# Predicting Revenue from Search Engine Advertising Data

MATH2319 - Machine Learning
Course Project

*Ben Cole - s3412349*

*Print Date: 12/06/2019*

## Contents

# 1 Phase 1 - Introduction, Cleaning, and Exploration

## 1.1 Outline

The prescribed data set contained advertising metrics provided by a prominent search engine. The data contained several descriptive features pertaining to a range of information. Finally, the target feature was a measure of revenue associated with each of the observations.
The dataset was used to create a supervised machine learning model to predict values for the target feature. Phase 1 of this report contains the introduction, cleaning, and exploration of the dataset. Phase 2 contains the creation, training, and deployment of the machine learning algorithm.

### 1.1.1 Nature of the Data

The below is an exerpt from accompanying documentation about the dataset.

Features in this data set are as follows:

- companyId: Company ID of record (categorical)
- countryId: Country ID of record (categorical)
- deviceType: Device type of record (categorical corresponding to desktop, mobile, tablet)
- day: Day of record (integer between 1 (oldest) and 30 for train, 31 and 35 (most recent) for test)
- dow: Day of week of the record (categorical)
- price1, price2, price3: Price combination for the record set by the company (numeric)
- ad_area: area of advertisement (numeric)
- ad_ratio: ratio of advertisement's length to its width (numeric)
- requests, impression, cpc, ctr, viewability: Various metrics related to the record (numeric)
- ratio1, …, ratio5: Ratio characteristics related to the record (numeric)
- y (target feature): revenue-related metric (numeric)

#### 1.1.1.1 Target Feature

The column/variable **y** was selected as the target feature in the dataset.

#### 1.1.1.2 Descriptive Features

All other columns/variables in the dataset, as outlined above, were chosen as descriptive features.

## 1.2 Data Processing

### 1.2.1 Libraries

The following libraries were used in the below data processing and exploration.

```
library(pacman)                         ## for loading multiple packages

suppressMessages(p_load(character.only = T,
                        install = F,
                        c("tidyverse",  ## thanks Hadley
                          "lubridate",  ## for handling dates
                          "forcats",    ## for categorial variables, not for felines
                          "zoo",        ## some data cleaning capabilities
                          "lemon",      ## add ons for ggplot
                          "rvest",      ## scraping web pages
                          "knitr",      ## knitting to RMarkdown
                          "kableExtra", ## add ons for knitr tables
                          "scales",     ## quick and easy formatting prettynums
                          "grid",       ## for stacking ggplots
                          "gridExtra",  ## also for stacking ggplots
                          "e1071",      ## for skew and kurtosis
                          "janitor",    ## cleaning colnames
                          "beepr",      ## plays a beep tone
                          "mlr",
                          "FSelector")))
```

Table 1: Sample of Advertising Data Frame

| case_id | companyId | countryId | deviceType | day | dow | price1 | price2 | price3 | ad_area | ad_ratio |
|---|---|---|---|---|---|---|---|---|---|---|
| 165646 | 43 | 38 | 1 | 24 | Monday | 0.00 | 0.00 | 0.0000 | 6.5520 | 0.12363 |
| 288 | 40 | 77 | 2 | 1 | Saturday | 0.00 | 0.00 | 0.0000 | 0.0001 | 1.00000 |
| 112444 | 43 | 56 | 1 | 17 | Monday | 0.59 | 0.79 | 1.5610 | 3.2000 | 0.31250 |
| 2875 | 43 | 56 | 3 | 1 | Saturday | 0.00 | 0.00 | 0.0000 | 6.5520 | 0.12363 |
| 95494 | 43 | 202 | 1 | 14 | Friday | 0.44 | 0.80 | 1.6188 | 14.4000 | 0.62500 |
| 132826 | 43 | 56 | 3 | 19 | Wednesday | 0.00 | 0.00 | 0.0000 | 9.4080 | 0.83333 |
| 211690 | 43 | 202 | 2 | 30 | Sunday | 0.00 | 0.00 | 0.0000 | 18.0000 | 2.00000 |
| 73920 | 95 | 102 | 2 | 12 | Wednesday | 0.10 | 0.25 | 0.2500 | 0.0001 | 1.00000 |
| 26680 | 159 | 144 | 1 | 5 | Wednesday | 0.01 | 0.21 | 0.4111 | 7.5000 | 0.83333 |
| 147331 | 43 | 202 | 2 | 21 | Friday | 0.49 | 1.13 | 2.2698 | 6.5520 | 0.12363 |
| 70054 | 43 | 13 | 3 | 11 | Tuesday | 0.00 | 0.00 | 0.0000 | 7.5000 | 0.83333 |
| 76481 | 43 | 56 | 2 | 12 | Wednesday | 0.00 | 0.00 | 0.0000 | 24.2500 | 0.25773 |
| 111739 | 43 | 202 | 1 | 17 | Monday | 0.00 | 0.00 | 0.0000 | 3.2000 | 0.31250 |
| 104359 | 43 | 105 | 2 | 16 | Sunday | 0.00 | 0.00 | 0.0000 | 24.2500 | 0.25773 |
| 186218 | 159 | 57 | 3 | 26 | Wednesday | 0.12 | 0.45 | 0.9040 | 0.0001 | 1.00000 |
| 60595 | 43 | 55 | 2 | 10 | Monday | 0.00 | 0.00 | 0.0000 | 0.0001 | 1.00000 |
| 200946 | 159 | 200 | 2 | 28 | Friday | 0.00 | 0.00 | 0.0000 | 0.0001 | 1.00000 |
| 4001 | 159 | 197 | 2 | 1 | Saturday | 0.00 | 0.00 | 0.0000 | 0.0001 | 1.00000 |
| 214107 | 43 | 234 | 1 | 30 | Sunday | 0.00 | 0.00 | 0.0000 | 0.0001 | 1.00000 |
| 39696 | 95 | 38 | 1 | 7 | Friday | 1.18 | 2.76 | 5.5200 | 1.6000 | 0.15625 |

### 1.2.2 Loading Data

The prescribed data was made available in comma separated value file format.

```
advertising_train <- read_csv("advertising_train.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   dow = col_character()
## )

## See spec(...) for full column specifications.
```

```
sample_adv <- sample_n(advertising_train, 20)

kable_styling(kable(sample_adv[ , 1:(ncol(sample_adv)/2)],
                    caption = "Sample of Advertising Data Frame"),
              font_size = 8.5, latex_options = c("striped"),
              full_width = F)
```

```
kable_styling(kable(sample_adv[ , c(1, ((ncol(sample_adv)/2)+1):ncol(sample_adv))],
                    caption = "Sample of Advertising Data Frame (cont)"),
              font_size = 8.5, latex_options = c("striped"),
              full_width = F)
```

### 1.2.3 Classifying Data

R and `dplyr` parse data files to guessed data types when loaded. Typically, columns with text are parsed as `character` type, columns with digits are parsed as `numeric`, and boolean columns are parsed as `logical`. Per the above feature definitions, the categorical data was re-classified as `factors`.

```
advertising_train$companyId <- as.factor(advertising_train$companyId)

advertising_train$countryId <- as.factor(advertising_train$countryId)
```

Table 2: Sample of Advertising Data Frame (cont)

| case_id | requests | impression | cpc | ctr | viewability | ratio1 | ratio2 | ratio3 | ratio4 | ratio5 | y |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 165646 | 58 | 58 | 0.2810 | 0.0517 | 0.1154 | 1.0000 | 0.9483 | 0.0000 | 0.6034 | 0.3793 | 14.8538462 |
| 288 | 29161 | 22850 | 0.0636 | 0.0003 | 0.2963 | 1.0000 | 0.9737 | 1.0002 | 0.0000 | 0.0000 | 0.0182470 |
| 112444 | 0 | 0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.5344828 |
| 2875 | 0 | 0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 1.7000000 |
| 95494 | 1970 | 1683 | 0.1141 | 0.0089 | 0.7474 | 0.6423 | 0.9584 | 0.0784 | 0.1040 | 0.8176 | 0.8478098 |
| 132826 | 3516 | 3513 | 0.1862 | 0.0043 | 0.6630 | 1.0000 | 0.9004 | 0.0279 | 0.2622 | 0.7111 | 0.8159578 |
| 211690 | 0 | 0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.5029605 |
| 73920 | 0 | 0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.1111111 |
| 26680 | 1317 | 1147 | 0.0381 | 0.0026 | 0.4264 | 0.3976 | 0.9991 | 0.0567 | 0.1953 | 0.7489 | 0.0945137 |
| 147331 | 0 | 0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.3098417 |
| 70054 | 68 | 55 | 0.1543 | 0.0182 | 0.8235 | 1.0000 | 0.6909 | 0.0000 | 0.5455 | 0.4545 | 2.3250000 |
| 76481 | 0 | 0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.2605828 |
| 111739 | 794 | 793 | 0.1364 | 0.0038 | 0.7631 | 1.0000 | 0.9420 | 0.0845 | 0.1223 | 0.7932 | 0.5020378 |
| 104359 | 0 | 0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0298861 |
| 186218 | 2189 | 2175 | 0.1275 | 0.0097 | 0.8476 | 0.9747 | 0.8837 | 0.0441 | 0.5467 | 0.4097 | 1.1674568 |
| 60595 | 0 | 0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0500000 |
| 200946 | 719 | 703 | 0.0344 | 0.0028 | 0.5385 | 1.0000 | 0.9260 | 1.0000 | 0.0000 | 0.0000 | 0.1007018 |
| 4001 | 0 | 0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.3640000 |
| 214107 | 280 | 255 | 0.1430 | 0.0157 | 0.3630 | 1.0000 | 0.3882 | 0.0745 | 0.4980 | 0.4275 | 1.9464883 |
| 39696 | 15663 | 4181 | 0.6634 | 0.0029 | 0.5794 | 0.5834 | 0.8560 | 0.0440 | 0.3028 | 0.6532 | 0.5599659 |

```r
advertising_train$deviceType <- as.factor(advertising_train$deviceType)

advertising_train$dow <- as.factor(advertising_train$dow)

sapply(advertising_train, class)
```

```
##     case_id   companyId   countryId  deviceType         day         dow
##   "numeric"    "factor"    "factor"    "factor"   "numeric"    "factor"
##      price1      price2      price3     ad_area    ad_ratio    requests
##   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"
##  impression         cpc         ctr viewability      ratio1      ratio2
##   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"
##      ratio3      ratio4      ratio5           y
##   "numeric"   "numeric"   "numeric"   "numeric"
```

### 1.2.4 Descriptive Statistics

#### 1.2.4.1 Numeric Features

The below table outlines basic descriptive statistics about the centre and spread of the data for each of the numeric descriptive features, and numeric target feature. This table indicates that the numeric features each had distributions of different shapes and locations.

```r
advertising_train_long_num <- select(advertising_train,
                                     colnames(advertising_train),
                                     -case_id, -countryId,
                                     -companyId, -deviceType,
                                     -dow)

advertising_train_long_num <- gather(advertising_train_long_num,
                                     key = "Variable",
                                     value = "Value")

summary_adv_num <- summarise(group_by(advertising_train_long_num,
```

Table 3: Summary Statistics of Numeric Variables

| Variable | Mean | Std Dev | Min | Q1 | Median | Q3 | Max | Number of NA |
|---|---|---|---|---|---|---|---|---|
| ad_area | 4.724 | 6.273 | 0.000 | 0.000 | 0.000 | 7.500 | 36.000 | 0.000 |
| ad_ratio | 0.923 | 0.482 | 0.083 | 0.833 | 1.000 | 1.000 | 5.000 | 0.000 |
| cpc | 0.178 | 0.707 | 0.000 | 0.000 | 0.016 | 0.125 | 132.534 | 0.000 |
| ctr | 0.033 | 0.093 | 0.000 | 0.000 | 0.002 | 0.012 | 2.000 | 0.000 |
| day | 15.791 | 8.386 | 1.000 | 9.000 | 16.000 | 23.000 | 30.000 | 0.000 |
| impression | 5,585.714 | 98,713.340 | 0.000 | 0.000 | 99.000 | 1,058.000 | 6,100,324.000 | 0.000 |
| price1 | 0.438 | 1.281 | 0.000 | 0.000 | 0.010 | 0.190 | 14.690 | 0.000 |
| price2 | 0.630 | 1.482 | 0.000 | 0.000 | 0.090 | 0.570 | 63.120 | 0.000 |
| price3 | 0.932 | 1.840 | 0.000 | 0.000 | 0.295 | 0.986 | 78.900 | 0.000 |
| ratio1 | 0.558 | 0.447 | 0.000 | 0.000 | 0.750 | 1.000 | 1.000 | 0.000 |
| ratio2 | 0.491 | 0.414 | 0.000 | 0.000 | 0.627 | 0.896 | 1.027 | 0.000 |
| ratio3 | 0.312 | 0.444 | 0.000 | 0.000 | 0.028 | 1.000 | 1.500 | 0.000 |
| ratio4 | 0.131 | 0.240 | 0.000 | 0.000 | 0.000 | 0.164 | 1.077 | 0.000 |
| ratio5 | 0.188 | 0.297 | 0.000 | 0.000 | 0.000 | 0.385 | 1.200 | 0.000 |
| requests | 8,678.997 | 122,347.229 | 0.000 | 0.000 | 147.000 | 1,633.000 | 6,701,924.000 | 0.000 |
| viewability | 0.378 | 0.366 | 0.000 | 0.000 | 0.332 | 0.716 | 7.000 | 0.000 |
| y | 0.847 | 1.391 | 0.000 | 0.150 | 0.419 | 0.959 | 47.060 | 0.000 |

```
                                             Variable),
                                    "Mean" = mean(Value, na.rm = T),
                                    "Std Dev" = sd(Value, na.rm = T),
                                    "Min" = min(Value, na.rm = T),
                                    "Q1" = quantile(Value, 0.25, na.rm = T),
                                    "Median" = median(Value, na.rm = T),
                                    "Q3" = quantile(Value, 0.75, na.rm = T),
                                    "Max" = max(Value, na.rm = T),
                                    "Number of NA" = sum(is.na(Value)))

kable_styling(kable(summary_adv_num,
                    digits = 3, format.args = list(nsmall = 3,
                                                   scientific = F,
                                                   big.mark = ","),
                    caption = "Summary Statistics of Numeric Variables"),
              font_size = 8.5, latex_options = c("striped"),
              full_width = F)
```

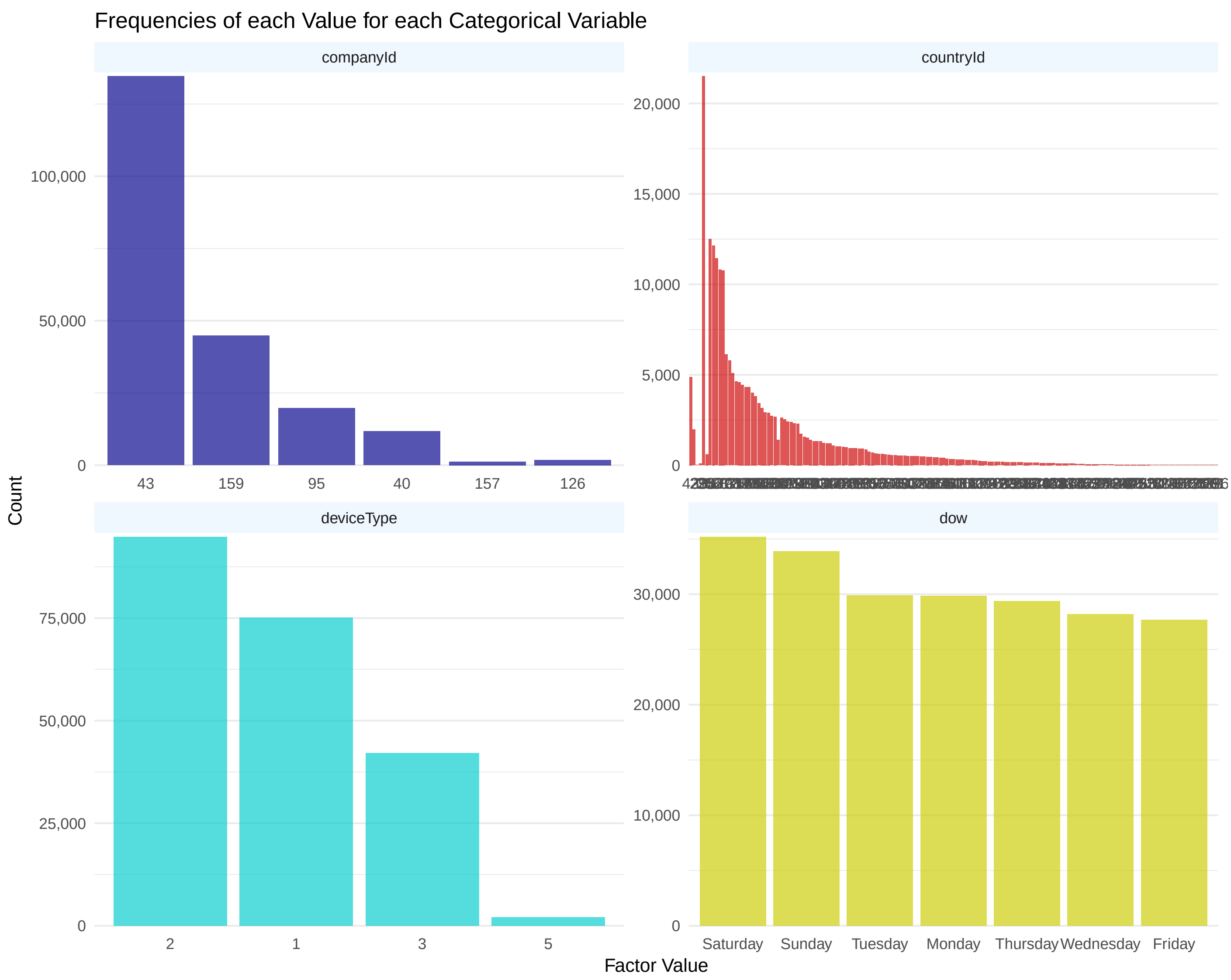
#### 1.2.4.2 Categorical and Non-Numeric Features

When examining the frequencies of individual levels of each Categorical (non-numeric) descriptive feature, variability was observed in `companyId`, `countryId`, and `deviceType`. Far less variability in frequencies was observed in `dow`, with Sunday being the only day of the week to return a markedly lower frequency.

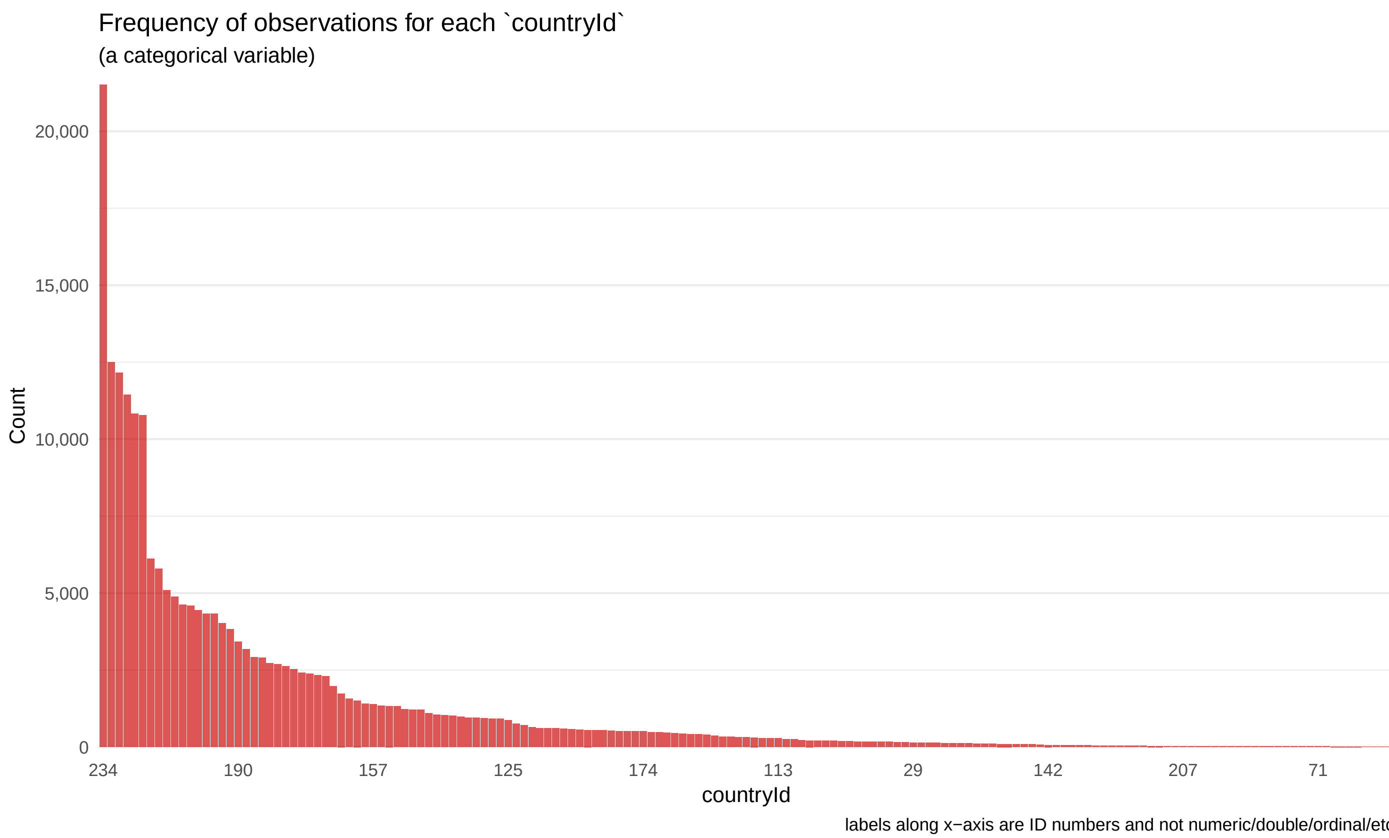
```
advertising_train_long_cat <- select(advertising_train,
                                     countryId,
                                     companyId, deviceType,
                                     dow)

advertising_train_long_cat <- gather(advertising_train_long_cat,
                                     key = "Variable",
                                     value = "Value")
```

```
## Warning: attributes are not identical across measure variables;
## they will be dropped
```

```r
advertising_train_long_cat$Variable <- as.factor(advertising_train_long_cat$Variable)

advertising_train_long_cat$Value <- as.factor(advertising_train_long_cat$Value)

ggplot(advertising_train_long_cat) +
  geom_bar(aes(x = fct_infreq(Value),
               fill = Variable),
           show.legend = F, alpha = 2/3) +
  facet_rep_wrap(~Variable,
                 repeat.tick.labels = T,
                 scales = "free") +
  scale_y_continuous(labels = comma,
                     expand = c(0.01, 0),
                     "Count") +
  scale_x_discrete("Factor Value") +
  scale_fill_manual(values = c("blue4", "red3", "cyan3", "yellow3")) +
  labs(title = "Frequencies of each Value for each Categorical Variable") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
        strip.background = element_rect(fill = "aliceblue",
                                        colour = NA))
```



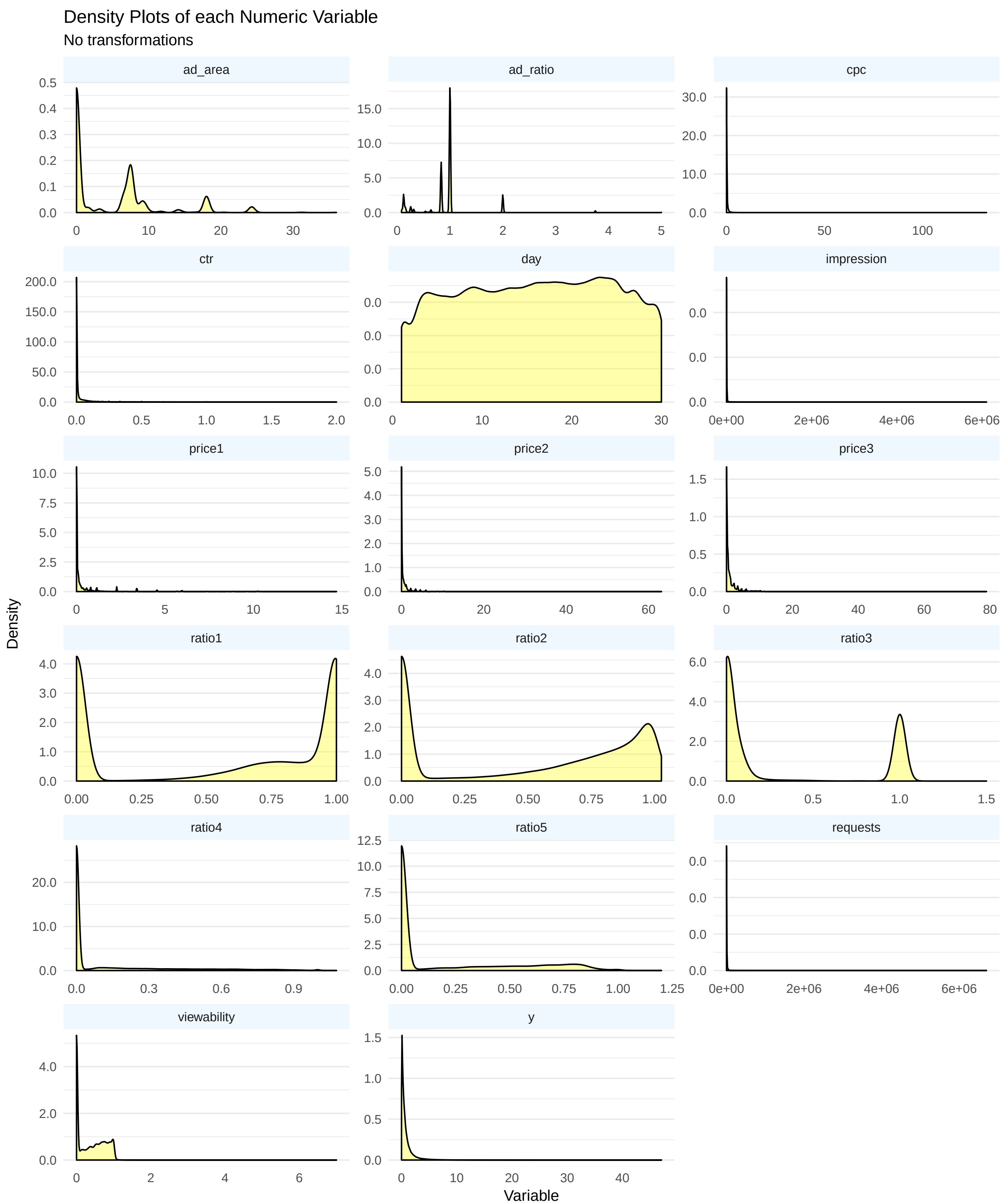

```
country_labels <- levels(fct_infreq(advertising_train$countryId))[c(seq(1,
                                                                   length(levels(fct_infreq(adv
                                                                   ceiling(length(levels(fct_in

ggplot(advertising_train) +
  geom_bar(aes(x = fct_infreq(countryId)),
           fill = "red3", alpha = 2/3) +
  scale_y_continuous(labels = comma,
                     expand = c(0.01, 0),
                     "Count") +
  scale_x_discrete(breaks = country_labels,
                   "countryId") +
  labs(title = "Frequency of observations for each \`countryId\`",
       subtitle = "(a categorical variable)",
       caption = "labels along x-axis are ID numbers and not numeric/double/ordinal/etc") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())
```


Frequency of observations for each `countryId`
(a categorical variable)
Count
20,000
15,000
10,000
5,000
0
234
190
157
125
174
113
29
142
207
71
countryId
labels along x−axis are ID numbers and not numeric/double/ordinal/etc

### 1.2.5 Univariate Plots

#### 1.2.5.1 Numeric Variables

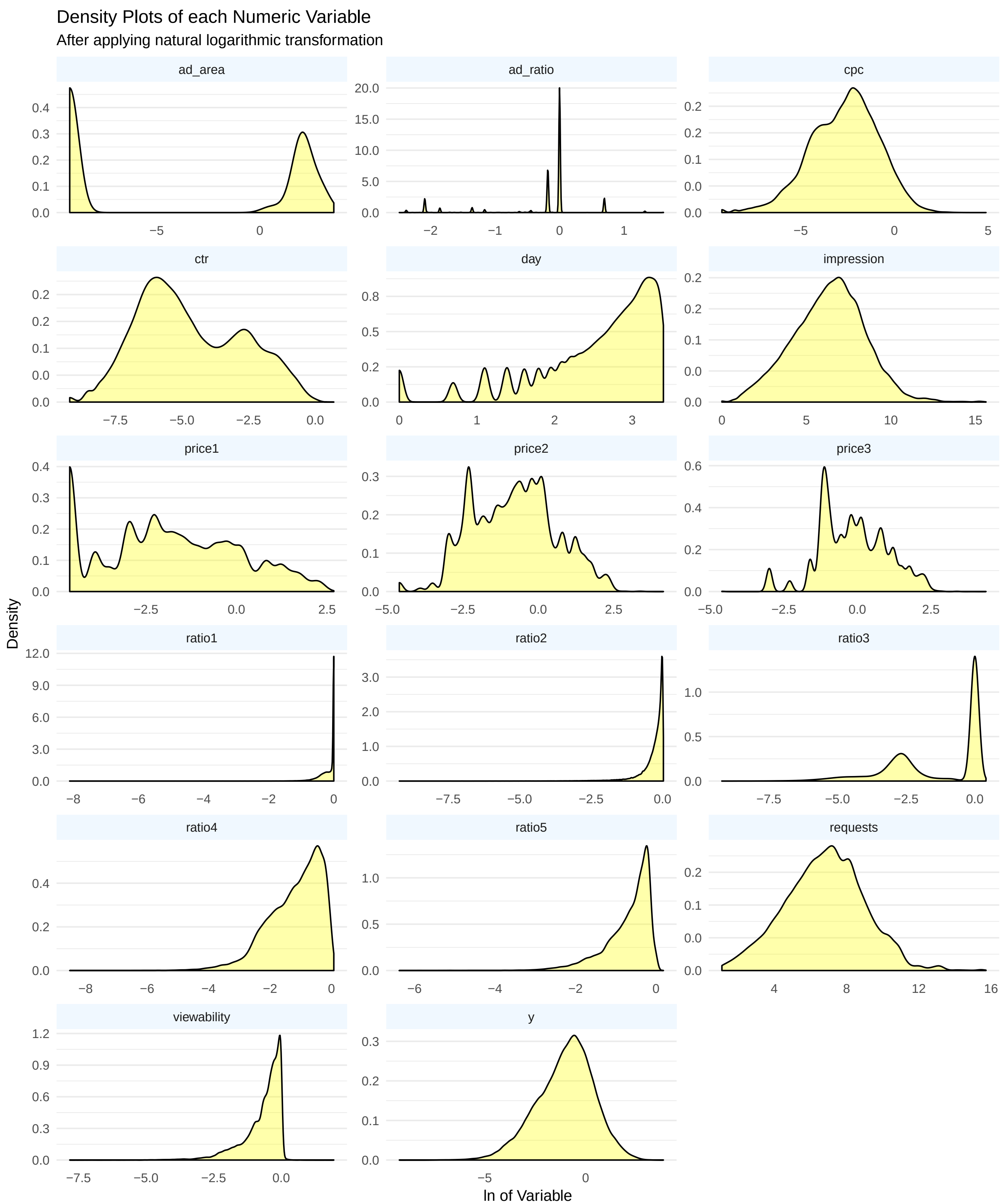
```
ggplot(advertising_train_long_num) +
  geom_density(aes(x = Value),
               fill = "yellow",
               alpha = 1/3) +
  facet_rep_wrap(~Variable,
                 repeat.tick.labels = T,
                 scales = "free",
                 ncol = 3) +
  scale_y_continuous(labels = comma_format(accuracy = 0.1)) +
  labs(title = "Density Plots of each Numeric Variable",
       subtitle = "No transformations",
       x = "Variable",
       y = "Density")+
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
        strip.background = element_rect(fill = "aliceblue",
                                        colour = NA))
```




```r
ggplot(advertising_train_long_num) +
  geom_density(aes(x = log(Value)),
               fill = "yellow",
               alpha = 1/3) +
```

```r
  facet_rep_wrap(~Variable,
                 repeat.tick.labels = T,
                 scales = "free",
                 ncol = 3) +
  scale_y_continuous(labels = comma_format(accuracy = 0.1)) +
  labs(title = "Density Plots of each Numeric Variable",
       subtitle = "After applying natural logarithmic transformation",
       x = "ln of Variable",
       y = "Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
        strip.background = element_rect(fill = "aliceblue",
                                        colour = NA))
```

```
## Warning: Removed 1213004 rows containing non-finite values (stat_density).
```




#### 1.2.5.2 Logarithmic Transformations

It was observed from the plots above that natural logarithmic transformations were applicable for descriptive features `cpc`, `impression`, and potentially `ctr`. Target feature `y` was also suitable for a logarithmic transformation.

Table 4: Sample of advertising_train Data Frame After Logarithmic Transformations

| case_id | companyId | countryId | deviceType | day | dow | price1 | price2 | price3 | ad_area | ad_ratio | requests | impression |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 146984 | 43 | 139 | 2 | 21 | Friday | 0.00 | 0.00 | 0.000 | 0.0001 | 1.000 | 0 | 0 |
| 182164 | 159 | 98 | 1 | 26 | Wednesday | 0.00 | 0.00 | 0.000 | 7.5000 | 0.833 | 5591 | 5507 |
| 1103 | 43 | 56 | 3 | 1 | Saturday | 0.00 | 0.00 | 0.000 | 24.2500 | 0.258 | 0 | 0 |
| 79692 | 43 | 19 | 2 | 12 | Wednesday | 0.00 | 0.00 | 0.000 | 0.0001 | 1.000 | 69 | 64 |
| 31797 | 43 | 101 | 1 | 6 | Thursday | 0.00 | 0.00 | 0.000 | 6.5520 | 0.124 | 382 | 382 |
| 176490 | 43 | 103 | 2 | 25 | Tuesday | 0.00 | 0.00 | 0.000 | 0.0001 | 1.000 | 47 | 40 |
| 30218 | 43 | 127 | 1 | 5 | Wednesday | 0.01 | 0.14 | 0.423 | 7.5000 | 0.833 | 0 | 0 |
| 53528 | 159 | 144 | 2 | 9 | Sunday | 0.02 | 0.15 | 0.283 | 0.0001 | 1.000 | 0 | 0 |
| 123731 | 159 | 13 | 1 | 18 | Tuesday | 0.26 | 0.88 | 1.762 | 0.0001 | 1.000 | 277 | 256 |
| 52813 | 43 | 56 | 2 | 9 | Sunday | 0.00 | 0.00 | 0.000 | 0.0001 | 1.000 | 98 | 98 |
| 210661 | 43 | 57 | 3 | 30 | Sunday | 0.00 | 0.00 | 0.000 | 7.5000 | 0.833 | 12 | 11 |
| 4284 | 43 | 59 | 2 | 1 | Saturday | 5.99 | 5.99 | 5.988 | 0.0001 | 1.000 | 5572 | 269 |
| 65107 | 159 | 57 | 3 | 10 | Monday | 0.14 | 0.51 | 1.032 | 0.0001 | 1.000 | 382 | 381 |
| 16126 | 43 | 234 | 1 | 3 | Monday | 0.02 | 0.82 | 1.642 | 0.0001 | 1.000 | 6180 | 5115 |
| 151997 | 43 | 56 | 2 | 22 | Saturday | 0.03 | 0.92 | 1.838 | 24.2500 | 0.258 | 268 | 268 |
| 169363 | 43 | 202 | 2 | 24 | Monday | 0.82 | 1.01 | 2.019 | 14.0400 | 0.641 | 62925 | 36334 |
| 30335 | 159 | 144 | 3 | 5 | Wednesday | 0.00 | 0.00 | 0.000 | 0.0001 | 1.000 | 0 | 0 |
| 84792 | 43 | 179 | 1 | 13 | Thursday | 0.14 | 0.36 | 0.723 | 7.5000 | 0.833 | 15909 | 5552 |
| 196327 | 40 | 55 | 1 | 28 | Friday | 0.10 | 0.10 | 0.200 | 0.0001 | 1.000 | 19061 | 6880 |
| 56897 | 159 | 17 | 1 | 9 | Sunday | 0.02 | 0.07 | 0.340 | 0.0001 | 1.000 | 13224 | 7936 |

```r
advertising_train <- mutate(advertising_train,
                            "ln_cpc" = log(cpc + 0.005),
                            "ln_ctr" = log(ctr + 0.005),
                            "ln_impr" = log(impression + 0.005),
                            "ln_req" = log(requests + 0.005),
                            "ln_y" = log(y + 0.005))

sample_adv <- sample_n(advertising_train, 20)

kable_styling(kable(sample_adv[ , 1 : floor(ncol(sample_adv)/2) ],
                    format.args = list(digits = 3),
                    caption = "Sample of advertising\\_train Data Frame After Logarithmic Transforma
              font_size = 8.5, latex_options = c("striped"),
              full_width = F)

kable_styling(kable(sample_adv[ , c(1, seq(from = floor(ncol(sample_adv)/2)+1,
                                           to = ncol(sample_adv),
                                           by = 1))],
                    format.args = list(digits = 3),
                    caption = "Sample of advertising\\_train Data Frame After Logarithmic Transforma
              font_size = 8.5, latex_options = c("striped"),
              full_width = F)
```

#### 1.2.5.3 Comparison of Transformed Features to Normal Curve

As the logarithmic transformation resulted in infinite values, the data frame was trimmed to only include finite values. The finite data frame was then used to calculate the centre and spread of `ln_cpc`, `ln_ctr`, `ln_impr`, `ln_req`, and `ln_y`.

```r
finite_cpc <- filter(advertising_train,
                     is.finite(ln_cpc))

p_cpc <- ggplot(finite_cpc) +
```

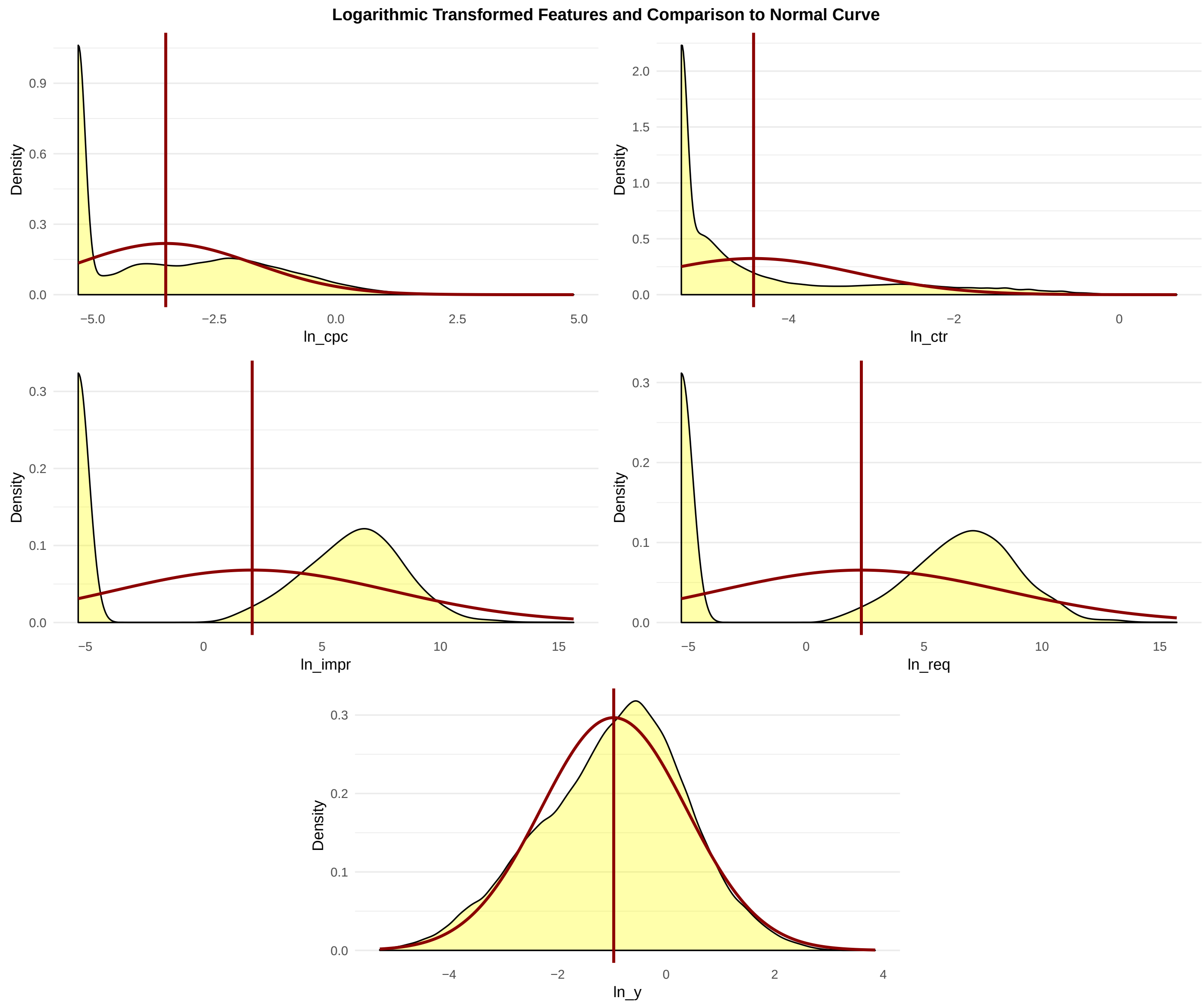
Table 5: Sample of advertising_train Data Frame After Logarithmic Transformations (cont)

| case_id | cpc | ctr | viewability | ratio1 | ratio2 | ratio3 | ratio4 | ratio5 | y | ln_cpc | ln_ctr | ln_impr | ln_req | ln_y |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 146984 | 0.0000 | 0.0000 | 0.0000 | 0.000 | 0.000 | 0.0000 | 0.0000 | 0.000 | 0.0250 | -5.298 | -5.30 | -5.30 | -5.30 | -3.5066 |
| 182164 | 0.0069 | 0.0045 | 0.6730 | 1.000 | 0.991 | 0.0750 | 0.1745 | 0.751 | 0.0267 | -4.431 | -4.66 | 8.61 | 8.63 | -3.4524 |
| 1103 | 0.0000 | 0.0000 | 0.0000 | 0.000 | 0.000 | 0.0000 | 0.0000 | 0.000 | 1.2462 | -5.298 | -5.30 | -5.30 | -5.30 | 0.2241 |
| 79692 | 0.0151 | 0.0312 | 0.9412 | 1.000 | 1.000 | 1.0000 | 0.0000 | 0.000 | 0.5760 | -3.907 | -3.32 | 4.16 | 4.23 | -0.5430 |
| 31797 | 0.0412 | 0.0052 | 0.1671 | 1.000 | 0.819 | 0.4215 | 0.0366 | 0.542 | 0.2507 | -3.075 | -4.59 | 5.95 | 5.95 | -1.3639 |
| 176490 | 0.0146 | 0.1000 | 1.0000 | 1.000 | 0.975 | 1.0000 | 0.0000 | 0.000 | 0.9388 | -3.932 | -2.25 | 3.69 | 3.85 | -0.0578 |
| 30218 | 0.0000 | 0.0000 | 0.0000 | 0.000 | 0.000 | 0.0000 | 0.0000 | 0.000 | 0.0143 | -5.298 | -5.30 | -5.30 | -5.30 | -3.9468 |
| 53528 | 0.0000 | 0.0000 | 0.0000 | 0.000 | 0.000 | 0.0000 | 0.0000 | 0.000 | 0.0590 | -5.298 | -5.30 | -5.30 | -5.30 | -2.7496 |
| 123731 | 0.2843 | 0.0039 | 0.7967 | 0.719 | 0.648 | 0.0625 | 0.4297 | 0.508 | 1.3055 | -1.240 | -4.72 | 5.55 | 5.62 | 0.2704 |
| 52813 | 0.0106 | 0.1837 | 0.9032 | 1.000 | 0.469 | 1.0000 | 0.0000 | 0.000 | 1.9472 | -4.160 | -1.67 | 4.59 | 4.59 | 0.6690 |
| 210661 | 0.0077 | 0.0909 | 0.9000 | 1.000 | 0.273 | 0.0000 | 0.3636 | 0.545 | 0.9111 | -4.366 | -2.34 | 2.40 | 2.49 | -0.0876 |
| 4284 | 0.0175 | 0.3123 | 0.6872 | 1.000 | 0.513 | 1.0037 | 0.0000 | 0.000 | 0.3396 | -3.794 | -1.15 | 5.59 | 8.63 | -1.0654 |
| 65107 | 0.1257 | 0.0184 | 0.9164 | 0.992 | 0.840 | 0.0210 | 0.6063 | 0.373 | 2.2375 | -2.035 | -3.76 | 5.94 | 5.95 | 0.8076 |
| 16126 | 0.0440 | 0.0125 | 0.5943 | 0.399 | 0.877 | 0.0371 | 0.6004 | 0.362 | 0.5491 | -3.016 | -4.05 | 8.54 | 8.73 | -0.5903 |
| 151997 | 0.5060 | 0.0037 | 0.7547 | 0.776 | 0.888 | 1.0000 | 0.0000 | 0.000 | 1.8811 | -0.671 | -4.74 | 5.59 | 5.59 | 0.6345 |
| 169363 | 0.1805 | 0.0035 | 0.8172 | 0.775 | 0.966 | 1.0004 | 0.0000 | 0.000 | 0.2521 | -1.685 | -4.77 | 10.50 | 11.05 | -1.3582 |
| 30335 | 0.0000 | 0.0000 | 0.0000 | 0.000 | 0.000 | 0.0000 | 0.0000 | 0.000 | 0.0267 | -5.298 | -5.30 | -5.30 | -5.30 | -3.4525 |
| 84792 | 0.0496 | 0.0045 | 0.5224 | 0.704 | 0.965 | 0.0657 | 0.0794 | 0.855 | 0.0802 | -2.908 | -4.66 | 8.62 | 9.67 | -2.4631 |
| 196327 | 0.0729 | 0.0015 | 0.5666 | 0.989 | 0.992 | 0.0811 | 0.3026 | 0.616 | 0.0346 | -2.552 | -5.04 | 8.84 | 9.86 | -3.2277 |
| 56897 | 0.0417 | 0.0009 | 0.0521 | 0.429 | 0.997 | 0.0931 | 0.0707 | 0.836 | 0.0217 | -3.064 | -5.13 | 8.98 | 9.49 | -3.6219 |

```r
  geom_density(aes(x = ln_cpc),
               fill = "yellow", alpha = 1/3) +
  stat_function(geom = "path", fun = dnorm,
                n = 200, col = "red4", size = 1,
                args = list(mean(finite_cpc$ln_cpc),
                            sd(finite_cpc$ln_cpc))) +
  geom_vline(xintercept = mean(finite_cpc$ln_cpc),
             col = "red4", size = 1) +
  ylab("Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())

finite_ctr <- filter(advertising_train,
                     is.finite(ln_ctr))

p_ctr <- ggplot(finite_ctr) +
  geom_density(aes(x = ln_ctr),
               fill = "yellow", alpha = 1/3) +
  stat_function(geom = "path", fun = dnorm,
                n = 200, col = "red4", size = 1,
                args = list(mean(finite_ctr$ln_ctr),
                            sd(finite_ctr$ln_ctr))) +
  geom_vline(xintercept = mean(finite_ctr$ln_ctr),
             col = "red4", size = 1) +
  ylab("Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())

finite_impr <- filter(advertising_train,
                      is.finite(ln_impr))
```

```r
p_impr <- ggplot(finite_impr) +
  geom_density(aes(x = ln_impr),
               fill = "yellow", alpha = 1/3) +
  stat_function(geom = "path", fun = dnorm,
                n = 200, col = "red4", size = 1,
                args = list(mean(finite_impr$ln_impr),
                            sd(finite_impr$ln_impr))) +
  geom_vline(xintercept = mean(finite_cpc$ln_impr),
             col = "red4", size = 1) +
  ylab("Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())

finite_req <- filter(advertising_train,
                     is.finite(ln_req))

p_req <- ggplot(finite_req) +
  geom_density(aes(x = ln_req),
               fill = "yellow", alpha = 1/3) +
  stat_function(geom = "path", fun = dnorm,
                n = 200, col = "red4", size = 1,
                args = list(mean(finite_req$ln_req),
                            sd(finite_req$ln_req))) +
  geom_vline(xintercept = mean(finite_cpc$ln_req),
             col = "red4", size = 1) +
  ylab("Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())

finite_y <- filter(advertising_train,
                   is.finite(ln_y))

p_y <- ggplot(finite_y) +
  geom_density(aes(x = ln_y),
               fill = "yellow", alpha = 1/3) +
  stat_function(geom = "path", fun = dnorm,
                n = 200, col = "red4", size = 1,
                args = list(mean(finite_y$ln_y),
                            sd(finite_y$ln_y))) +
  geom_vline(xintercept = mean(finite_cpc$ln_y),
             col = "red4", size = 1) +
  ylab("Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())

ln_vars_title <- textGrob("Logarithmic Transformed Features and Comparison to Normal Curve",
                          gp = gpar(fontface = "bold"))
```

```r
grid.arrange(top = ln_vars_title,
             p_cpc, p_ctr,
             p_impr, p_req,
             p_y,
             layout_matrix = matrix(c(1,1,2,2,
                                      3,3,4,4,
                                      NA,5,5,NA),
                                    ncol = 4,
                                    byrow = T))
```

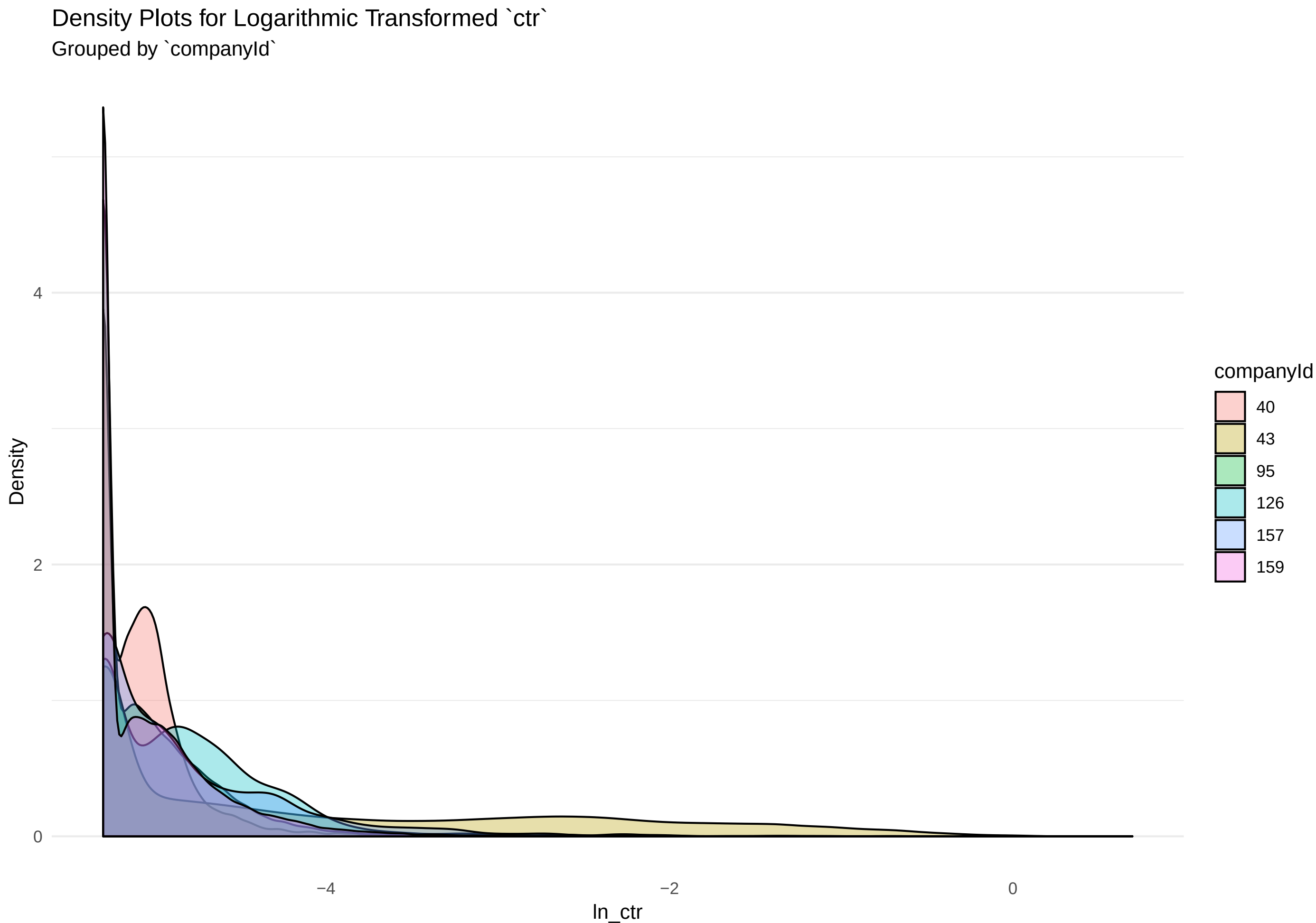



The natural logarithmic transformations of `impression` and `requests` clearly approached a normal distribution. The transformed `y` target feature somewhat resembled a normal distribution, albeit less closely as compared to `impression`. Both `cpc` and `ctr` appeared to be bimodal distributions after logarithmic transformation, with `ln_ctr` inarguably so.

### 1.2.6 Multivariate Plots

After transformation, grouping the `ln_ctr` distribution by level within the `companyId` factor revealed several distinct distributions. The distribution for `companyId` == 43 still appeared bimodal, which possibly indicated a further dimension of the multivariate relationship.

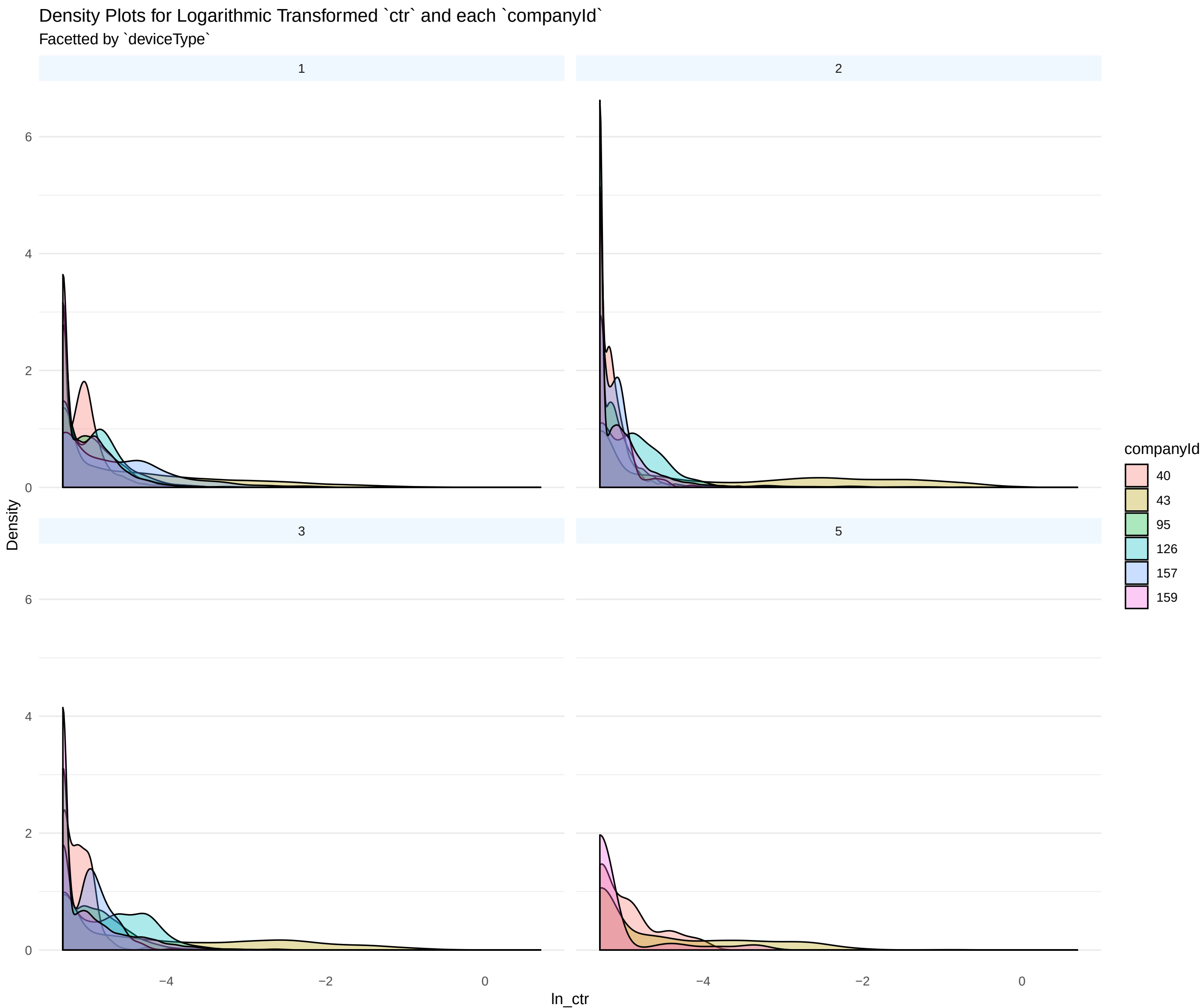
```r
ggplot(advertising_train) +
  geom_density(aes(x = ln_ctr, fill = companyId),
               alpha = 1/3) +
  labs(title = "Density Plots for Logarithmic Transformed \`ctr\`",
       subtitle = "Grouped by \`companyId\`",
       y = "Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())
```




Producing separate density plots for each level within `deviceType` suggested some trivariate relationship between `ln_ctr`, `companyId`, and `deviceType`. The effect of facetting by `deviceType` was particularly apparent when examining `companyId == 43`, yet it still did not yield Gaussian distributions.

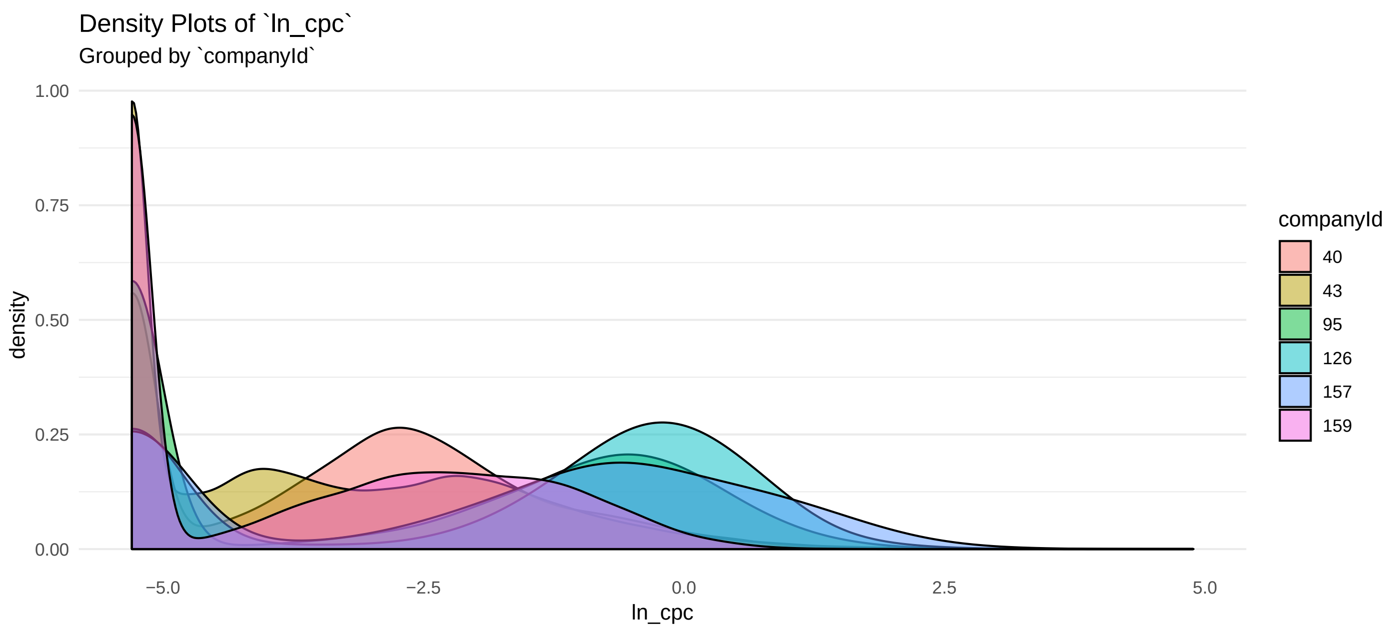
```r
ggplot(advertising_train) +
  geom_density(aes(x = ln_ctr, fill = companyId),
               alpha = 1/3) +
  facet_rep_wrap(~deviceType) +
  labs(title = "Density Plots for Logarithmic Transformed \`ctr\` and each \`companyId\`",
       subtitle = "Facetted by \`deviceType\`",
       y = "Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
```

```r
      strip.background = element_rect(fill = "aliceblue",
                                        colour = NA))
```

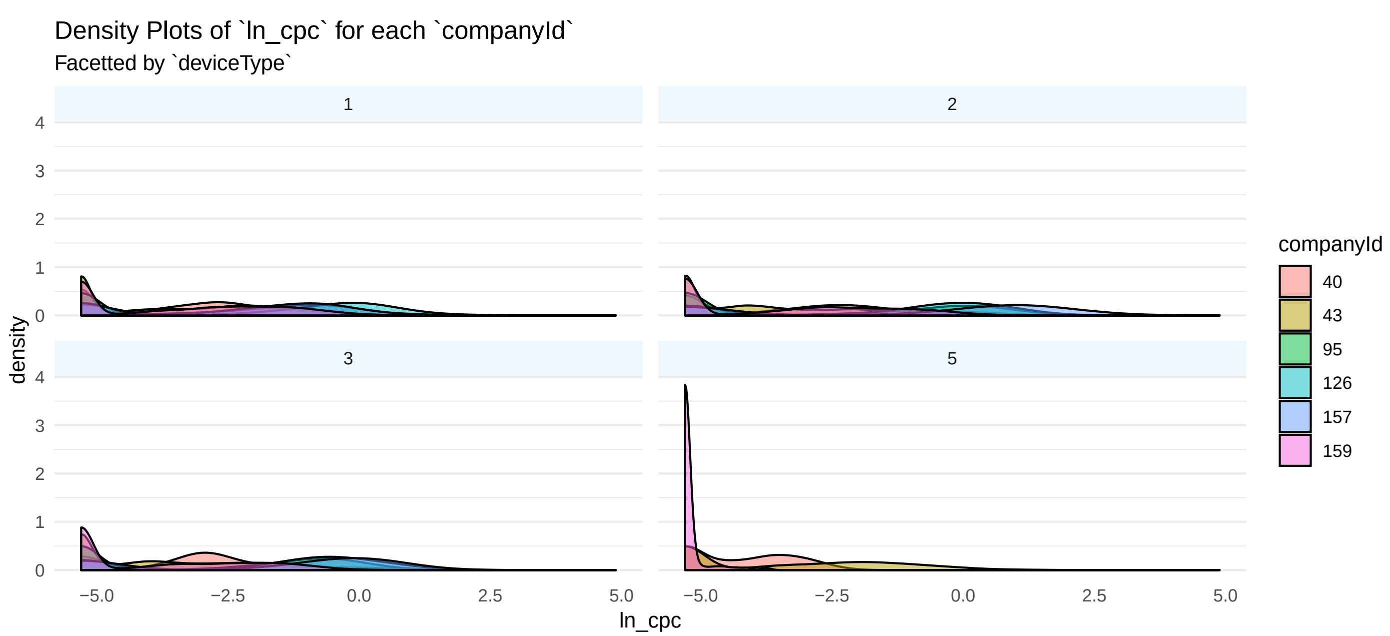



As above for `ln_ctr`, grouping by `companyId` and facetting by `deviceType` revealed a multivariate relationship between aforementioned descriptive features and the transformed `ln_cpc`.

```r
ggplot(advertising_train) +
  geom_density(aes(x = ln_cpc, fill = companyId),
               alpha = 1/2) +
  labs(title = "Density Plots of \`ln_cpc\`",
       subtitle = "Grouped by \`companyId\`",
       ylab = "Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())
```




```r
ggplot(advertising_train) +
  geom_density(aes(x = ln_cpc, fill = companyId),
               alpha = 1/2) +
  facet_rep_wrap(~deviceType) +
  labs(title = "Density Plots of \`ln_cpc\` for each \`companyId\`",
       subtitle = "Facetted by \`deviceType\`",
       ylab = "Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
        strip.background = element_rect(fill = "aliceblue",
                                        colour = NA))
```

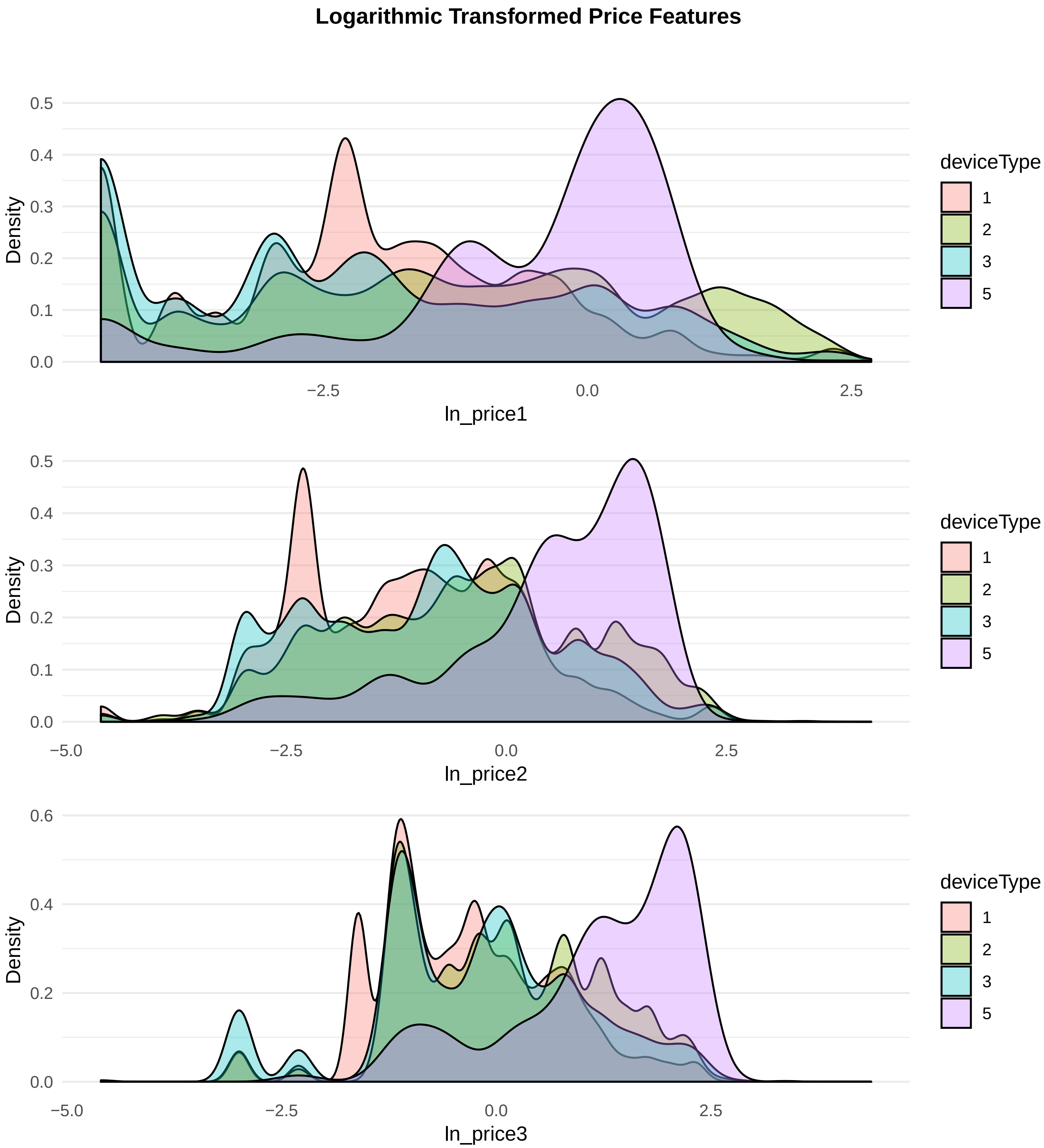



Each of the pricing features, (`price1`, `price2`, `price3`) were not suitably transformed by either logarithmic, square root, or cube root. Logarithmic transformations appeared to spread the data the most, but these transformations considerably diverged from a symmetrical normal distribution. Further grouping by `deviceType` did not reveal Gaussian distributions.

```r
price_trans <- mutate(advertising_train,
                      "ln_price1" = log(price1),
                      "ln_price2" = log(price2),
                      "ln_price3" = log(price3))

p_price1_trans <- ggplot(price_trans) +
  geom_density(aes(x = ln_price1, fill = deviceType),
               alpha = 1/3) +
  labs(y = "Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())

p_price2_trans <- ggplot(price_trans) +
  geom_density(aes(x = ln_price2, fill = deviceType),
               alpha = 1/3) +
  labs(y = "Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())

p_price3_trans <- ggplot(price_trans) +
  geom_density(aes(x = ln_price3, fill = deviceType),
               alpha = 1/3) +
  labs(y = "Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())


price_vars_title <- textGrob("Logarithmic Transformed Price Features",
                             gp = gpar(fontface = "bold"))

grid.arrange(price_vars_title,
             p_price1_trans, p_price2_trans,
             p_price3_trans,
             layout_matrix = matrix(c(1,
                                      2,
                                      2,
                                      2,
                                      3,
                                      3,
                                      3,
                                      4,
                                      4,
                                      4),
                                    ncol = 1,
                                    byrow = T))
```

```
## Warning: Removed 92892 rows containing non-finite values (stat_density).

## Warning: Removed 92804 rows containing non-finite values (stat_density).
```

```
## Warning: Removed 92804 rows containing non-finite values (stat_density).
```

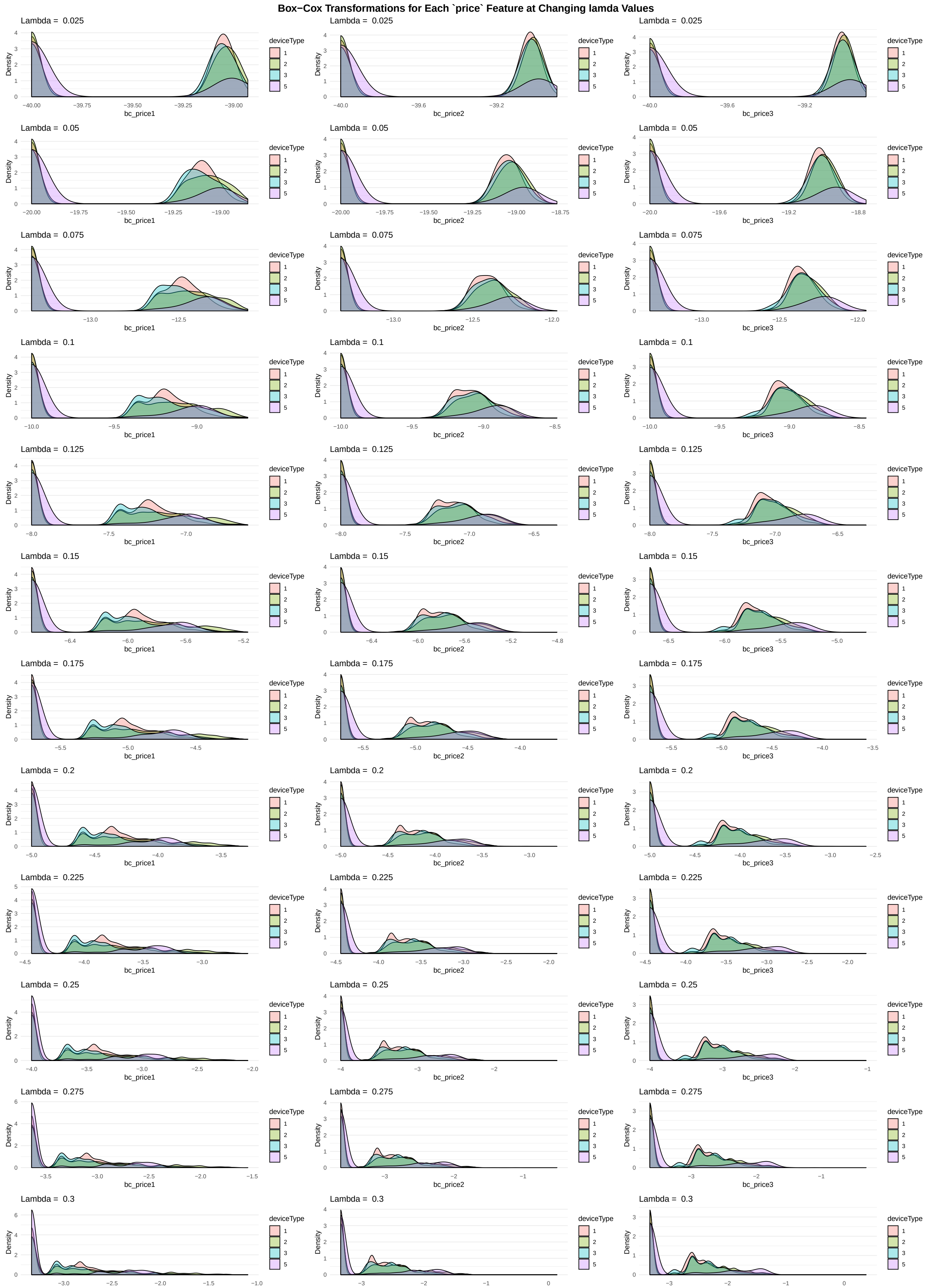



Box-Cox transformations with a range of lamda values also did not convert the price features into distributions that resembled a normal curve.

```
boxcox <- function(x, lambda = 1) {
```

```
  (x^(lambda) - 1 /
     (lambda))

}

box_grobs_2 <- list()
box_grobs_higher <- list()

for (i in 1:length(seq(0.025, 0.3, 0.025))) {

  j <- seq(0.025, 0.3, 0.025)[i]

  boxcox_price <- mutate(advertising_train,
                         "bc_price1" = boxcox(x = price1,
                                              lambda = j),
                         "bc_price2" = boxcox(x = price2,
                                              lambda = j),
                         "bc_price3" = boxcox(x = price3,
                                              lambda = j))

  bc_colnames <- colnames(boxcox_price)[str_detect(colnames(boxcox_price), "bc_price")]

  for (k in bc_colnames) {

    m <- which(bc_colnames %in% k)

    box_grobs_2[[m]] <- ggplot(select(boxcox_price,
                                      k, deviceType)) +
      geom_density(aes(x = .data[[k]], fill = deviceType),
                   alpha = 1/3) +
      labs(title = paste("Lambda = ", j)) +
      ylab("Density") + xlab(k) +
      theme_minimal() +
      theme(panel.grid.major.x = element_blank(),
            panel.grid.minor.x = element_blank())

  }

  box_grobs_higher[[i]] <- box_grobs_2

}

density_by_lambda <- list()

for (i in 1:12) {

  density_by_lambda[[i]] <-  do.call(what = grid.arrange,
                                     args = list(grobs = box_grobs_higher[[i]],
                                                 nrow = 1))

}
```

```
do.call(what = grid.arrange,
        args = list(grobs = density_by_lambda,
                    top = textGrob("Box-Cox Transformations for Each \`price\` Feature at Changing l
                                   gp = gpar(fontsize=16,
                                             fontface = "bold")),
                    ncol = 1))
```


Box−Cox Transformations for Each `price` Feature at Changing lamda Values
Lambda = 0.025
Lambda = 0.05
Lambda = 0.075
Lambda = 0.1
Lambda = 0.125
Lambda = 0.15
Lambda = 0.175
Lambda = 0.2
Lambda = 0.225
Lambda = 0.25
Lambda = 0.275
Lambda = 0.3
Density
bc_price1
bc_price2
bc_price3
deviceType
1
2
3
5

The remaining numeric features (`ad_area`, `ad_ratio`, `day`, `ratio1`, `ratio2`, `ratio3`, `ratio4`, `ratio5`, and `viewability`) were not able to be transformed to distributions that approached normal curves via root or logarithmic methods. Despite the accompanying documentation for the prescribed dataset, the `ad_area` and `day` may not strictly be classed as numeric/double variables. Considering the low range, `ad_area` could be intepreted as an identifier, and so categorical. The feature `day`, values 1 - 30, is better interpreted as an ordinal or time value. However, time series forecasting is outside the scope of this project, and so the `day` feature will be largely ignored from the model and only used for partitioning.

#### 1.2.6.1 Data Normalisation

Considering each of the features span differing ranges, both in their raw and transformed applications, it was deemed necessary to normalise each. Normalising the data allowed for more

As outlined in **Fundamentals of Machine Learning**, the below formula was used for normalising the data:

$$a_i^{'} = \left(\frac{a_i - min(a)}{max(a) - min(a)}\right) \times (high - low) + low$$

Where $a$ is the feature, whether descriptive or target, $high$ is the highest value in the normalised data range, and $low$ is the lowest value in the normalised data range. A range of 0 - 1 was chosen, so these values were used for $low$ and $high$ respectively.

```
normalise <- function(x) {

  x[is.infinite(x)] <- NA

  (((x - min(x, na.rm = T)) /
      (max(x, na.rm = T) - min(x, na.rm = T))) * (1 - 0) + 0)

}

num_feats <- select(advertising_train,
                    case_id,
                    which(sapply(advertising_train, class)=="numeric"))

for ( i in colnames(num_feats)) {

  newfeat <- paste0("norm_", i)

  advertising_train[[newfeat]] <- normalise(num_feats[[i]])

  advertising_train[[newfeat]][is.na(advertising_train[[newfeat]])] <- advertising_train[[i]]

}


sample_adv <- sample_n(advertising_train, 20)

kable_styling(kable(sample_adv[, 1:floor(ncol(sample_adv)/3)],
                    caption = "Sample of advertising\\_train Data Frame with Normalised Numeric Feat
                    format.args = list(digits = 2, scientific = F,
                                       big.mark = ",")),
              font_size = 8, latex_options = c("striped"),
              full_width = T)
```

```
kable_styling(kable(sample_adv[, c(1,
                                   seq(from = floor(ncol(sample_adv)/3)*1+1,
                                       to = floor(ncol(sample_adv)/3)*2,
                                       by = 1))],
                    caption = "Sample of advertising\\_train Data Frame with Normalised Numeric Feat
                    format.args = list(digits = 2, scientific = F,
                                       big.mark = ",")),
```

Table 6: Sample of advertising_train Data Frame with Normalised Numeric Features (1/3)

| case_id | companyId | countryId | deviceType | day | dow | price1 | price2 | price3 | ad_area | ad_ratio | requests | impression | cpc | ctr | viewability |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 51,925 | 95 | 234 | 3 | 8 | Saturday | 0.01 | 0.74 | 1.49 | 7.5000 | 0.83 | 1,923 | 948 | 1.1486 | 0.0011 | 0.21 |
| 43,376 | 43 | 91 | 2 | 7 | Friday | 0.00 | 0.00 | 0.00 | 0.0001 | 1.00 | 24 | 20 | 0.0015 | 0.3000 | 0.87 |
| 30,903 | 159 | 200 | 1 | 5 | Wednesday | 0.01 | 0.27 | 0.55 | 0.0001 | 1.00 | 243 | 240 | 0.0360 | 0.0042 | 0.29 |
| 34,363 | 159 | 110 | 1 | 6 | Thursday | 0.07 | 0.11 | 0.34 | 7.5000 | 0.83 | 0 | 0 | 0.0000 | 0.0000 | 0.00 |
| 90,812 | 43 | 82 | 2 | 14 | Friday | 5.66 | 5.66 | 5.66 | 0.0001 | 1.00 | 200 | 70 | 0.0194 | 0.0286 | 0.80 |
| 186,289 | 43 | 12 | 2 | 26 | Wednesday | 1.30 | 1.72 | 3.45 | 24.2500 | 0.26 | 0 | 0 | 0.0000 | 0.0000 | 0.00 |
| 110,701 | 43 | 36 | 2 | 16 | Sunday | 0.36 | 1.00 | 2.01 | 0.0001 | 1.00 | 114 | 63 | 0.0342 | 0.0317 | 0.98 |
| 16,871 | 159 | 12 | 2 | 3 | Monday | 0.03 | 0.15 | 0.31 | 0.0001 | 1.00 | 0 | 0 | 0.0000 | 0.0000 | 0.00 |
| 115,978 | 43 | 12 | 2 | 17 | Monday | 3.42 | 3.42 | 3.42 | 0.0001 | 1.00 | 37 | 32 | 0.0183 | 0.2812 | 0.90 |
| 131,004 | 95 | 77 | 2 | 19 | Wednesday | 0.01 | 0.10 | 0.29 | 6.5520 | 0.12 | 0 | 0 | 0.0000 | 0.0000 | 0.00 |
| 55,896 | 43 | 231 | 2 | 9 | Sunday | 4.54 | 4.54 | 4.54 | 0.0001 | 1.00 | 327 | 85 | 0.0027 | 0.0941 | 0.84 |
| 195,395 | 43 | 13 | 1 | 28 | Friday | 0.00 | 0.00 | 0.00 | 6.5520 | 0.12 | 0 | 0 | 0.0000 | 0.0000 | 0.00 |
| 118,799 | 43 | 191 | 1 | 18 | Tuesday | 0.80 | 0.80 | 0.80 | 0.0001 | 1.00 | 152 | 42 | 0.0391 | 0.0238 | 1.47 |
| 57,434 | 43 | 125 | 2 | 9 | Sunday | 0.03 | 0.07 | 0.33 | 6.5520 | 0.12 | 0 | 0 | 0.0000 | 0.0000 | 0.00 |
| 89,162 | 43 | 113 | 2 | 14 | Friday | 0.00 | 0.00 | 0.00 | 0.0001 | 1.00 | 81 | 33 | 0.0022 | 0.1212 | 0.80 |
| 64,224 | 43 | 77 | 3 | 10 | Monday | 4.54 | 4.54 | 4.54 | 0.0001 | 1.00 | 2,790 | 798 | 0.0997 | 0.0351 | 0.67 |
| 52,335 | 43 | 205 | 1 | 9 | Sunday | 0.00 | 0.00 | 0.00 | 0.0001 | 1.00 | 0 | 0 | 0.0000 | 0.0000 | 0.00 |
| 130,951 | 43 | 95 | 2 | 19 | Wednesday | 0.80 | 0.80 | 0.80 | 0.0001 | 1.00 | 480 | 108 | 0.1444 | 0.0185 | 0.83 |
| 158,670 | 43 | 179 | 2 | 23 | Sunday | 5.96 | 5.96 | 5.96 | 0.0001 | 1.00 | 63,256 | 1,881 | 0.0288 | 0.4588 | 0.84 |
| 175,709 | 43 | 179 | 2 | 25 | Tuesday | 0.12 | 0.31 | 0.61 | 6.5520 | 0.12 | 0 | 0 | 0.0000 | 0.0000 | 0.00 |

Table 7: Sample of advertising_train Data Frame with Normalised Numeric Features (2/3)

| case_id | ratio1 | ratio2 | ratio3 | ratio4 | ratio5 | y | ln_cpc | ln_ctr | ln_impr | ln_req | ln_y | norm_case_id | norm_day | norm_price1 | norm_price2 | norm_price3 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 51,925 | 0.86 | 0.78 | 0.0084 | 0.55 | 0.44 | 0.452 | 0.14 | -5.10 | 6.9 | 7.6 | -0.78 | 0.242 | 0.241 | 0.00068 | 0.0117 | 0.0189 |
| 43,376 | 1.00 | 0.95 | 1.0000 | 0.00 | 0.00 | 0.052 | -5.04 | -1.19 | 3.0 | 3.2 | -2.86 | 0.203 | 0.207 | 0.00000 | 0.0000 | 0.0000 |
| 30,903 | 0.71 | 0.99 | 0.0583 | 0.10 | 0.84 | 0.167 | -3.19 | -4.69 | 5.5 | 5.5 | -1.76 | 0.144 | 0.138 | 0.00068 | 0.0043 | 0.0069 |
| 34,363 | 0.00 | 0.00 | 0.0000 | 0.00 | 0.00 | 0.101 | -5.30 | -5.30 | -5.3 | -5.3 | -2.24 | 0.160 | 0.172 | 0.00477 | 0.0017 | 0.0044 |
| 90,812 | 1.00 | 0.81 | 1.0000 | 0.00 | 0.00 | 0.033 | -3.71 | -3.39 | 4.2 | 5.3 | -3.28 | 0.424 | 0.448 | 0.38530 | 0.0897 | 0.0717 |
| 186,289 | 0.00 | 0.00 | 0.0000 | 0.00 | 0.00 | 0.238 | -5.30 | -5.30 | -5.3 | -5.3 | -1.41 | 0.870 | 0.862 | 0.08850 | 0.0272 | 0.0437 |
| 110,701 | 0.65 | 1.00 | 1.0000 | 0.00 | 0.00 | 0.440 | -3.24 | -3.30 | 4.1 | 4.7 | -0.81 | 0.517 | 0.517 | 0.02451 | 0.0158 | 0.0254 |
| 16,871 | 0.00 | 0.00 | 0.0000 | 0.00 | 0.00 | 0.338 | -5.30 | -5.30 | -5.3 | -5.3 | -1.07 | 0.079 | 0.069 | 0.00204 | 0.0024 | 0.0039 |
| 115,978 | 0.97 | 0.88 | 1.0000 | 0.00 | 0.00 | 5.310 | -3.76 | -1.25 | 3.5 | 3.6 | 1.67 | 0.542 | 0.552 | 0.23281 | 0.0542 | 0.0433 |
| 131,004 | 0.00 | 0.00 | 0.0000 | 0.00 | 0.00 | 0.175 | -5.30 | -5.30 | -5.3 | -5.3 | -1.72 | 0.612 | 0.621 | 0.00068 | 0.0016 | 0.0037 |
| 55,896 | 1.00 | 0.42 | 1.0000 | 0.00 | 0.00 | 0.372 | -4.87 | -2.31 | 4.4 | 5.8 | -0.98 | 0.261 | 0.276 | 0.30905 | 0.0719 | 0.0575 |
| 195,395 | 0.00 | 0.00 | 0.0000 | 0.00 | 0.00 | 1.136 | -5.30 | -5.30 | -5.3 | -5.3 | 0.13 | 0.913 | 0.931 | 0.00000 | 0.0000 | 0.0000 |
| 118,799 | 1.00 | 0.26 | 0.0476 | 0.00 | 0.95 | 0.398 | -3.12 | -3.55 | 3.7 | 5.0 | -0.91 | 0.555 | 0.586 | 0.05446 | 0.0127 | 0.0101 |
| 57,434 | 0.00 | 0.00 | 0.0000 | 0.00 | 0.00 | 0.033 | -5.30 | -5.30 | -5.3 | -5.3 | -3.28 | 0.268 | 0.276 | 0.00204 | 0.0011 | 0.0042 |
| 89,162 | 1.00 | 0.48 | 1.0000 | 0.00 | 0.00 | 0.038 | -4.93 | -2.07 | 3.5 | 4.4 | -3.15 | 0.416 | 0.448 | 0.00000 | 0.0000 | 0.0000 |
| 64,224 | 0.98 | 0.64 | 0.0113 | 0.69 | 0.30 | 1.563 | -2.26 | -3.22 | 6.7 | 7.9 | 0.45 | 0.300 | 0.310 | 0.30905 | 0.0719 | 0.0575 |
| 52,335 | 0.00 | 0.00 | 0.0000 | 0.00 | 0.00 | 0.259 | -5.30 | -5.30 | -5.3 | -5.3 | -1.33 | 0.244 | 0.276 | 0.00000 | 0.0000 | 0.0000 |
| 130,951 | 0.99 | 0.27 | 1.0000 | 0.00 | 0.00 | 0.672 | -1.90 | -3.75 | 4.7 | 6.2 | -0.39 | 0.612 | 0.621 | 0.05446 | 0.0127 | 0.0101 |
| 158,670 | 0.77 | 0.81 | 1.0005 | 0.00 | 0.00 | 0.444 | -3.39 | -0.77 | 7.5 | 11.1 | -0.80 | 0.741 | 0.759 | 0.40572 | 0.0944 | 0.0755 |
| 175,709 | 0.00 | 0.00 | 0.0000 | 0.00 | 0.00 | 0.031 | -5.30 | -5.30 | -5.3 | -5.3 | -3.33 | 0.821 | 0.828 | 0.00817 | 0.0049 | 0.0078 |

```r
              font_size = 8, latex_options = c("striped"),
              full_width = T)
```

```r
kable_styling(kable(sample_adv[, c(1,
                                   seq(from = floor(ncol(sample_adv)/3)*2+1,
                                       to = floor(ncol(sample_adv)/3)*3,
                                       by = 1))],
                    caption = "Sample of advertising\\_train Data Frame with Normalised Numeric Feat
                    format.args = list(digits = 2, scientific = F,
                                       big.mark = ",")),
              font_size = 8, latex_options = c("striped"),
              full_width = T)
```