Influencer Forecasting 2

Benjamin Jones

March 6, 2020

Note from the Author

This is an outline to creating forecasting models for influencer marketing. It is designed to be followed and replicated by one with the following prerequisite skills: programming and statistics or data science.

You may notice that some fundamental processes have not been performed, such as handling outliers or splitting into training and validation sets. This is due to having a very small sample size (85 records) and the nature of social media having large variance.

As you look over the results you will notice that even the best model are not textbook quality. By that I mean you likely won't find models of these caliber in any textbook as they all have at least one big downside. But this is the real world and not a homework problem. The real world throws a wave of problems: small sample size, inaccuracies, and inconsistencies. Adding in the high variance nature of the particular industry results in less than ideal models with large residuals.

With that being said, don't be discouraged but be realistic. Treat it as a balancing act. You can't have the best of everything, so make every adjustment meaningful and ensure all tradeoffs increase the overall effectiveness of the model. Key features to look at include homoscedasticity, R^2, and F/p-values. Aim for **parsimony** - go for the simplest answer with the greatest power.

Setup

```
library(data.world)
library(jsonlite)
library(MASS)
setwd("~/Desktop")
dw_auth <- read_json("dw_cred.json")
saved_cfg <- save_config(dw_auth$token)
set_config(saved_cfg)
```

Query

Dataset for this iteration is stored in data.world.

```
df <- query(qry_sql(paste0(
  "SELECT influencer_program_results_03042020.retailer_cat as RETAILER_CAT,
  influencer_program_results_03042020.prod_cat as PROD_CAT,
  influencer_program_results_03042020.prod_sub_cat as PROD_SUB_CAT,
  influencer_program_results_03042020.planned_impressions as IMPRESSIONS,
  influencer_program_results_03042020.budget as BUDGET,
  influencer_program_results_03042020.paid_flag as PAID,
  influencer_program_results_03042020.reach as REACH,
  influencer_program_results_03042020.engagements as ENGAGEMENTS,
  influencer_program_results_03042020.conversions as CONVERSIONS,
  influencer_program_results_03042020.notes as NOTES ",
```

```
  "FROM influencer_program_results_03042020")),
  dataset="https://data.world/benjones-mirum/basketballdemo")
```

Cleaning

Initial inspection can be done by manually investigating **df** or by **summary(df)**. This is a small enough dataset that manual inspection will suffice.

1. Convert characters to factors
2. Replace missing budget values (using mode)

Lastly verify that the only column with missing values is **NOTES**.

```
df$RETAILER_CAT <- as.factor(df$RETAILER_CAT)
df$PROD_CAT <- as.factor(df$PROD_CAT)
df$PROD_SUB_CAT <- as.factor(df$PROD_SUB_CAT)

mode_budget <- unique(df$BUDGET)[which.max(tabulate(match(df$BUDGET, unique(df$BUDGET))))]
df$BUDGET[is.na(df$BUDGET)] <- mode_budget

colnames(df)[colSums(is.na(df)) > 0]
```

```
## [1] "NOTES"
```

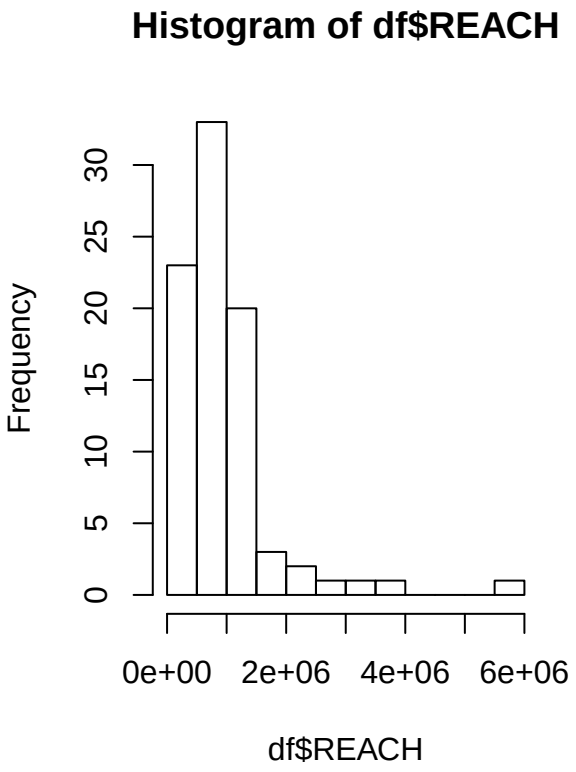
```
head(df)
```

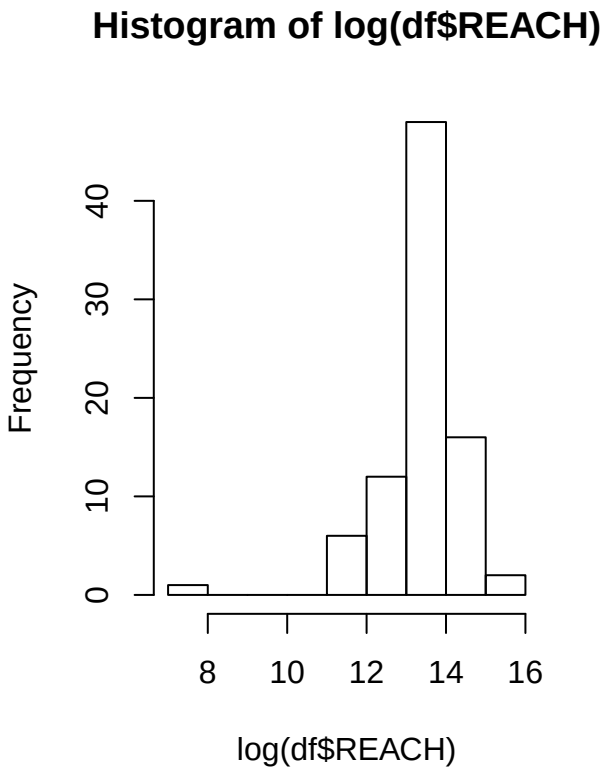
```
## # A tibble: 6 x 10
##   RETAILER_CAT PROD_CAT PROD_SUB_CAT IMPRESSIONS BUDGET PAID   REACH ENGAGEMENTS
##   <fct>        <fct>    <fct>              <dbl>  <dbl> <lgl>  <dbl>       <dbl>
## 1 national     persona~ personal_ca~    12000000  35000 FALSE 586136        9603
## 2 national     persona~ hair             5000000  35000 FALSE 537194        4169
## 3 national     persona~ deodorant       12000000  36896 FALSE 697605        1895
## 4 national     persona~ personal_ca~     5000000  35000 FALSE 512538        1763
## 5 national     persona~ lotion           5000000  30000 TRUE  433612        3818
## 6 national     persona~ lotion          12000000  37033 TRUE  865393        3487
## # ... with 2 more variables: CONVERSIONS <dbl>, NOTES <chr>
```

Reach

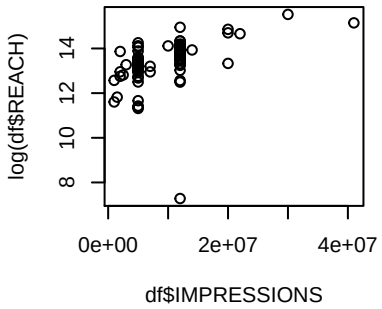
Looking at the distribution, it kind of resembles a normal distribution with the peak pushed to the left. This familiar shape indicates it might be **log-normal**, meaning the log of the variable is normally distributed. Plotting the log of reach shows a distribution that looks more normal. Please note that it is not required for the outcome variable to be normally distributed for regression, but it often times makes things easier.

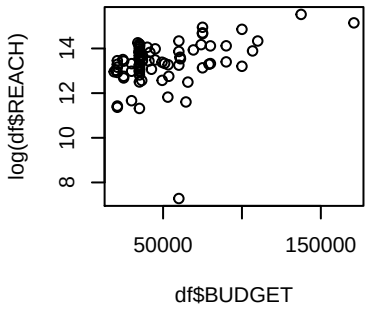
Note: code for plots will not be shown, but is available in the full script

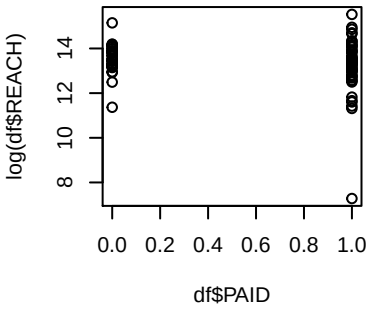


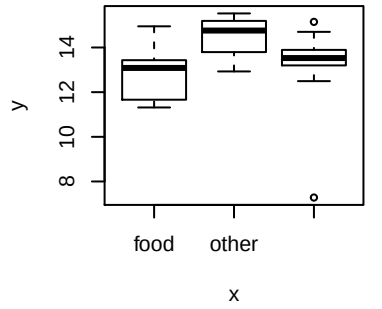


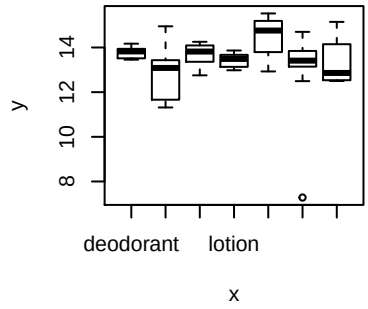
Now let's plot each possible dependent variable against **log(reach)** to gain clues on which variables may have a meaningful impact on reach:











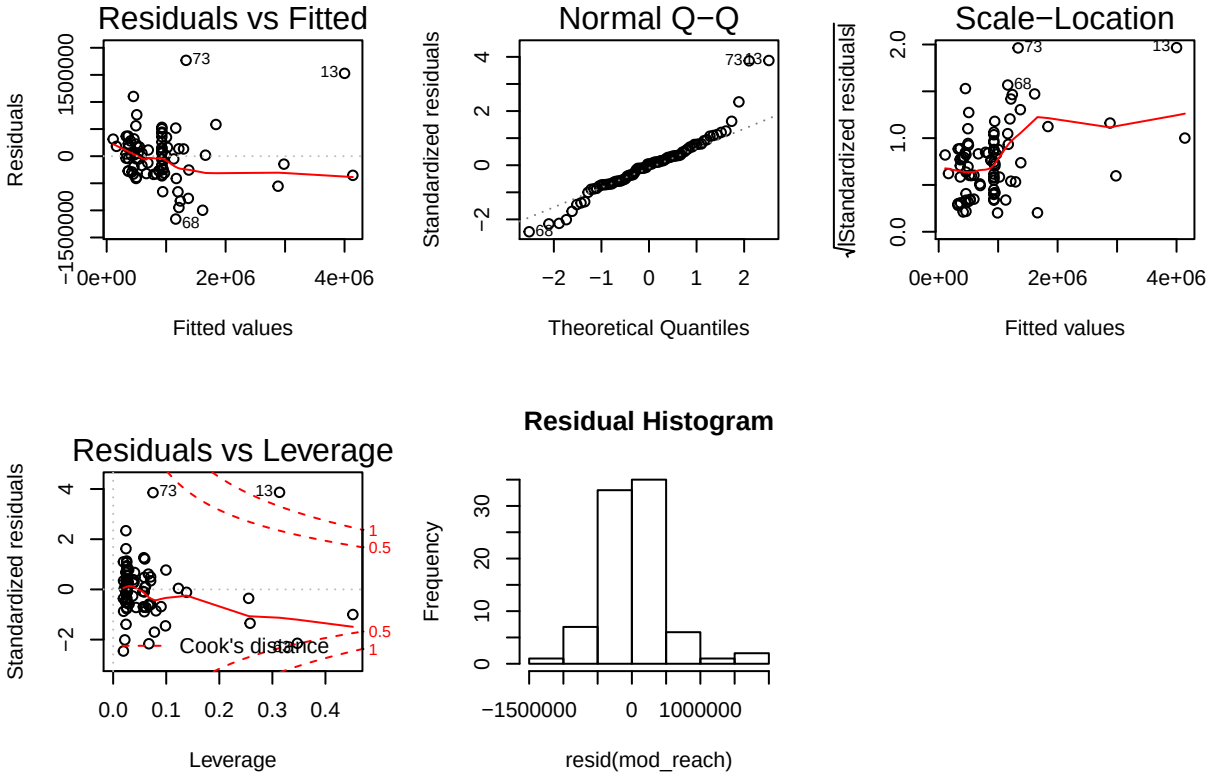
While not shown here, a lot of trial and error took place (which can be seen in the script). The general approach used here is **stepwise** - repeatedly adding and removing variables to see how it impacts the model. It is possible to do this procedure automatically, but with few variables I like to do it manually so I can see the results at each step.

You will notice that the better models dropped the variables which showed a weak relationship to reach in the scatterplots. A log transformation was also experimented with as it was noted earlier the log-normal shape of reach. However, it turns out that simplest model with the most explanatory power includes no transformations. Key features looked when comparing different models included homoscedasticity (constant variance of residuals), R^2, and F/p-values.

```
mod_reach <- lm(REACH ~ IMPRESSIONS+BUDGET+PROD_CAT, data=df)
summary(mod_reach)
```

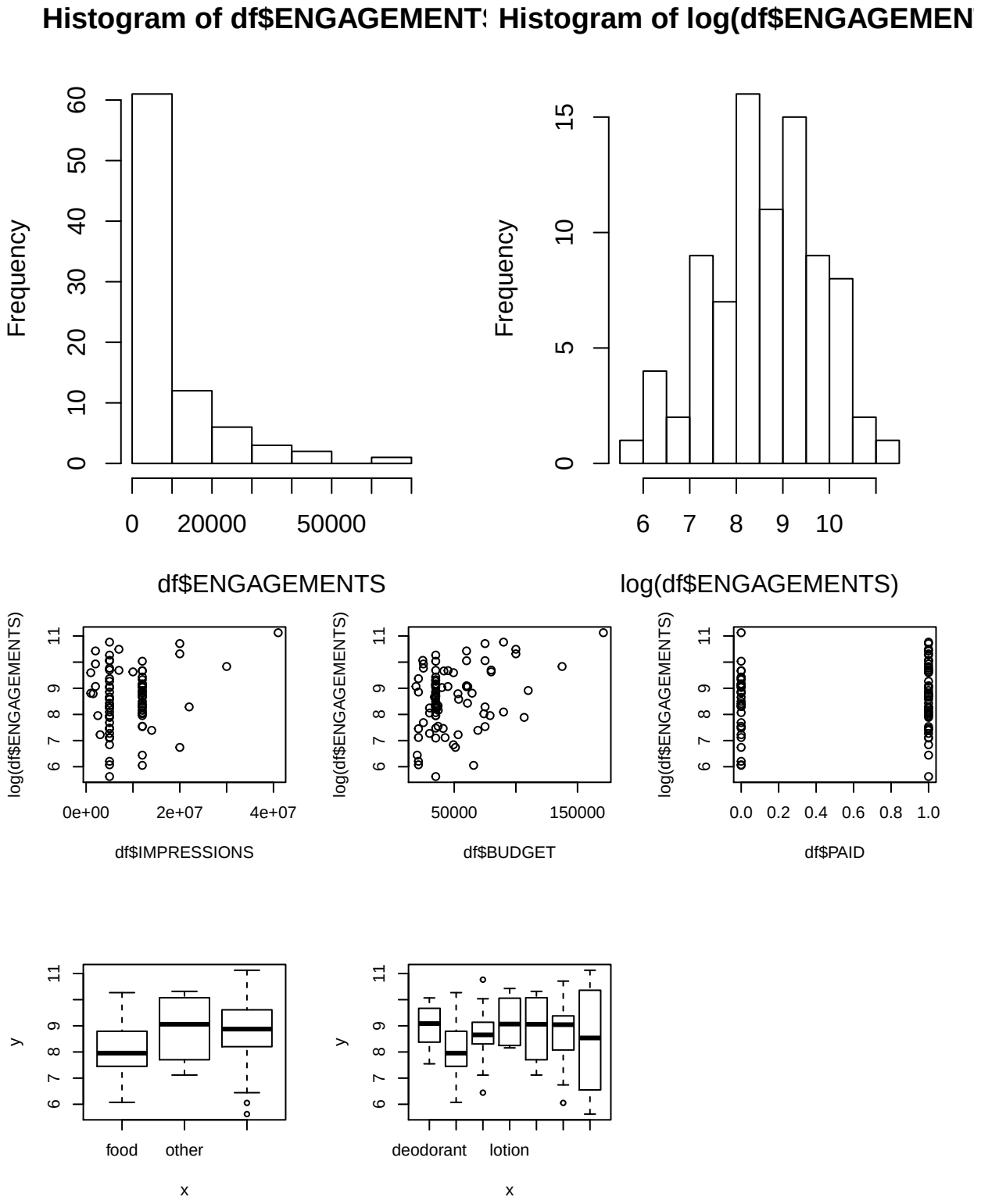
```
##
## Call:
## lm(formula = REACH ~ IMPRESSIONS + BUDGET + PROD_CAT, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1159994  -272682    19149   180841  1768078
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -1.679e+05  1.470e+05  -1.143 0.256651
## IMPRESSIONS            6.705e-02  1.077e-02   6.225 2.09e-08 ***
## BUDGET                 9.317e+00  2.292e+00   4.064 0.000112 ***
## PROD_CATother          8.758e+05  2.933e+05   2.986 0.003745 **
## PROD_CATpersonal_care -3.428e+04  1.350e+05  -0.254 0.800141
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 476900 on 80 degrees of freedom
## Multiple R-squared:  0.6856, Adjusted R-squared:  0.6699
## F-statistic: 43.61 on 4 and 80 DF,  p-value: < 2.2e-16
```

Visual plots are a helpful tool in evaluating and comparing models. The following displays a visual evaluation of the best model for reach:



Engagements

Building a model for engagements will follow in a similar fashion. Engagement also appears to have a log-normal distribution.



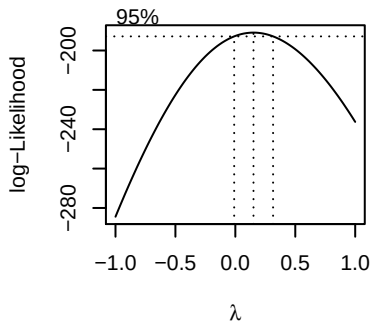
Just eyeballing the graphs, we can see there is definitely a much weaker relationship between engagements

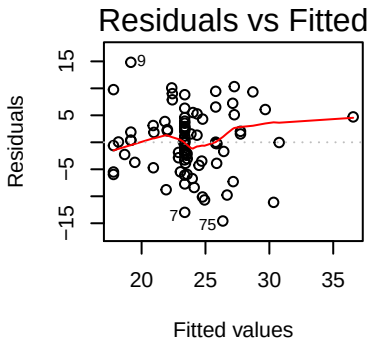
and the input variables. It is not surprising that choosing a model took a bit more experimentation, required a transformation, and ultimately had less explanatory power compared to reach. This makes sense as in the social media industry, the *lower-end funnel* metrics are often even more volatile.

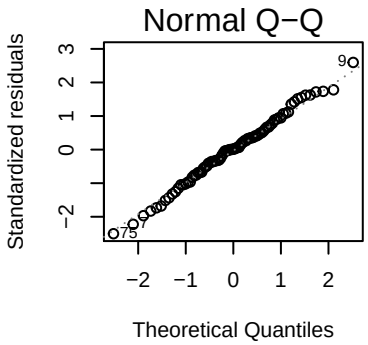
The transformation chosen for engagements was found via the **Box-Cox Transformation**. Essentially this considers a family of transformations and identifies λ which maximizes the log-likelihood.

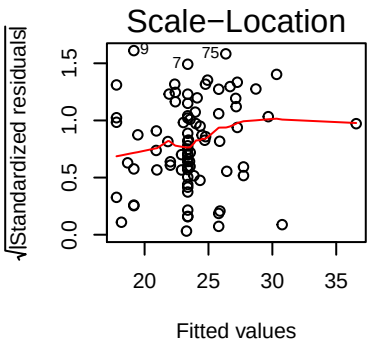
```
mod_engagements <- lm((((ENGAGEMENTS^0.2)-1)/0.2)  ~ BUDGET+PROD_CAT, data=df)

summary(mod_engagements)
```

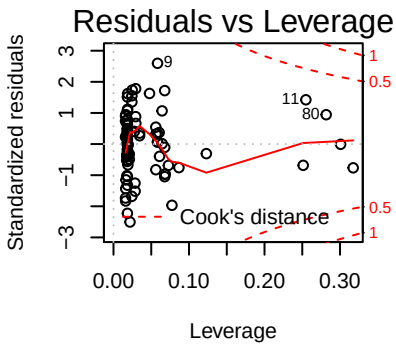
```
##
## Call:
## lm(formula = (((ENGAGEMENTS^0.2) - 1)/0.2) ~ BUDGET + PROD_CAT,
##     data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.5815  -3.4795   0.0678   3.5301  14.8252
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           1.576e+01  1.813e+00   8.696 3.18e-13 ***
## BUDGET                9.698e-05  2.450e-05   3.958 0.000161 ***
## PROD_CATother         1.670e+00  3.369e+00   0.496 0.621373
## PROD_CATpersonal_care 4.218e+00  1.573e+00   2.682 0.008874 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.885 on 81 degrees of freedom
## Multiple R-squared:  0.2218, Adjusted R-squared:  0.193
## F-statistic: 7.695 on 3 and 81 DF,  p-value: 0.000138
```

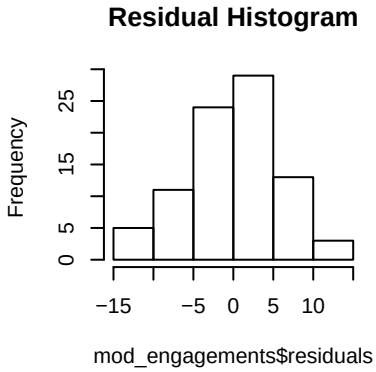

95%
log−Likelihood
−280
−240
−200
−1.0
−0.5
0.0
0.5
1.0
λ


Residuals vs Fitted
Residuals
15
5
−5
−15
9
7
75
20
25
30
35
Fitted values


Normal Q−Q
Standardized residuals
3
2
1
0
−2
9
75
7
−2
−1
0
1
2
Theoretical Quantiles

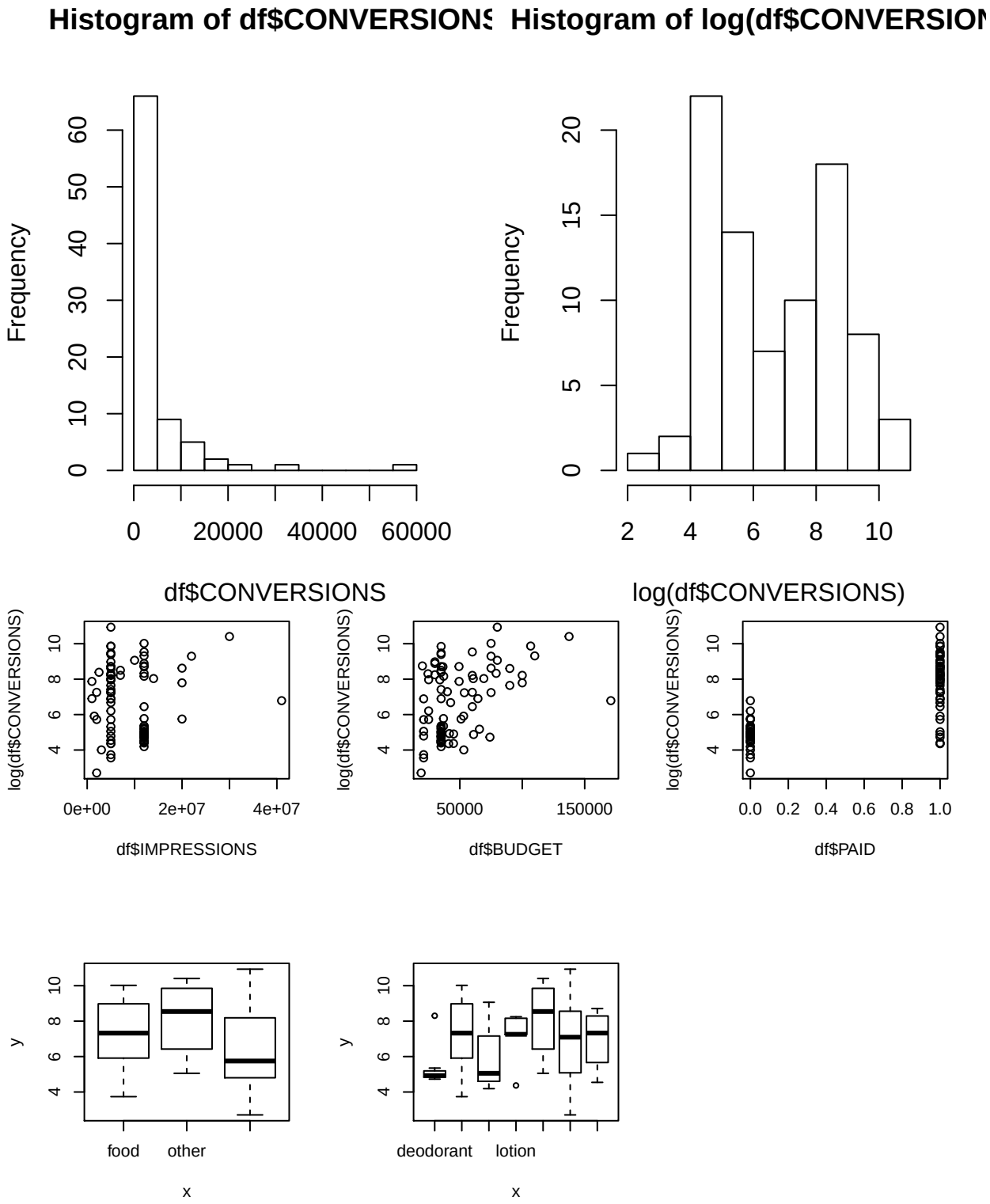

Scale−Location
√|Standardized residuals|
1.5
1.0
0.5
0.0
9
75
7
20
25
30
35
Fitted values


Residuals vs Leverage
Standardized residuals
3
2
1
−1
−3
9
11
80
1
0.5
0.5
1
Cook's distance
0.00
0.10
0.20
0.30
Leverage


Residual Histogram
Frequency
25
15
5
0
−15
−5
0
5
10
mod_engagements$residuals

Conversions

Lastly a model for conversions begins the same way:



From the plots above we can see that, more so than reach and engagements, **PAID** has a much stronger relationship with conversions. This makes sense, as majority of the time, paid amplification on social media is used specifically to increase link clicks (conversions). Given this we will likely see this variable be an

important independent variable when forecasting conversions.

```
mod_conversions <- lm(log(CONVERSIONS) ~ BUDGET+PAID, data=df)

summary(mod_conversions)
```

```
##
## Call:
## lm(formula = log(CONVERSIONS) ~ BUDGET + PAID, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4305 -0.5910  0.0892  0.8092  2.6791
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.247e+00  3.030e-01  14.015  < 2e-16 ***
## BUDGET      1.626e-05  5.326e-06   3.053  0.00305 **
## PAIDTRUE    2.706e+00  2.968e-01   9.119 4.17e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.273 on 82 degrees of freedom
## Multiple R-squared:  0.5934, Adjusted R-squared:  0.5834
## F-statistic: 59.83 on 2 and 82 DF,  p-value: < 2.2e-16
```