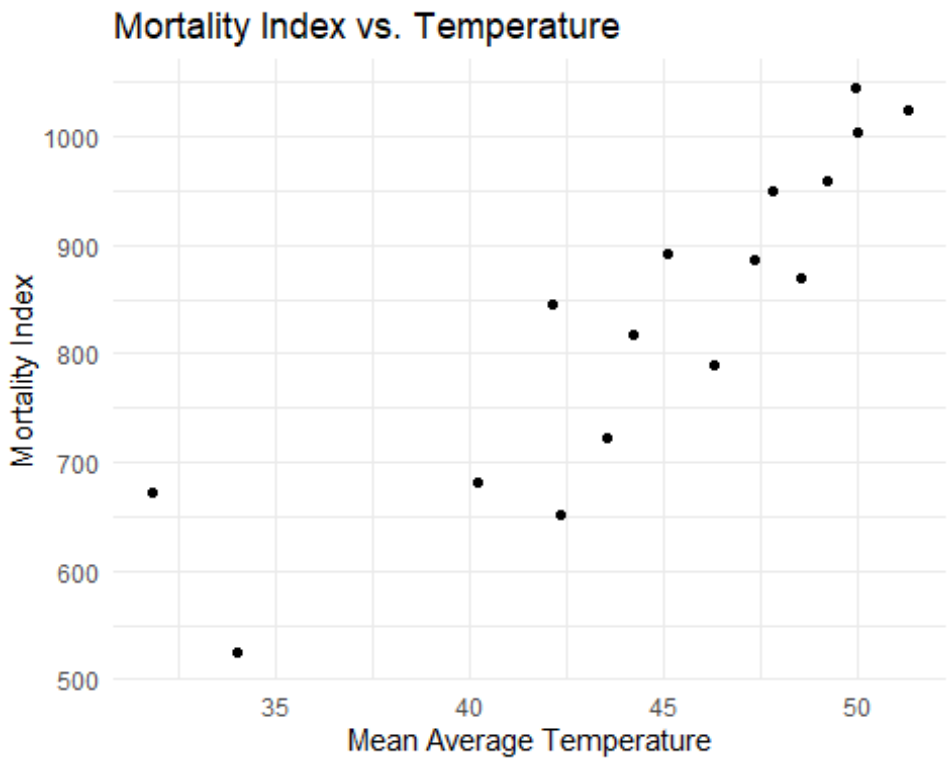
```
mortality_data <-
read.csv("C:\\Users\\spike\\Downloads\\mortality_by_latitude.csv")

mortality_data
```

```
##    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. Using the Mortality by Latitude data, make a plot of mortality index against mean average temperature. Is it hollow up or hollow down? Try to identify a

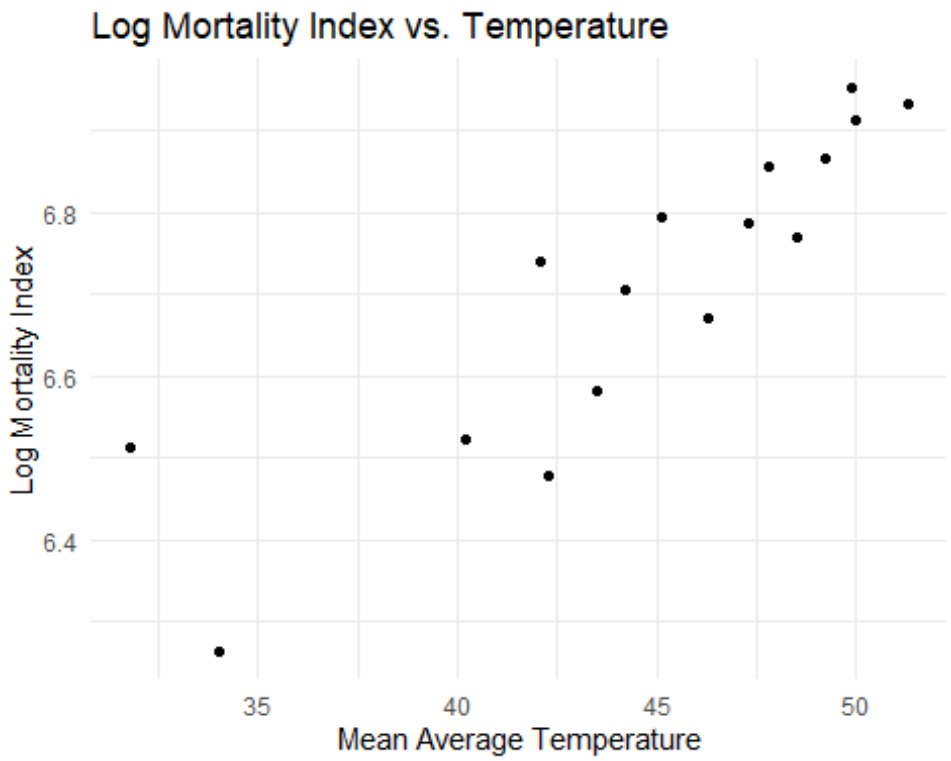
transformation of one of the variables that will straighten out the relationship, and make a plot of the residuals to check for any remaining patterns.

```r
library(ggplot2)

ggplot(mortality_data, aes(x = temperature, y = mortality_index)) +
  geom_point() +
  labs(title = "Mortality Index vs. Temperature",
       x = "Mean Average Temperature",
       y = "Mortality Index") +
  theme_minimal()
```

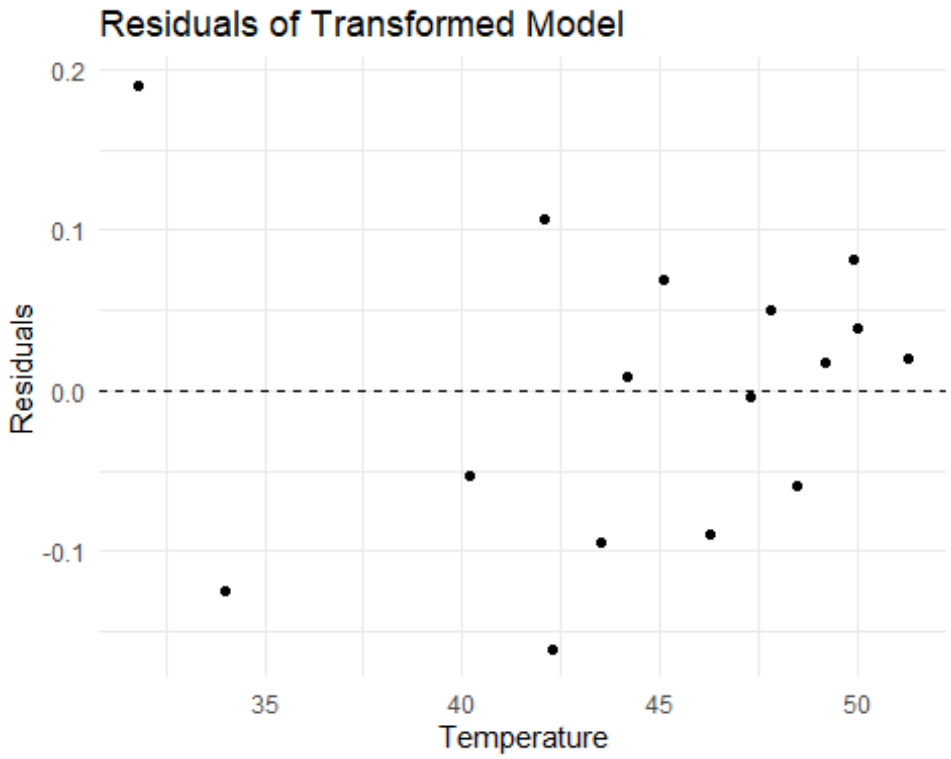


insights: From the scatter plot, the relationship between Mortality Index and Mean Average Temperature appears to be hollow up (concave up). This suggests that applying a log transformation to mortality_index might help linearize the relationship.

```r
mortality_data$log_mortality <- log(mortality_data$mortality_index)

ggplot(mortality_data, aes(x = temperature, y = log_mortality)) +
  geom_point() +
  labs(title = "Log Mortality Index vs. Temperature",
       x = "Mean Average Temperature",
       y = "Log Mortality Index") +
  theme_minimal()
```



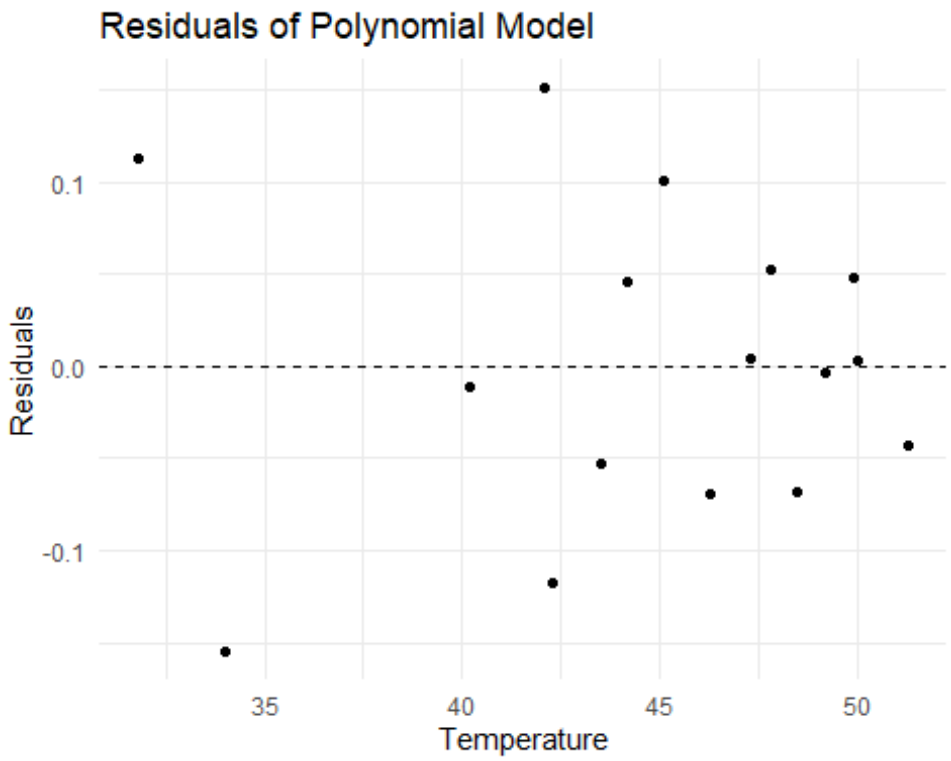

```r
model <- lm(log_mortality ~ temperature, data = mortality_data)
mortality_data$residuals <- residuals(model)

ggplot(mortality_data, aes(x = temperature, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  labs(title = "Residuals of Transformed Model",
       x = "Temperature",
       y = "Residuals") +
  theme_minimal()
```



Insights: The residuals plot shows some structure, which suggests that the log transformation helped but did not completely remove patterns in the residuals. There is a slight pattern where residuals tend to be higher for extreme values of temperature.Ideally, residuals should be randomly scattered around 0, but here, they still show some trends.

Possible next step is: Since the log transformation helped but did not fully linearize the data,I will try adding a quadratic term:

```
model_poly <- lm(log_mortality ~ temperature + I(temperature^2), data =
mortality_data)

mortality_data$residuals_poly <- residuals(model_poly)

ggplot(mortality_data, aes(x = temperature, y = residuals_poly)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  labs(title = "Residuals of Polynomial Model",
       x = "Temperature",
       y = "Residuals") +
  theme_minimal()
```



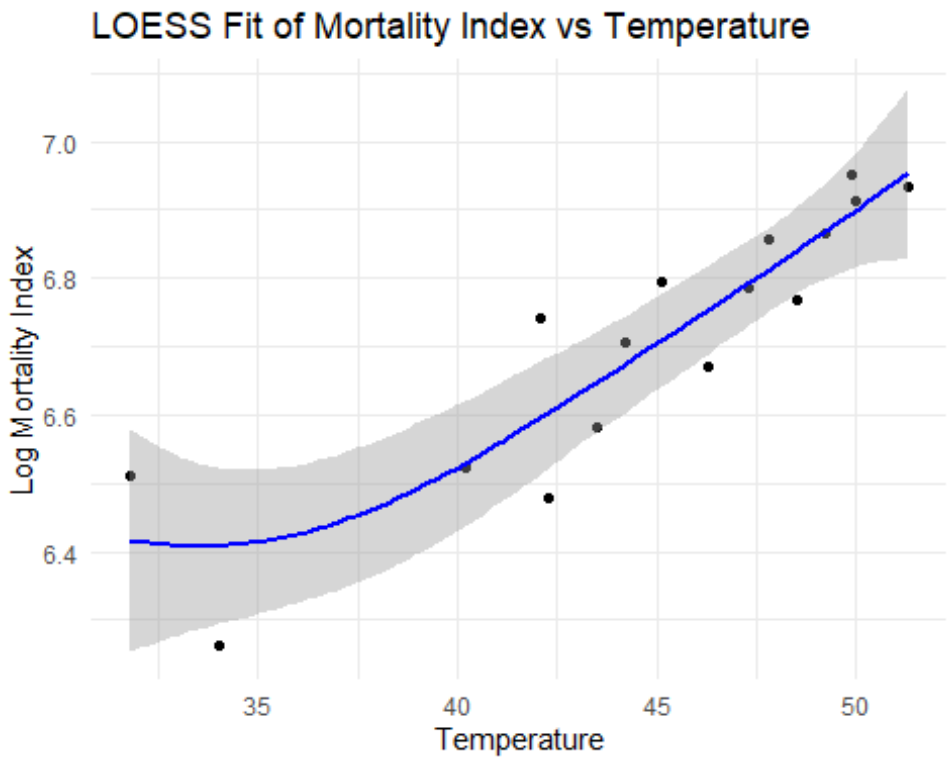
Insights: This residual plot looks improved compared to the previous one, but there are still some small patterns visible. The spread of residuals is closer to zero, meaning the polynomial model has reduced the non-linearity. However, a slight structure remains, particularly at the lower and higher temperature values. If residuals were perfectly random, we would expect a uniform scatter around the zero line.

Possible next step is:

We can try fitting a LOESS smoother with different span values to see which provides the best fit:

```
ggplot(mortality_data, aes(x = temperature, y = log_mortality)) +
  geom_point() +
  geom_smooth(method = "loess", span =1, color = "blue") +
  labs(title = "LOESS Fit of Mortality Index vs Temperature",
       x = "Temperature",
       y = "Log Mortality Index") +
  theme_minimal()
```

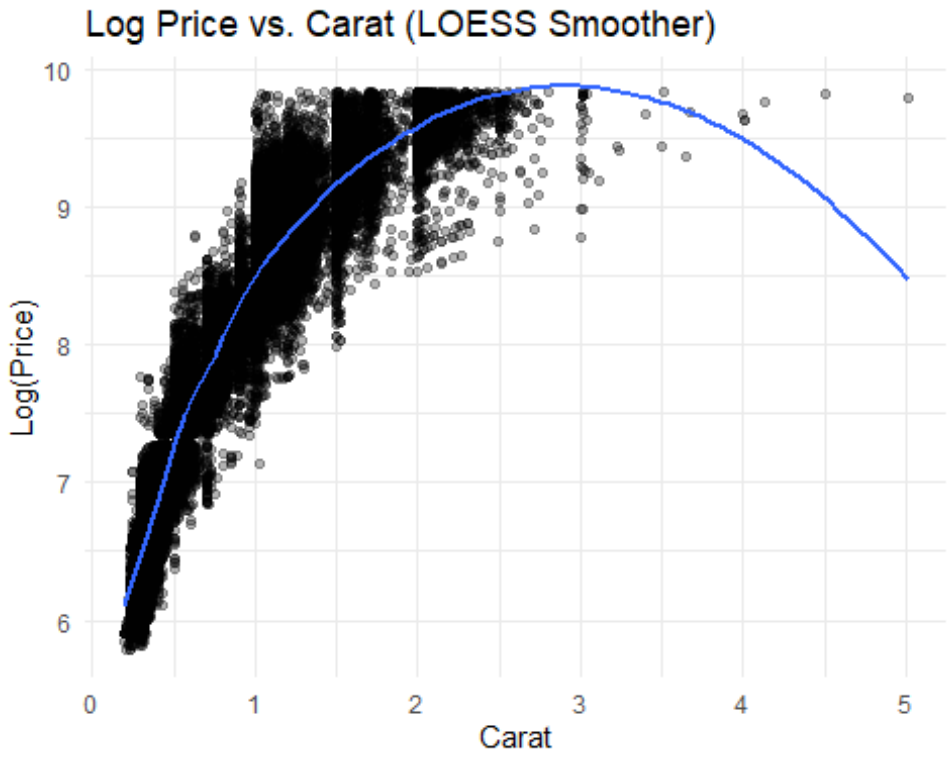
```
## `geom_smooth()` using formula = 'y ~ x'
```



With span = 1, the LOESS curve is quite smooth, capturing the overall trend while avoiding excessive flexibility. The trend appears stable and generalizes well, reducing the risk of overfitting. The confidence interval (shaded gray area) is wider in regions with fewer data points, reflecting higher uncertainty. The curve captures a gradual increase in log mortality index as temperature rises, which aligns with expectations.

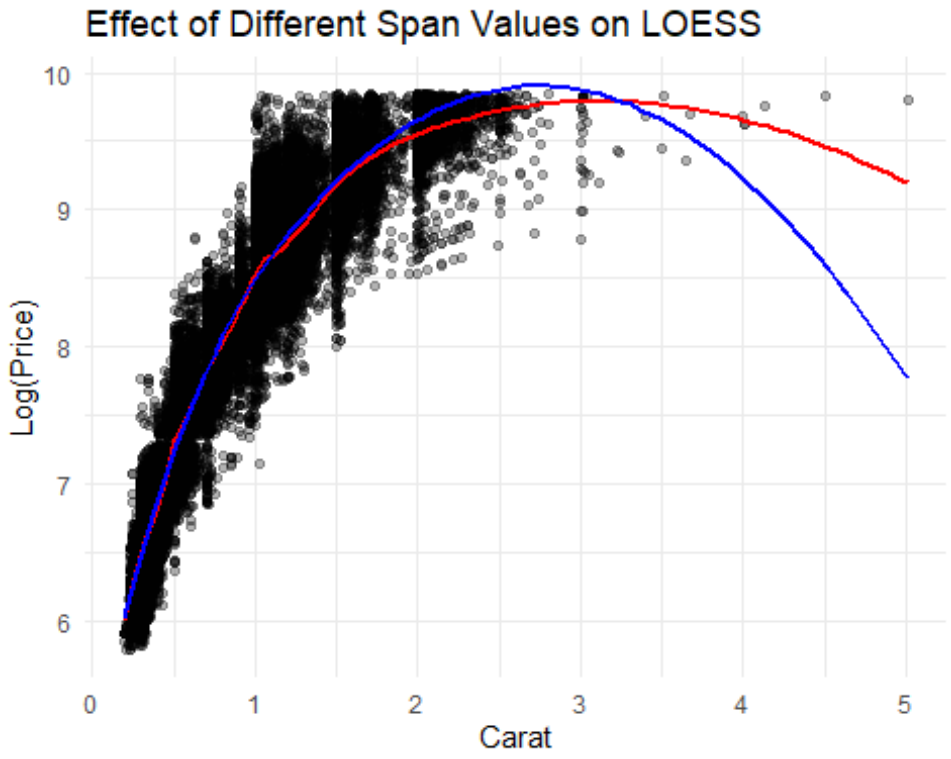
2. Using the same subset of the diamonds dataset, make a plot of log price as a function of carat with a loess smoother. Try several values for the span and degree arguments and comment briefly about your choice.

```
data(diamonds)

ggplot(diamonds, aes(x = carat, y = log(price))) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = "loess", span = 0.5, se = FALSE) +
  labs(title = "Log Price vs. Carat (LOESS Smoother)",
       x = "Carat",
       y = "Log(Price)") +
  theme_minimal()

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




```
ggplot(diamonds, aes(x = carat, y = log(price))) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = "loess", span = 0.2, se = FALSE, color = "red") +  # More flexible
  geom_smooth(method = "loess", span = 0.8, se = FALSE, color = "blue") +  # Smoother
  labs(title = "Effect of Different Span Values on LOESS",
       x = "Carat",
       y = "Log(Price)") +
  theme_minimal()
```

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



Insights:

I choose both high (0.8) and low (0.2) span value to see how it affects the data. Personally I prefer using the smaller value of span since creates a more flexible curve that follows the data closely.

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$carat_cut <- cut(diamonds$carat, breaks = c(0, 0.5, 1, 1.5, 2, 3,
5), labels = FALSE)

# Polynomial regression (degree 3)
poly_model <- lm(log(price) ~ poly(carat, 3), data = diamonds)

# Step function regression
step_model <- lm(log(price) ~ as.factor(carat_cut), data = diamonds)
```