Identifying Drivers of Outcomes: Linear Models

Luis Francisco Gómez López

FAEDIS

2025-05-05

UNIVERSIDAD MILITAR
NUEVA GRANADA

Table of contents I

- This presentation is based on (Chapman and Feit 2019, chap. 7)

- Apply linear modeling to understand a response variable and make predictions of forecasts

- **weekend**: whether the visit was on a weekend
- **num.child**: number of children in the visit
- **distance**: how far the customer traveled to the park in miles
- **rides**: satisfaction with rides using a scale [0, 100]
- **games**: satisfaction with games using a scale [0, 100]
- **wait**: satisfaction with waiting times using a scale [0, 100]
- **clean**: satisfaction with cleanliness using a scale [0, 100]
- **overall**: overall satisfaction rating using a scale [0, 100]

- Import data

```
amusement_park <- read_csv("http://goo.gl/HKnl74")
amusement_park |> head(n = 5)
```

```
# A tibble: 5 x 8
  weekend num.child distance rides games  wait clean overall
  <chr>       <dbl>    <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>
1 yes             0    115.     87    73    60    89      47
2 yes             2     27.0    87    78    76    87      65
3 no              1     63.3    85    80    70    88      61
4 yes             0     25.9    88    72    66    89      37
5 no              4     54.7    84    87    74    87      68
```

- Transform data

```r
amusement_park <- amusement_park |>
  mutate(weekend = factor(x = weekend,
                          labels = c('no', 'yes'),
                          ordered = FALSE),
         num.child = as.integer(num.child),
         # logarithmic transform
         logdist = log(distance, base = exp(x = 1)))
amusement_park |> head(n = 5)
```

```
# A tibble: 5 x 9
  weekend num.child distance rides games  wait clean overall logdist
  <fct>       <int>    <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>   <dbl>
1 yes             0    115.     87    73    60    89      47    4.74
2 yes             2     27.0    87    78    76    87      65    3.30
3 no              1     63.3    85    80    70    88      61    4.15
4 yes             0     25.9    88    72    66    89      37    3.25
5 no              4     54.7    84    87    74    87      68    4.00
```

- Summarize data
  - Ups the table is really big!!! Try it in your console to see the complete table

```
amusement_park |> skim()
```

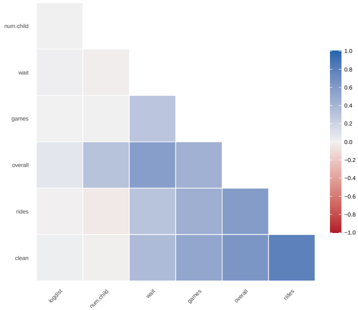
- Correlation matrices
  - Pearson correlation coefficients for samples in a tibble

```
correlation_matrix <- amusement_park |>
  select(num.child, rides:logdist) |>
  corrr::correlate()
correlation_matrix
```

```
# A tibble: 7 x 8
  term      num.child   rides    games    wait   clean overall  logdist
  <chr>         <dbl>   <dbl>    <dbl>   <dbl>   <dbl>   <dbl>    <dbl>
1 num.child  NA       -0.0403  0.00466 -0.0210 -0.0135  0.319  -0.00459
2 rides      -0.0403  NA        0.455    0.314   0.790   0.586  -0.0110
3 games       0.00466  0.455   NA        0.299   0.517   0.437   0.00187
4 wait       -0.0210   0.314    0.299   NA       0.368   0.573   0.0175
5 clean      -0.0135   0.790    0.517    0.368  NA       0.639   0.0221
6 overall     0.319    0.586    0.437    0.573   0.639  NA       0.0763
7 logdist    -0.00459 -0.0110   0.00187  0.0175  0.0221  0.0763 NA
```

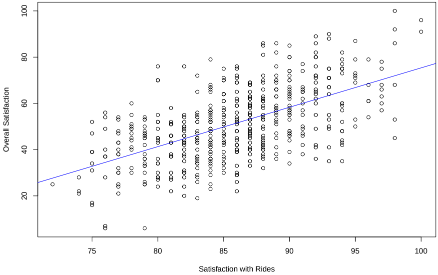
- Correlation matrices
  - Pearson correlation coefficients for samples in a tibble

```
correlation_matrix |> autoplot(triangular = "lower")
```



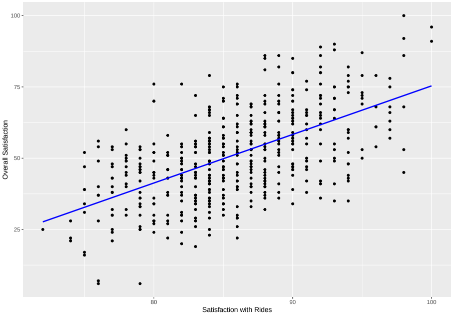
- **Bivariate Association: the base R way**

```
plot(overall~rides, data=amusement_park,
     xlab="Satisfaction with Rides", ylab="Overall Satisfaction")
abline(reg = lm(formula = overall~rides, data = amusement_park),
       col = 'blue')
```

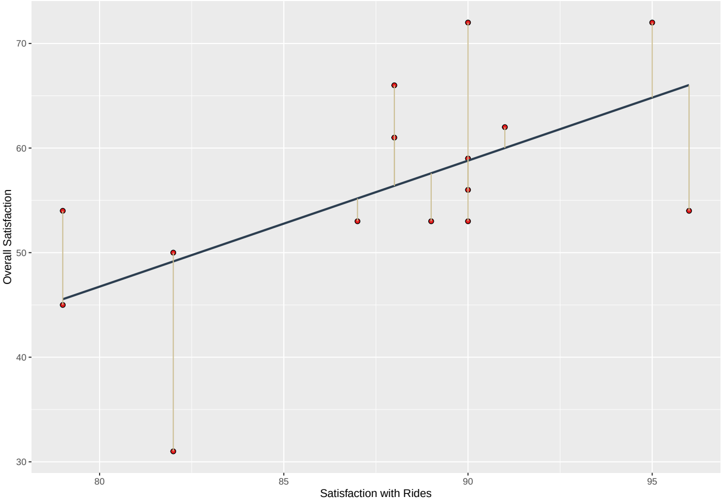


- **Bivariate Association: the tidyverse way**

```
amusement_park |> ggplot(aes(x = rides, y = overall)) +
  geom_point() +
  geom_smooth(method = 'lm',
              color = 'blue',
              se = FALSE) +
  labs(x = "Satisfaction with Rides",
       y = "Overall Satisfaction")
```



- Linear Model with a Single Predictor



- Linear Model with a Single Predictor

$$overall_i = \beta_0 + \beta_1 rides_i + \epsilon_i \text{ where } \epsilon_i \sim \mathcal{N}(0, \sigma^2) \text{ and } i = 1, \dots, 500$$

$$\widehat{overall}_i = \hat{\beta}_0 + \hat{\beta}_1 rides_i \text{ and } \hat{\sigma}^2 \text{ where } i = 1, \dots, 500$$

$$overall_i - \widehat{overall}_i = \hat{\epsilon}_i \text{ where } i = 1, \dots, 500$$

```
model1 <- lm(formula = overall ~ rides, data = amusement_park)
model1
```

```
Call:
lm(formula = overall ~ rides, data = amusement_park)

Coefficients:
(Intercept)        rides
    -94.962        1.703
```

- Linear Model with a Single Predictor

```r
ls.str(model1)
```

```
assign :  int [1:2] 0 1
call :  language lm(formula = overall ~ rides, data = amusement_park)
coefficients :  Named num [1:2] -95 1.7
df.residual :  int 498
effects :  Named num [1:500] -1146.2 -207.9 11.5 -17.9 20.3 ...
fitted.values :  Named num [1:500] 53.2 53.2 49.8 54.9 48.1 ...
model : 'data.frame':   500 obs. of  2 variables:
 $ overall: num  47 65 61 37 68 27 40 30 58 36 ...
 $ rides  : num  87 87 85 88 84 81 77 82 90 88 ...
qr : List of 5
 $ qr   : num [1:500, 1:2] -22.3607 0.0447 0.0447 0.0447 0.0447 ...
 $ qraux: num [1:2] 1.04 1.01
 $ pivot: int [1:2] 1 2
 $ tol  : num 1e-07
 $ rank : int 2
rank :  int 2
residuals :  Named num [1:500] -6.22 11.78 11.18 -17.93 19.89 ...
terms : Classes 'terms', 'formula'  language overall ~ rides
xlevels :  Named list()
```

- Linear Model with a Single Predictor

```r
summary(model1)
```

```
Call:
lm(formula = overall ~ rides, data = amusement_park)

Residuals:
    Min      1Q  Median      3Q     Max
-33.597 -10.048   0.425   8.694  34.699

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -94.9622     9.0790  -10.46   <2e-16 ***
rides         1.7033     0.1055   16.14   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 12.88 on 498 degrees of freedom
Multiple R-squared:  0.3434,	Adjusted R-squared:  0.3421
F-statistic: 260.4 on 1 and 498 DF,  p-value: < 2.2e-16
```

- Linear Model with a Single Predictor

```r
model1$coefficients
```

```
(Intercept)       rides
 -94.962246    1.703285
```

```r
# Make some predictions
# We want to forecast the overall satisfaction rating
# if the satisfaction with rides is 95
-94.962246 + 1.703285*95
```

```
[1] 66.84983
```

- Linear Model with a Single Predictor
  - Std. Error column
    - Indicates uncertainty in the coefficient estimate
    - We can build a confidence interval

```
summary(model1)$coefficients[, 2]
```

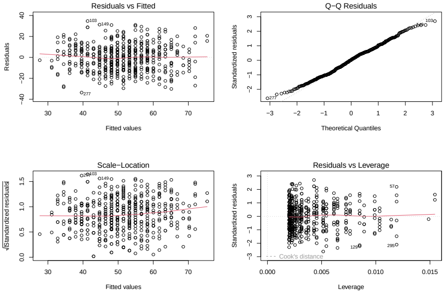
```
(Intercept)       rides
  9.0790049   0.1055462
```

```
confint(model1, level = 0.95)
```

```
                  2.5 %     97.5 %
(Intercept) -112.800120 -77.124371
rides          1.495915   1.910656
```

- Linear Model with a Single Predictor

```
par(mfrow=c(2,2))
plot(model1)
```




```
par(mfrow=c(1,1))
```

- Linear Model with a Single Predictor
    - **Linearity**: plot $(1, 1)$
        - Reference line should be flat and horizontal
    - **Normality of residuals**: plot $(1, 2)$
        - Dots should fall along the line
    - **Homogeneity of variance**: plot $(2, 1)$
        - Reference line should be flat and horizontal
    - **Influential observations**: plot $(2, 2)$
        - Points should be inside the contour lines

- Linear Model with Multiple Predictors

$$
\begin{aligned}
overall_i = \beta_0 + \beta_1 rides_i + \beta_2 games_i \\
+ \beta_3 wait_i + \beta_4 clean_i + \epsilon_i \\
\text{where } \epsilon_i \sim \mathcal{N}(0, \sigma^2) \text{ and } i = 1, \dots, 500
\end{aligned}
$$

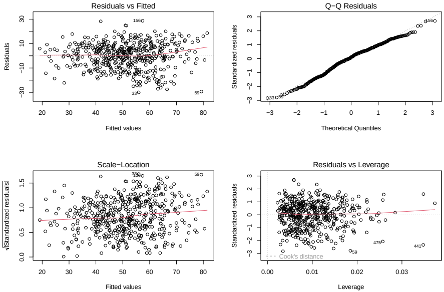
```r
model2 <- lm(formula = overall ~ rides + games + wait + clean,
             data = amusement_park)
model2
```

```
Call:
lm(formula = overall ~ rides + games + wait + clean, data = amusement_park)

Coefficients:
(Intercept)        rides        games         wait        clean
  -131.4092       0.5291       0.1533       0.5533       0.9842
```

- Linear Model with Multiple Predictors

```r
par(mfrow=c(2,2))
plot(model2)
```




```r
par(mfrow=c(1,1))
```

- Linear Model with Multiple Predictors

```r
summary(model2)
```

```
Call:
lm(formula = overall ~ rides + games + wait + clean, data = amusement_park)

Residuals:
    Min      1Q  Median      3Q     Max 
-29.944  -6.841   1.072   7.167  28.618 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -131.40919    8.33377 -15.768  < 2e-16 ***
rides          0.52908    0.14207   3.724 0.000219 ***
games          0.15334    0.06908   2.220 0.026903 *  
wait           0.55333    0.04781  11.573  < 2e-16 ***
clean          0.98421    0.15987   6.156 1.54e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 10.59 on 495 degrees of freedom
Multiple R-squared:  0.5586,	Adjusted R-squared:  0.5551 
F-statistic: 156.6 on 4 and 495 DF,  p-value: < 2.2e-16
```

- Linear Model with Multiple Predictors

$$H_0 : \beta_1 = 0$$

$$H_1 : \beta_1 \neq 0$$

$$t_{rides} = \frac{\hat{\beta}_1 - \beta_1}{\sqrt{\widehat{Var(\hat{\beta}_1)}}} = \frac{0.529078 - 0}{0.14207176} = 3.724019$$

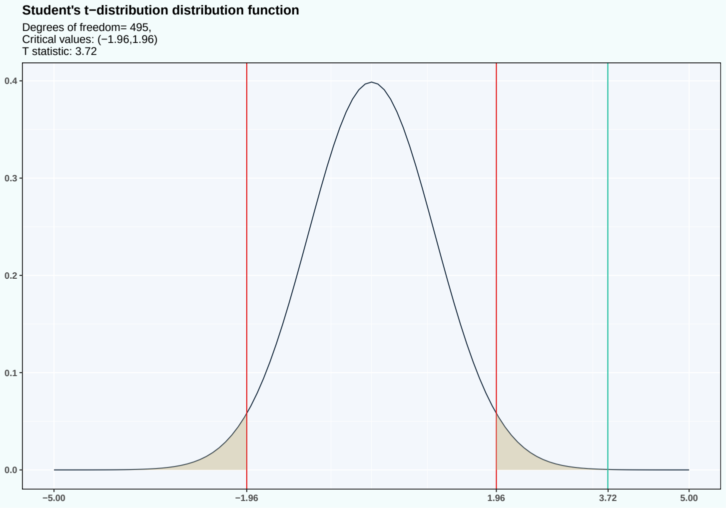
```r
model2$coefficients
```

```
 (Intercept)        rides        games         wait        clean
-131.4091939    0.5290780    0.1533361    0.5533264    0.9842126
```

```r
# Calculate the variance-covariance matrix, extract
# the diagonal and calculate the standard deviaton of
# the parameters
model2 |> vcov() |> diag() |> sqrt()
```

```
(Intercept)       rides       games        wait       clean
 8.33376643  0.14207176  0.06908486  0.04781282  0.15986712
```

- Linear Model with Multiple Predictors



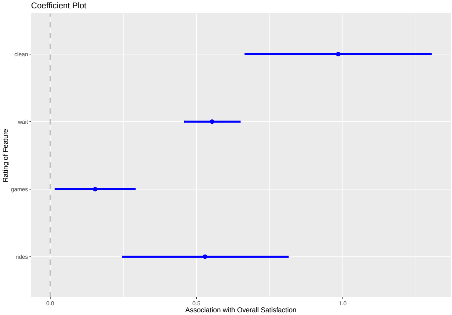
- Linear Model with Multiple Predictors

```
confint(model2, level = 0.95)
```

```
                   2.5 %       97.5 %
(Intercept) -147.78311147 -115.0352764
rides          0.24993998    0.8082161
games          0.01760038    0.2890718
wait           0.45938535    0.6472675
clean          0.67011082    1.2983144
```

- Linear Model with Multiple Predictors

```
library(coefplot) # Remember to install the package if it is not installed
coefplot(model = model2,
        # The intercept is relatively large: -131.4092
        intercept = FALSE,
        ylab="Rating of Feature",
        xlab="Association with Overall Satisfaction",
        lwdOuter = 1.5)
```



- Comparing models

```
summary(model1)$r.squared
```

```
[1] 0.3433799
```

```
summary(model2)$r.squared
```

```
[1] 0.558621
```

```
summary(model1)$adj.r.squared
```

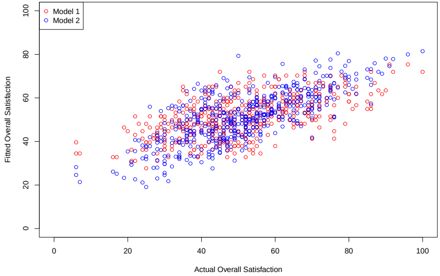
```
[1] 0.3420614
```

```
summary(model2)$adj.r.squared
```

```
[1] 0.5550543
```

- Comparing models
  - **Base R way**

```r
plot(x = amusement_park$overall, y = fitted(model1),
     col = "red", xlim = c(0,100), ylim = c(0,100),
     xlab = "Actual Overall Satisfaction",
     ylab = "Fitted Overall Satisfaction")
points(x = amusement_park$overall, y = fitted(model2),
       col = "blue")
legend(x = "topleft", legend = c("Model 1", "Model 2"), col = c("red", "blue"), pch = 1)
```



- Comparing models
  - **Tidymodels and tidyverse way**: Prepare data

```
model1_augment <- augment(x = model1) |> mutate(model = "Model 1")
model2_augment <- augment(x = model2) |> mutate(model = "Model 2")
models_performance <- model1_augment |> bind_rows(model2_augment)

models_performance |> glimpse()
```

```
Rows: 1,000
Columns: 12
$ overall    <dbl> 47, 65, 61, 37, 68, 27, 40, 30, 58, 36, 71, 48, 75, 46, 59,~
$ rides      <dbl> 87, 87, 85, 88, 84, 81, 77, 82, 90, 88, 93, 79, 94, 81, 86,~
$ .fitted    <dbl> 53.22359, 53.22359, 49.81702, 54.92688, 48.11373, 43.00388,~
$ .resid     <dbl> -6.2235914, 11.7764086, 11.1829795, -17.9268769, 19.8862650~
$ .hat       <dbl> 0.002089430, 0.002089430, 0.002048063, 0.002311576, 0.00222~
$ .sigma     <dbl> 12.88964, 12.88182, 12.88289, 12.86751, 12.86171, 12.87260,~
$ .cooksd    <dbl> 2.449537e-04, 8.770564e-04, 7.751689e-04, 2.249493e-03, 2.6~
$ .std.resid <dbl> -0.48371422, 0.91529407, 0.86915315, -1.39348008, 1.5457218~
$ model      <chr> "Model 1", "Model 1", "Model 1", "Model 1", "Model 1", "Mod~
$ games      <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
$ wait       <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
$ clean      <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
```

- Comparing models
  - **Tidymodels and tidyverse way**: Visualize

```
models_performance |>
  ggplot() +
  geom_point(aes(x = overall, y = .fitted,
                 color = model)) +
  labs(x = "Actual Overall Satisfaction",
       y = "Fitted Overall Satisfaction")
```

- Comparing models
  - Analysis of variance (anova) for nested models[1]

```r
anova_lm <- anova(model1, model2, test = "F")
anova_lm
```

```
Analysis of Variance Table

Model 1: overall ~ rides
Model 2: overall ~ rides + games + wait + clean
  Res.Df   RSS Df Sum of Sq      F    Pr(>F)
1    498 82612
2    495 55532  3     27080 80.463 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

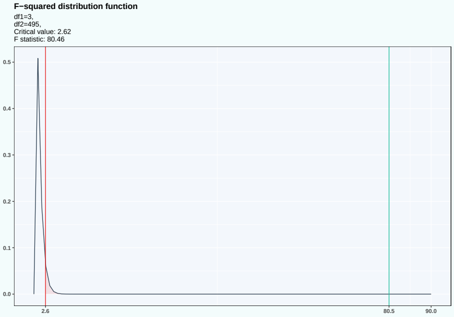
[1] This statistical analysis only make sense for nested models that are fitted with the same data where the convention is to include the models from smallest to largest. See `?anova.lm`

- Comparing models

$H_0 : \beta_0 = \beta_1 = \beta_2 = \beta_3 = \beta_4 = 0$

H_1 : At least one $\beta_j \neq 0$ for $j = 0, 1, 2, 3, 4$

$$F = \frac{\frac{RSS_1 - RSS_2}{p_2 - p_1}}{\frac{RSS_2}{n - p_2}} = \frac{\frac{82611.81 - 55531.53}{5-2}}{\frac{55531.53}{500-5}} = 80.46323$$



- Predictions

$$\widehat{overall}_j = \hat{\beta}_0 + \hat{\beta}_1 rides_j + \hat{\beta}_2 games_j + \hat{\beta}_3 wait_j + \hat{\beta}_4 clean_j$$

```r
coef(model2) |> enframe(name = "coef")
```

```
# A tibble: 5 x 2
  coef           value
  <chr>          <dbl>
1 (Intercept) -131.
2 rides          0.529
3 games          0.153
4 wait           0.553
5 clean          0.984
```

- Predictions
  - Manual

```r
(coef(model2)["(Intercept)"]*1 + coef(model2)["rides"]*30 + coef(model2)["games"]*10 +
    coef(model2)["wait"]*57 +  coef(model2)["clean"]*90) |>
  unname()
```

```
[1] 6.11525
```

- Predictions
  - Matrix multiplication

```r
coef(model2) %*% c(1, 30, 10, 57, 90)
```

```
        [,1]
[1,] 6.11525
```

- Predictions
  - predict

```
# New data
new_data <- tibble(rides = c(30, 70),
                   games = c(10, 80),
                   wait =  c(57, 60),
                   clean = c(90, 93))
# Result
predict(object = model2, newdata = new_data) |>
  enframe(name = "observation", value = "overall_pred") |>
  bind_cols(new_data)
```

```
# A tibble: 2 x 6
  observation overall_pred rides games  wait clean
  <chr>              <dbl> <dbl> <dbl> <dbl> <dbl>
1 1                   6.12    30    10    57    90
2 2                  42.6     70    80    60    93
```

- Standardizing the predictors
  - Compare the effect that different predictor variables have on a response variable
  - It must be interpreted in terms of standard deviations
    - One standard deviation in x variable is associated with a standard deviation increase of decrease depending on the value of the estimated parameter

```
amusement_park_std <- amusement_park |>
  select(-distance) |>
  mutate(across(rides:logdist,
                .fns = ~ scale(x = .x,
                               center = TRUE,
                               scale = TRUE)[,1]))
amusement_park_std |> head()
```

```
# A tibble: 6 x 8
  weekend num.child  rides   games     wait   clean overall logdist
  <fct>       <int>  <dbl>   <dbl>    <dbl>   <dbl>   <dbl>   <dbl>
1 yes             0  0.211 -0.698  -0.919    0.215   -0.268   1.79
2 yes             2  0.211 -0.0820  0.567   -0.176    0.865   0.323
3 no              1 -0.155  0.164   0.00966  0.0199   0.614   1.19
4 yes             0  0.394 -0.821  -0.362    0.215   -0.898   0.280
5 no              4 -0.338  1.03    0.381   -0.176    1.05    1.04
6 no              5 -0.887  0.0411 -2.03    -1.74    -1.53    0.145
```

- Standardizing the predictors

```r
model2_std <- lm(formula = overall ~ rides + games + wait + clean,
             data = amusement_park_std)
summary(model2_std)
```

```
Call:
lm(formula = overall ~ rides + games + wait + clean, data = amusement_park_std)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.88578 -0.43082  0.06749  0.45136  1.80231 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -2.607e-16  2.983e-02   0.000 1.000000    
rides        1.820e-01  4.888e-02   3.724 0.000219 ***
games        7.844e-02  3.534e-02   2.220 0.026903 *  
wait         3.753e-01  3.243e-02  11.573  < 2e-16 ***
clean        3.170e-01  5.150e-02   6.156 1.54e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.667 on 495 degrees of freedom
Multiple R-squared:  0.5586,	Adjusted R-squared:  0.5551 
F-statistic: 156.6 on 4 and 495 DF,  p-value: < 2.2e-16
```

- Using factors as predictors

```r
model3 <- lm(formula = overall ~ rides + games + wait + clean + weekend + logdist + num.child,
             data = amusement_park_std)
tidy(model3)
```

```
# A tibble: 8 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)  -0.373     0.0465    -8.01  8.41e-15
2 rides         0.213     0.0420     5.07  5.57e- 7
3 games         0.0707    0.0303     2.34  1.99e- 2
4 wait          0.381     0.0278    13.7   1.45e-36
5 clean         0.297     0.0441     6.72  4.89e-11
6 weekendyes   -0.0459    0.0514    -0.893 3.73e- 1
7 logdist       0.0647    0.0257     2.52  1.22e- 2
8 num.child     0.227     0.0171    13.3   1.37e-34
```

```r
glance(model3)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic   p.value    df logLik   AIC   BIC
      <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>  <dbl> <dbl> <dbl>
1     0.679         0.674 0.571      148. 5.97e-117     7  -425.  868.  906.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

- Using factors as predictors
  - Overall satisfaction is about the same regardless the number of children

```
amusement_park_std <- amusement_park_std |>
  mutate(num.child.factor = factor(num.child))
model4 <- lm(formula = overall ~ rides + games + wait + clean + weekend + logdist + num.child.factor,
             data = amusement_park_std)
tidy(model4) |> slice(1, 2, 8:12)
```

```
# A tibble: 7 x 5
  term              estimate std.error statistic  p.value
  <chr>                <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)         -0.691    0.0449    -15.4  7.00e-44
2 rides                0.223    0.0354      6.30 6.61e-10
3 num.child.factor1    1.02     0.0713     14.3  8.96e-39
4 num.child.factor2    1.04     0.0564     18.4  8.77e-58
5 num.child.factor3    0.980    0.0702     14.0  1.75e-37
6 num.child.factor4    0.932    0.0803     11.6  1.22e-27
7 num.child.factor5    1.00     0.104       9.66 2.50e-20
```

```
glance(model4)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic   p.value    df logLik   AIC   BIC
      <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>  <dbl> <dbl> <dbl>
1     0.775         0.770 0.480      153. 2.68e-150    11  -336.  698.  753.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

- Using factors as predictors
    - Preparing data

```
amusement_park_std <- amusement_park_std |>
  mutate(has.child = factor(x = num.child > 0, labels = c("No", "Yes")))
model5 <- lm(formula = overall ~ rides + games + wait + clean + logdist + has.child,
             data = amusement_park_std)
tidy(model5) |> slice(1, 2, 7)
```

```
# A tibble: 3 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)   -0.702    0.0391    -18.0  6.68e-56
2 rides          0.223    0.0351      6.34 5.12e-10
3 has.childYes   1.01     0.0468     21.5  1.08e-72
```

```
glance(model5)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic   p.value    df logLik   AIC   BIC
      <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>  <dbl> <dbl> <dbl>
1     0.774         0.771 0.478      282. 1.03e-155     6  -337.  690.  724.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

- Using factors as predictors
  - Maybe having children and the visits on weekends are important for the scores so an interaction will be useful

```
model6 <- lm(formula = overall ~ rides + games + wait + clean + weekend + logdist +
                                 has.child + rides:has.child + games:has.child + wait:has.child +
                                 clean:has.child + rides:weekend + games:weekend + wait:weekend +
                                 clean:weekend, data = amusement_park_std)
tidy(model6) |> slice(9:16)
```

```
# A tibble: 8 x 5
  term                estimate std.error statistic  p.value
  <chr>                  <dbl>     <dbl>     <dbl>    <dbl>
1 rides:has.childYes  0.0578      0.0731    0.792  4.29e- 1
2 games:has.childYes -0.0640      0.0528   -1.21   2.26e- 1
3 wait:has.childYes   0.351       0.0472    7.42   5.21e-13
4 clean:has.childYes -0.00185     0.0797   -0.0233 9.81e- 1
5 rides:weekendyes    0.0618      0.0678    0.912  3.62e- 1
6 games:weekendyes    0.0185      0.0490    0.377  7.06e- 1
7 wait:weekendyes     0.0352      0.0445    0.791  4.29e- 1
8 clean:weekendyes   -0.0273      0.0710   -0.385  7.01e- 1
```

```
glance(model6)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic   p.value    df logLik   AIC   BIC
      <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>  <dbl> <dbl> <dbl>
1     0.802         0.796 0.452      130. 3.69e-159    15  -304.  643.  714.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

- Using factors as predictors
  - Only an interaction was significant

```r
model7 <- lm(formula = overall ~ rides + games + wait + clean + logdist + has.child +
                       wait:has.child, data = amusement_park_std)
tidy(model7)
```

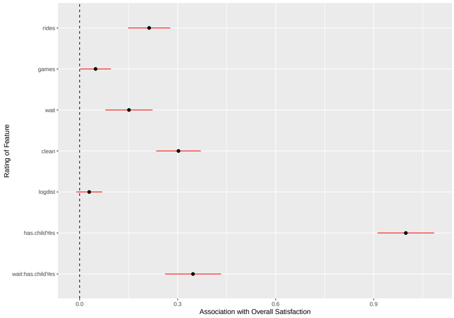
```
# A tibble: 8 x 5
  term              estimate std.error statistic  p.value
  <chr>                <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)        -0.693     0.0368    -18.8  6.91e-60
2 rides               0.213     0.0331      6.42 3.24e-10
3 games               0.0487    0.0239      2.03 4.25e- 2
4 wait                0.151     0.0369      4.09 4.98e- 5
5 clean               0.302     0.0349      8.68 5.94e-17
6 logdist             0.0292    0.0203      1.44 1.50e- 1
7 has.childYes        0.998     0.0442     22.6  4.02e-78
8 wait:has.childYes   0.347     0.0438      7.92 1.59e-14
```

```r
glance(model7)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic   p.value    df logLik   AIC   BIC
      <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>  <dbl> <dbl> <dbl>
1     0.800         0.797 0.451      280. 2.96e-167     7  -307.  632.  670.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

- Using factors as predictors
  - Final model

```
library(dotwhisker)  # Remember to install the package if it is not installed
tidy(model7) |>
  dwplot(ci = 0.95,
         dot_args = list(size = 2, color = "black"), whisker_args = list(color = "red"),
         vline = geom_vline(xintercept = 0, color = "black", linetype = 2)) +
  labs(x = "Association with Overall Satisfaction", y = "Rating of Feature")
```



- Formula syntax

Formula in R	Statistical Model
y ∼ x	$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$
y ∼ -1 + x	$y_i = \beta_1 x_i + \varepsilon_i$
y ∼ x + z	$y_i = \beta_0 + \beta_1 x_i + \beta_2 z_i + \varepsilon_i$
y ∼ x + z + x:z	$y_i = \beta_0 + \beta_1 x_i + \beta_2 z_i + \beta_3 x_i z_i + \varepsilon_i$
y ∼ x*z	$y_i = \beta_0 + \beta_1 x_i + \beta_2 z_i + \beta_3 x_i z_i + \varepsilon_i$
y ∼ (x + z + w)^2	$y_i = \beta_0 + \beta_1 x_i + \beta_2 z_i + \beta_3 w_i + \beta_4 x_i z_i + \beta_5 x_i w_i + \beta_6 w_i z_i + \varepsilon_i$
y ∼ (x + z + w)^2 - x:z	$y_i = \beta_0 + \beta_1 x_i + \beta_2 z_i + \beta_3 w_i + \beta_4 x_i w_i + \beta_5 w_i z_i + \varepsilon_i$
y ∼ x + I(x^2)	$y_i = \beta_0 + \beta_1 x_i + \beta_1 x_i^2 + \varepsilon_i$

- Try the following models using `tidy`:

```
lm(formula = overall ~ rides, data = amusement_park_std) |> tidy()
lm(formula = overall ~ -1 + rides, data = amusement_park_std) |> tidy()
lm(formula = overall ~ rides + has.child, data = amusement_park_std) |> tidy()
lm(formula = overall ~ rides + has.child + has.child, data = amusement_park_std) |> tidy()
lm(formula = overall ~ (rides + has.child + weekend)^2,
   data = amusement_park_std) |> tidy()
lm(formula = overall ~ (rides + has.child + weekend)^2 - rides:has.child,
   data = amusement_park_std) |> tidy()
lm(formula = overall ~ rides + I(rides^2) - rides:has.child, data = amusement_park_std) |> tidy()
```

- To my family that supports me
- To the taxpayers of Colombia and the **UMNG students** who pay my salary
- To the **Business Science** and **R4DS Online Learning** communities where I learn **R** and **π-thon**
- To the **R Core Team**, the creators of **RStudio IDE**, **Quarto** and the authors and maintainers of the packages **tidyverse**, **skimr**, **tidymodels**, **dotwhisker**, **kableExtra** and **tinytex** for allowing me to access these tools without paying for a license
- To the **Linux kernel community** for allowing me the possibility to use some **Linux distributions** as my main **OS** without paying for a license

References I

Chapman, Chris, and Elea McDonnell Feit. 2019. *R For Marketing Research and Analytics*. 2nd ed. 2019. Use R! Cham: Springer International Publishing : Imprint: Springer. https://doi-org.ezproxy.umng.edu.co/10.1007/978-3-030-14316-9.