Homework_7

Emmenta Janneh

2024-03-20

1. For each of the following regression models, is it possible to reformulate the model in terms of the general linear model (possibly after a suitable transformation)? Demonstrate how if the answer is "yes", and explain why not if the answer is "no".

    a. No, we can't directly reformulate this into general linear model. Although the term log10Xi2 might resemble a linear term after transformation,the presence of Xi1^2 still makes it nonlinear. A simple transformation won't suffice to linearize the equation.

    b. No, as it is, the model suggests a result form logging Yi and Ei; logYi = log(Ei exp(B0 + B1Xi1 + B2Xi2^2)). Using the logarithmic properties, we can simplify this to: log(Yi) = log(Ei) + B0 + B1Xi1 + B2Xi2^2. So the model is as a result of logging the error term whish is in appropriate because the error term is assumed to be random variable with certain properties(such as being normally distributed with mean zero).

    c. No, the term log(B1Xi1) involves the transformation of a constant, B1, within the logarithm. While we can manipulate the variable Xi1, we cannot directly manipulate the constant B1 with a transformation.

2. To ensure the system understands that B2 = 2, we use the `lm()` function to fit a linear model. In the section of Xi2, use the term `I(2 * Xi2)` ensures that the coefficient of Xi2 is forced to be 2.

    ```
    model <- lm(Yi ~ Xi1 + I(2 * Xi2) + Xi3, data = data)
    ```

    Finally, we print the summary of the model to see the estimated coefficients and other relevant statistics.

Brand Preference

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.2
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(broom)
```

```
## Warning: package 'broom' was built under R version 4.3.2
```

```
brand <- read_csv("https://dcgerard.github.io/stat_415_615/data/brand.csv")
```

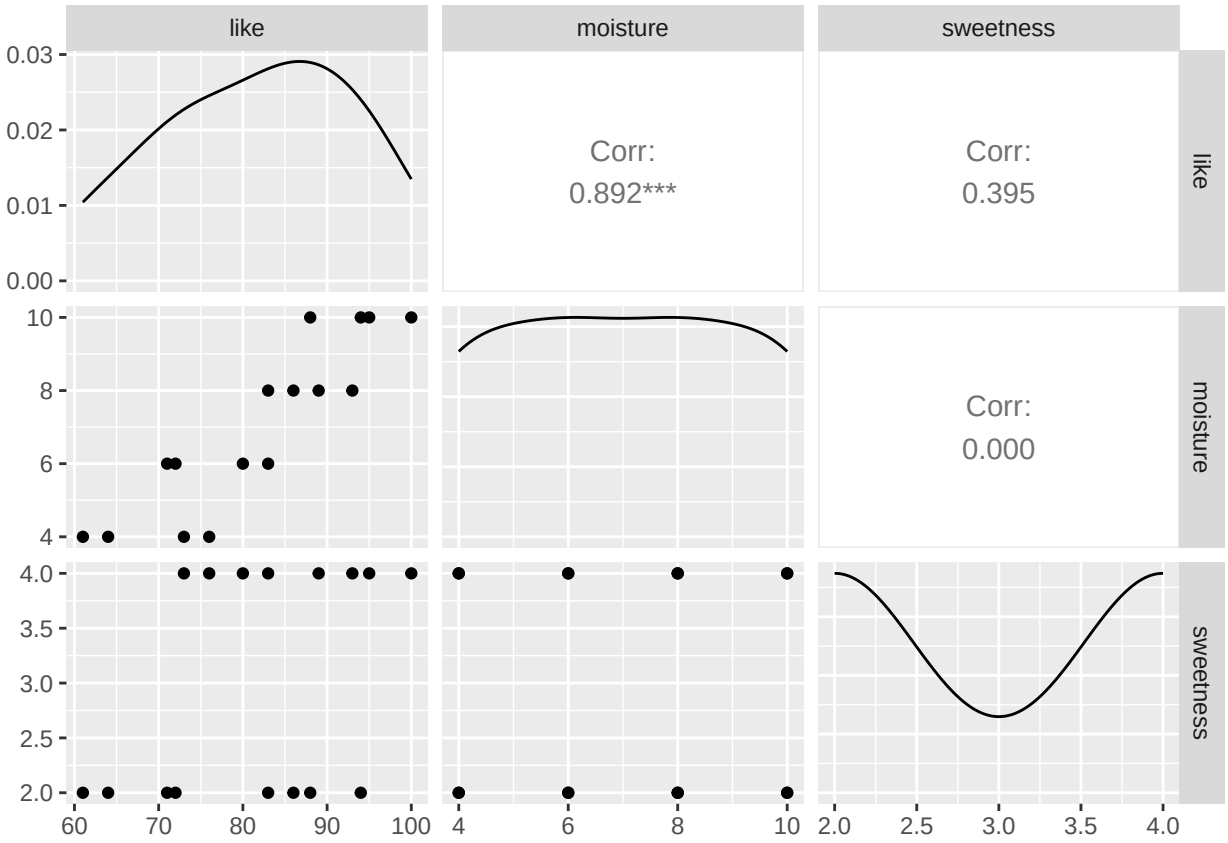
```
## Rows: 16 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (3): like, moisture, sweetness
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

1.

```
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggpairs(brand)
```



From the plot, there is a significant evidence of correlation between moisture and like. The more moist a product is, the more it is liked by customers. There isn't much evidence of correlation between sweetness and like. And sweetness and moistiure have no relationship.

2.

```
lm_brand <- lm(like ~ moisture + sweetness, data = brand)
tidy(lm_brand, conf.int = TRUE)
```

```
## # A tibble: 3 x 7
##   term        estimate std.error statistic       p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>         <dbl>    <dbl>     <dbl>
## 1 (Intercept)    37.7      3.00      12.6  0.0000000120     31.2      44.1
## 2 moisture        4.42     0.301     14.7  0.00000000178     3.77      5.08
## 3 sweetness       4.37     0.673      6.50 0.0000201         2.92      5.83
```

Yi = B0 + B1Xi1 + B2Xi2

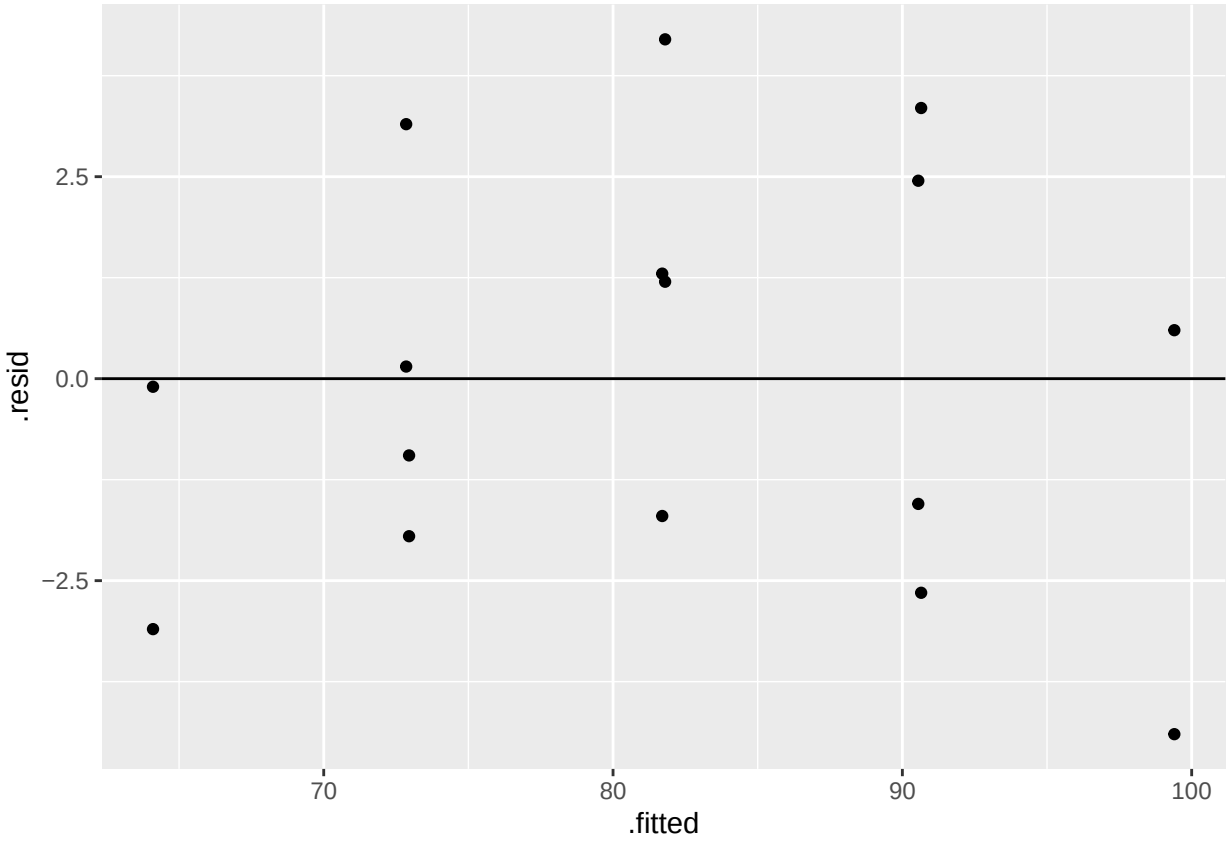
like = 37.65 + 4.425(mositure) + 4.375(sweetness)

3.

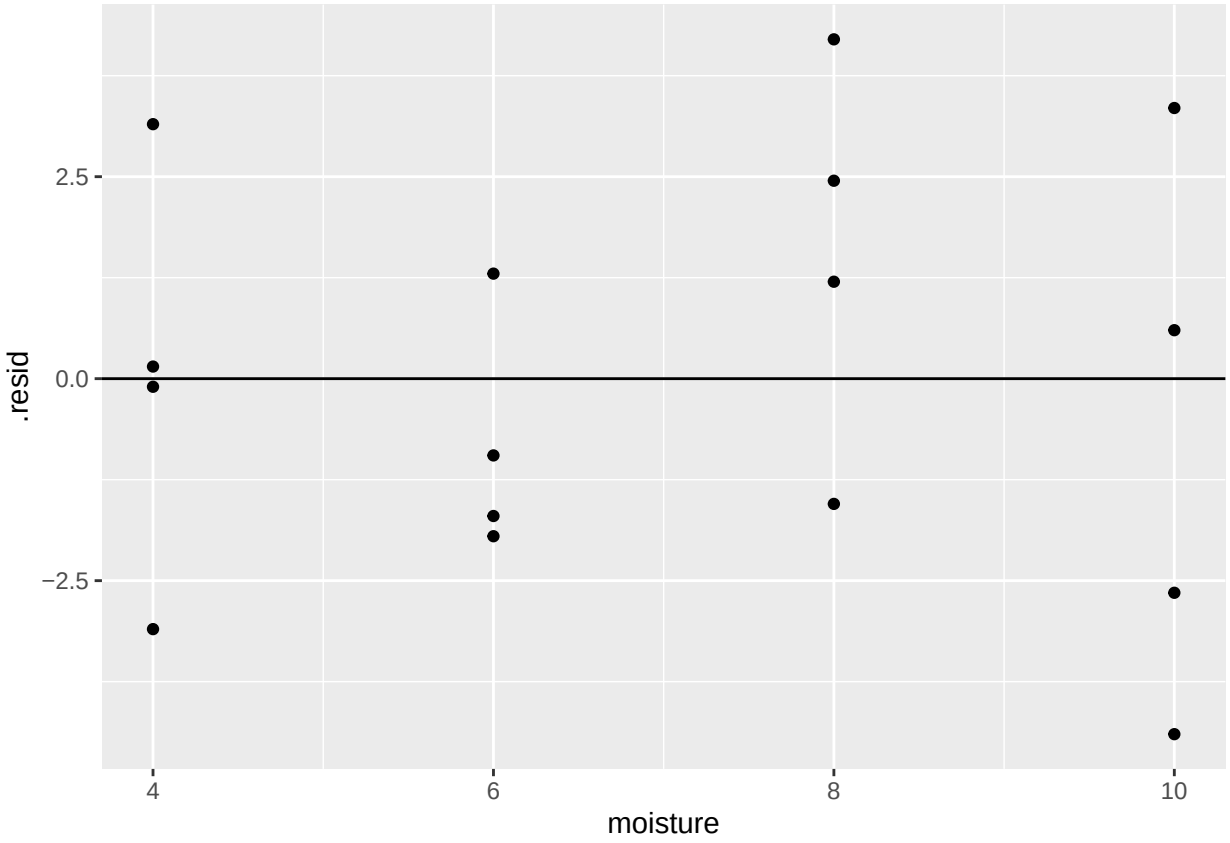
- Moisture: Brands that are 1000 times more moist, are estimated to have 4425 degree of liking on average (95% CI of 3774 to 5076), adjusting for sweetness.
- Sweetness: Brands that are 1000 times sweeter, are estimated to have 4375 degree of liking on average (95% CI of 2920 to 5830)

4.

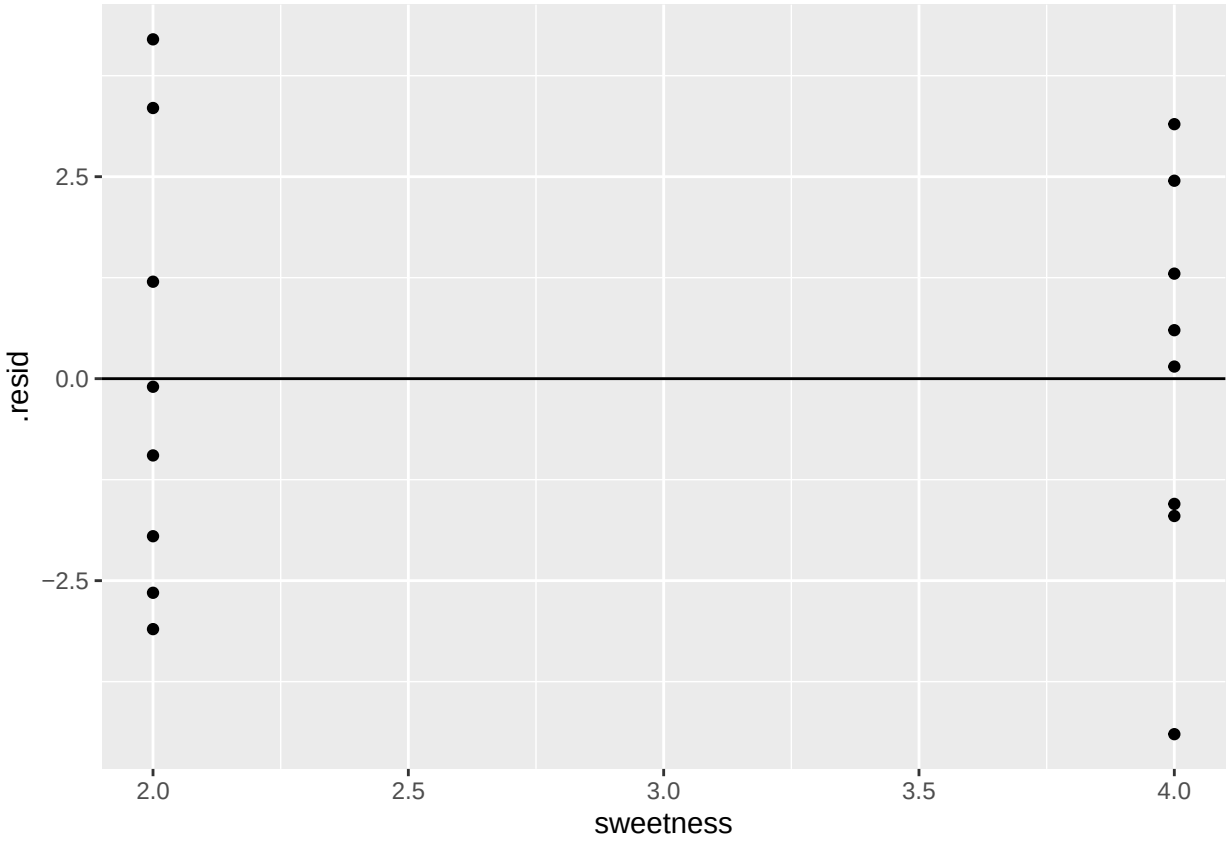
```
aug_brand <- augment(lm_brand)

ggplot(aug_brand, aes(x = .fitted, y = .resid)) +
  geom_point() +
  geom_hline(yintercept = 0)
```




```
ggplot(aug_brand, aes(x = moisture, y = .resid)) +
  geom_point() +
  geom_hline(yintercept = 0)
```




```
ggplot(aug_brand, aes(x = sweetness, y = .resid)) +
  geom_point() +
  geom_hline(yintercept = 0)
```



The residual plots on fitted and the predictor variables doesn't seem bad.

5.

```
newdf <- data.frame(moisture = 7, sweetness = 3)
predict(lm_brand, newdata = newdf, interval = "prediction")
```

```
##     fit      lwr      upr
## 1 81.75 75.75241 87.74759
```

Indicator Variables and Matrix Formulation

```
marry <- tribble(~happy, ~married,
                    73, "single",
                    79, "single",
                    72, "single",
                    58, "married",
                    72, "married",
                    74, "married",
                    77, "divorced",
                    51, "divorced",
                    63, "divorced")
dummy_data <- model.matrix(~ married - 1, data = marry)

dummy_data <- cbind(dummy_data, happy = marry$happy)

dummy_data <- as.data.frame(dummy_data)
dummy_data <- dummy_data |>
  select(happy, marriedmarried, marrieddivorced)
mlr_model <- lm(happy ~ ., data = dummy_data)

tidy(mlr_model)
```

```
## # A tibble: 3 x 5
##   term            estimate std.error statistic    p.value
##   <chr>              <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)        74.7       5.37    13.9   0.00000863
## 2 marriedmarried     -6.67      7.60    -0.878 0.414
## 3 marrieddivorced   -11         7.60    -1.45  0.198
```

-11.0 is the estimated mean happiness level of divorced individuals.