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
199272	43	57	3	28	Friday	2.28	2.28	2.2764	0.0001	1.00000
4751	95	234	3	1	Saturday	0.10	0.25	0.2500	0.0001	1.00000
178000	43	191	2	25	Tuesday	2.27	2.27	2.2691	0.0001	1.00000
95382	43	167	2	14	Friday	0.00	0.00	0.0000	0.0001	1.00000
81377	159	77	2	13	Thursday	0.01	0.08	0.3165	0.0001	1.00000
158058	43	75	2	23	Sunday	0.00	0.00	0.0000	0.0001	1.00000
36210	159	234	2	6	Thursday	0.00	0.00	0.0000	0.0001	1.00000
138078	43	189	2	20	Thursday	1.14	1.14	1.1392	0.0001	1.00000
21329	159	43	1	4	Tuesday	0.02	0.07	0.2849	7.5000	0.83333
45511	43	109	3	8	Saturday	0.00	0.00	0.0000	0.0001	1.00000
29240	43	202	2	5	Wednesday	0.00	0.00	0.0000	9.0000	1.00000
95616	43	13	2	14	Friday	2.73	4.21	8.4108	8.7300	0.09278
75896	43	202	2	12	Wednesday	0.00	0.00	0.0000	31.1850	0.31818
212052	159	12	1	30	Sunday	0.03	0.07	0.3283	7.5000	0.83333
168151	43	2	1	24	Monday	0.06	0.26	0.5332	7.5000	0.83333
157895	40	68	1	23	Sunday	0.10	0.10	0.2000	0.0001	1.00000
31454	95	77	1	5	Wednesday	0.04	0.12	0.3700	7.5000	0.83333
74245	159	17	3	12	Wednesday	0.01	0.09	0.3382	7.5000	0.83333
5203	43	110	3	1	Saturday	2.28	2.28	2.2813	0.0001	1.00000
206257	43	231	2	29	Saturday	0.00	0.00	0.0000	0.0001	1.00000

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
199272	0	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.6884615
4751	0	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1226580
178000	187	158	0.0062	0.2975	0.8919	0.9557	0.6899	1.0000	0.0000	0.0000	1.6663265
95382	135	127	0.0164	0.2047	1.0000	1.0000	0.9921	1.0000	0.0000	0.0000	2.9118421
81377	0	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2653398
158058	28	28	0.0916	0.0714	0.9565	1.0000	0.2500	1.0000	0.0000	0.0000	4.8666667
36210	0	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.2024691
138078	75	73	0.0117	0.1918	0.7969	0.9863	0.7260	1.0000	0.0000	0.0000	2.2848485
21329	0	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1925332
45511	0	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	2.0538462
29240	5631	5614	0.1293	0.0023	0.6876	1.0000	0.8954	1.0009	0.0000	0.0000	0.3116040
95616	372	123	0.5051	0.0081	0.9180	0.6829	0.9512	1.0000	0.0000	0.0000	2.0178637
75896	3000	2995	0.0986	0.0023	0.0637	1.0000	0.6831	1.0013	0.0000	0.0000	0.2474261
212052	8396	7804	0.1388	0.0013	0.2053	0.8540	0.8569	0.0627	0.2199	0.7175	0.1815472
168151	5049	1690	0.2179	0.0018	0.1504	0.8331	0.4166	0.0657	0.5355	0.3988	0.1188653
157895	4037	2853	0.0474	0.0032	0.3842	0.9758	0.8994	0.0673	0.3277	0.6050	0.0937990
31454	0	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0896552
74245	0	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0664723
5203	0	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.6666667
206257	668	658	0.0391	0.0061	0.4527	1.0000	0.8040	1.0000	0.0000	0.0000	0.2596522

```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)
```

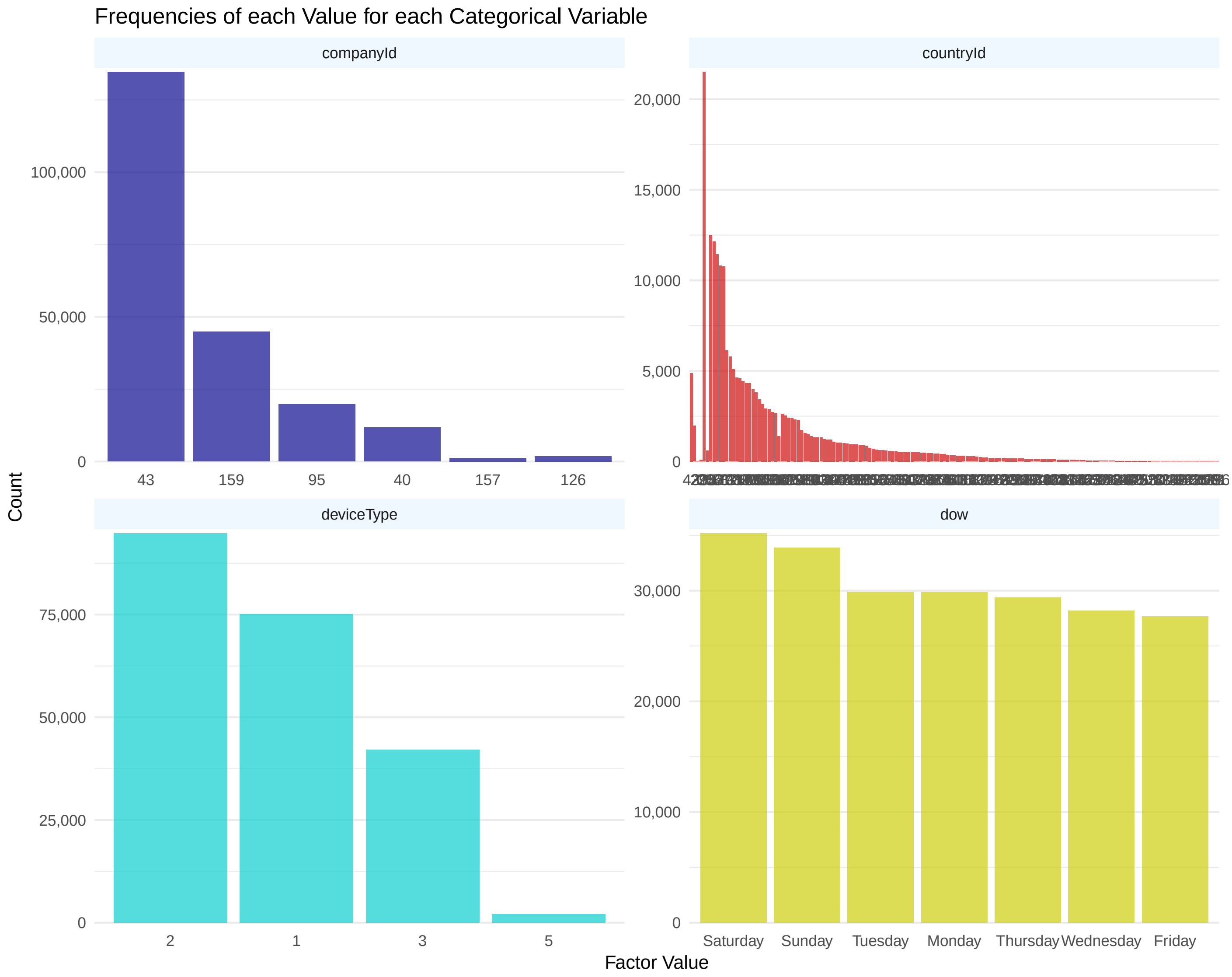
1.2.4.2 Categorical and Non-Numeric Features

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")
```

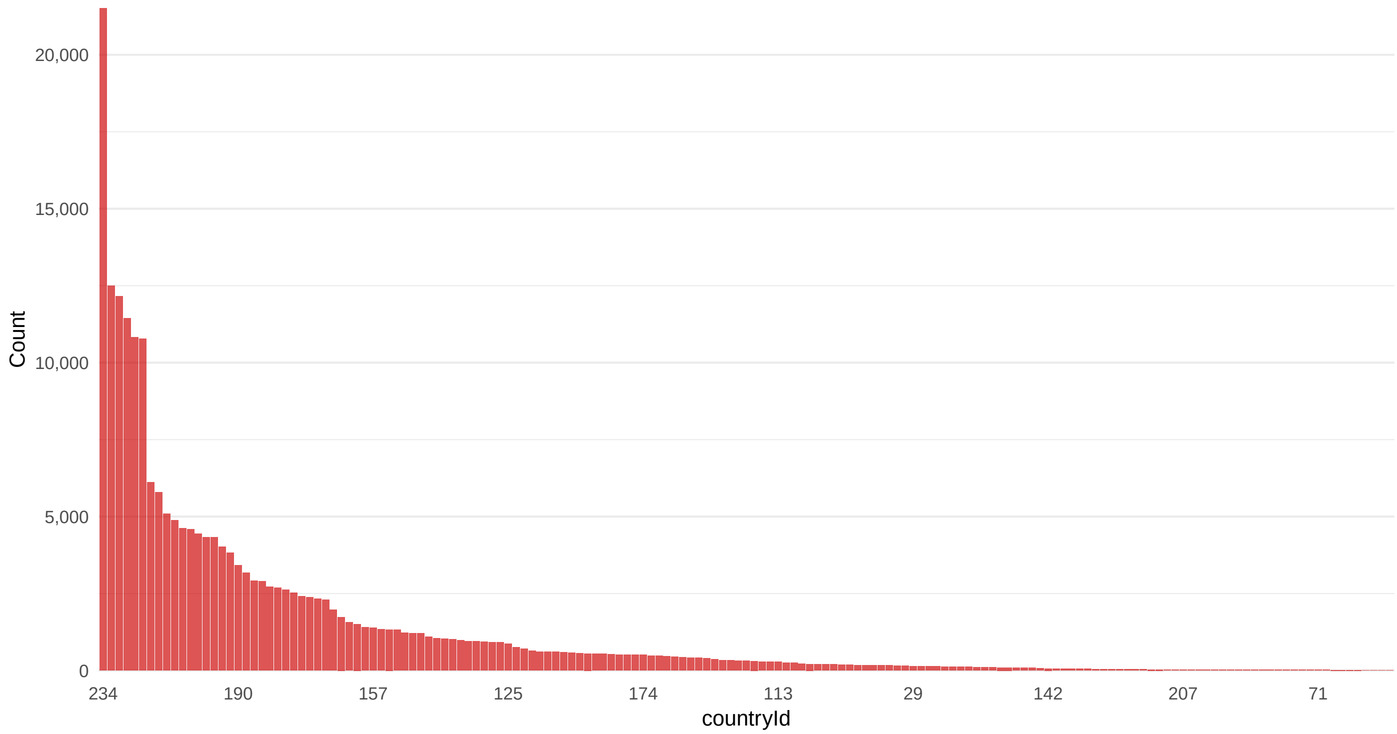
```
## Warning: attributes are not identical across measure variables;
## they will be dropped
```

```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)

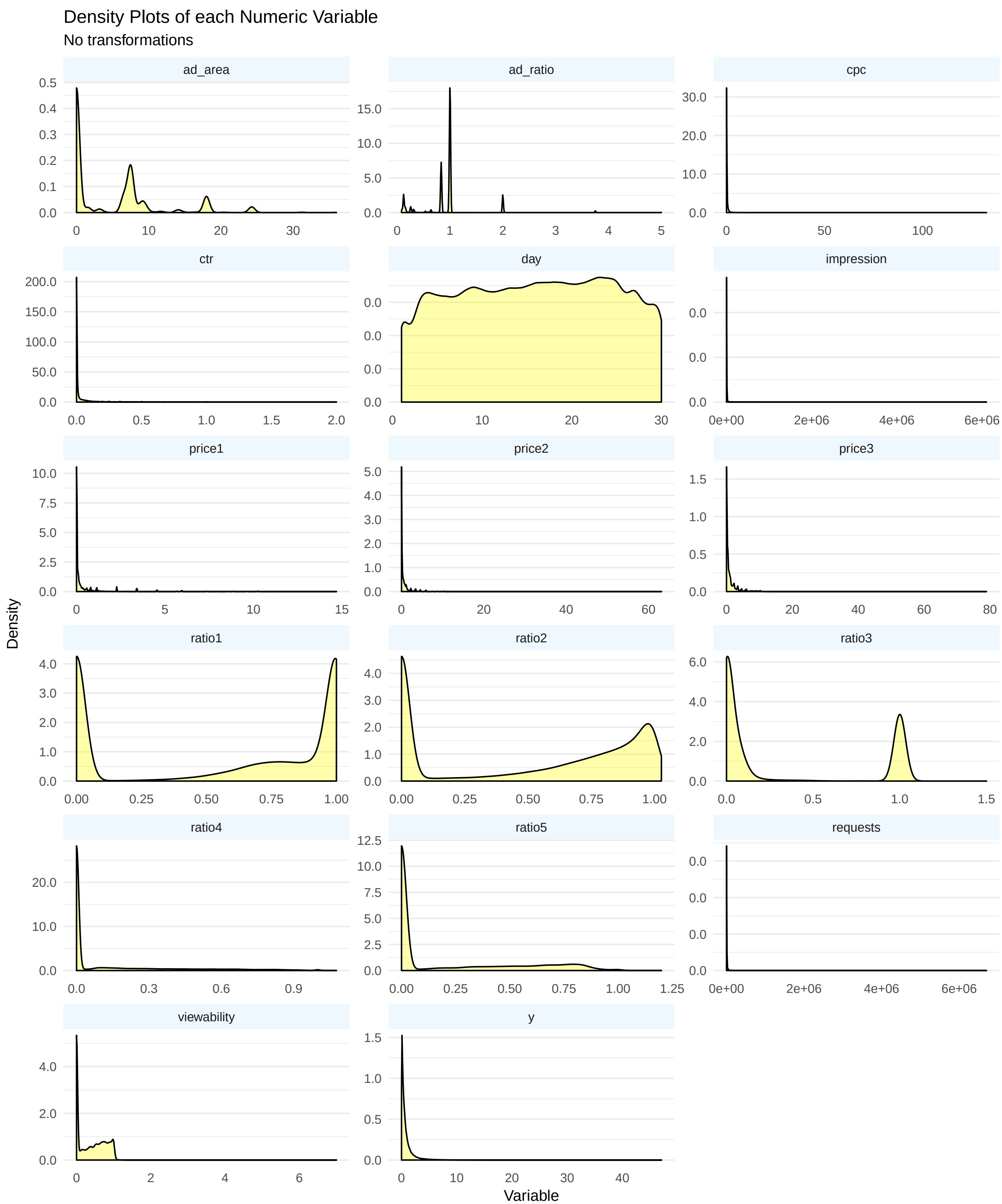


labels along x−axis are ID numbers and not numeric/double/ordinal/etc

1.2.5 Univariate Plots

1.2.5.1 Numeric Variables

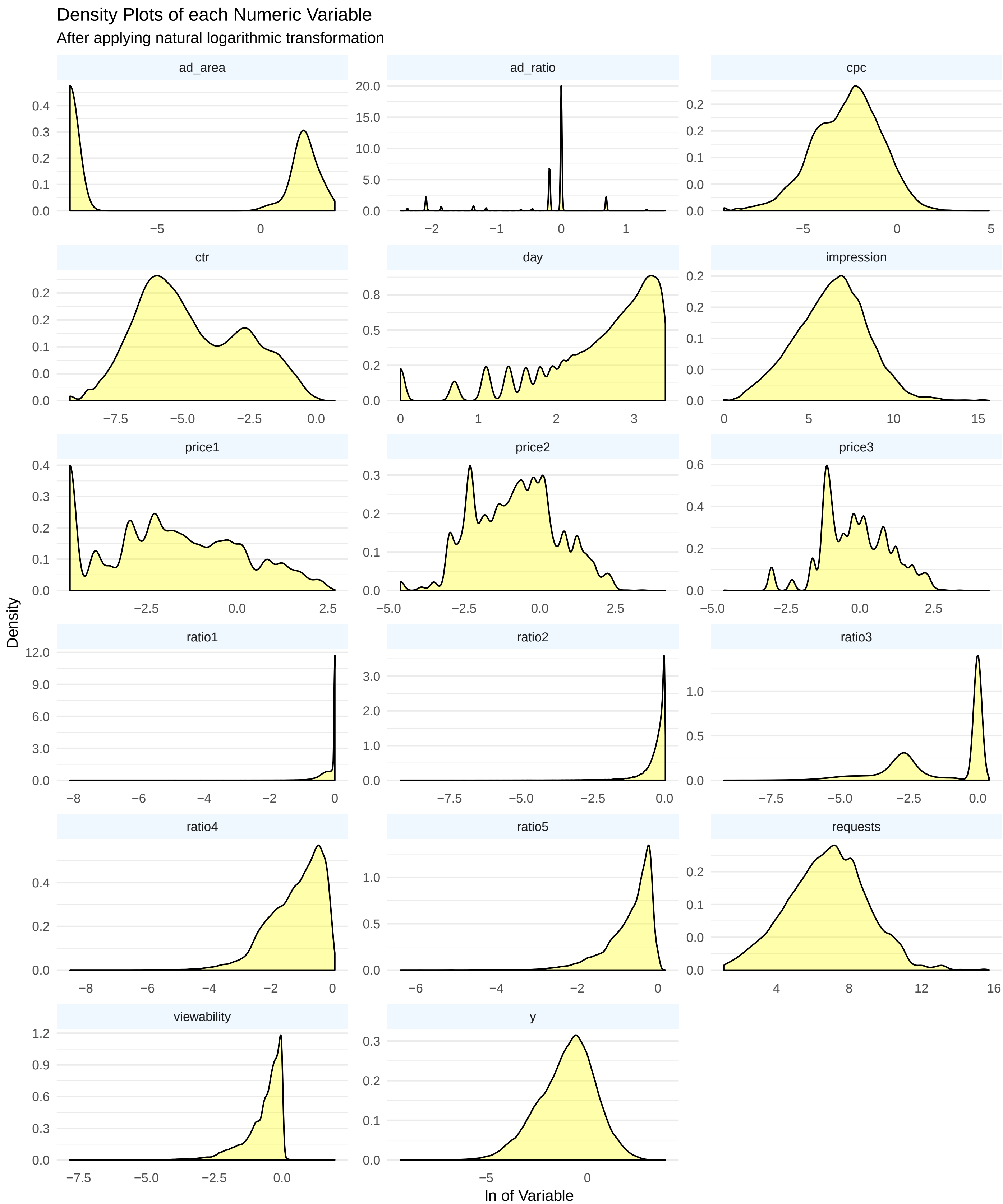
```r
ggplot(advertising_train_long_num) +
  geom_density(aes(x = Value),
               fill = "yellow",
               alpha = 1/3) +
  facet_rep_wrap(~Variable,
                 repeat.tick.labels = T,
                 scales = "free",
                 ncol = 3) +
  scale_y_continuous(labels = comma_format(accuracy = 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
206199	95	13	3	29	Saturday	0.01	0.35	0.700	7.5000	0.8333	0	0
104660	43	191	1	16	Sunday	0.00	0.00	0.000	6.5520	0.1236	152	149
203536	43	100	2	29	Saturday	0.00	0.00	0.000	0.0001	1.0000	135	129
76012	43	197	2	12	Wednesday	0.25	0.68	1.353	8.7300	0.0928	1777	1300
46329	159	234	2	8	Saturday	0.00	0.00	0.000	0.0001	1.0000	0	0
137859	43	56	3	20	Thursday	0.15	0.26	0.513	20.3700	0.2165	733	660
195741	43	172	1	28	Friday	2.28	2.28	2.276	0.0001	1.0000	276	95
14848	95	234	2	3	Monday	0.05	0.05	0.050	6.5520	0.1236	0	0
2887	43	135	1	1	Saturday	0.35	0.71	1.403	0.0001	1.0000	0	0
171623	43	234	2	24	Monday	1.53	2.81	5.616	6.5520	0.1236	3400	3099
112560	43	134	2	17	Monday	0.80	0.80	0.798	0.0001	1.0000	296	51
181313	43	202	1	26	Wednesday	0.00	0.00	0.000	7.5000	0.8333	1368	1365
185138	43	191	3	26	Wednesday	0.72	1.08	2.158	7.9200	0.1818	1631	532
124134	43	56	2	18	Tuesday	0.00	0.00	0.000	18.0000	2.0000	0	0
214049	43	38	2	30	Sunday	3.27	5.05	10.098	8.7300	0.0928	2080	772
66491	95	234	3	10	Monday	0.05	0.05	0.050	18.0000	2.0000	0	0
166485	43	57	1	24	Monday	0.07	0.39	0.771	3.2000	0.3125	0	0
180252	43	171	3	25	Tuesday	0.00	0.00	0.000	0.0001	1.0000	16	15
72433	43	193	1	11	Tuesday	0.00	0.00	0.000	6.5520	0.1236	0	0
63625	43	231	1	10	Monday	0.01	0.12	0.374	7.5000	0.8333	128	98

```
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`.

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

p_cpc <- ggplot(finite_cpc) +
```

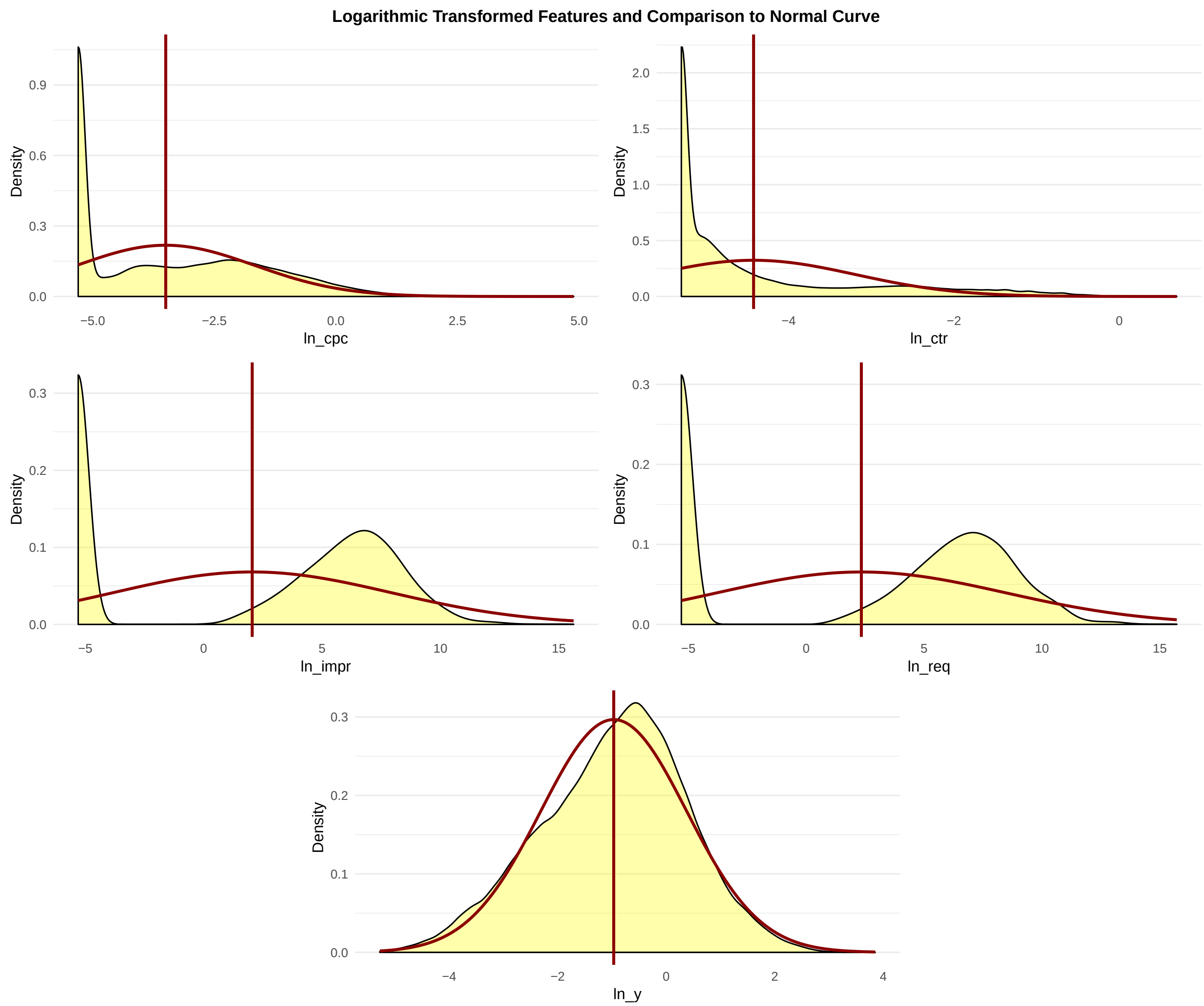
Table 5: Sample of advertising_train Data Frame After Logarithmic Transformations (cont)

case_id	cpc	ctr	viewability	ratio1	ratio2	ratio3	ratio4	ratio5	y	ln_cpc	ln_ctr	ln_impr	ln_req	ln_y
206199	0.0000	0.0000	0.000	0.000	0.000	0.0000	0.000	0.000	0.474	-5.29832	-5.30	-5.30	-5.30	-0.7351
104660	0.0718	0.0336	0.254	1.000	0.960	0.0201	0.329	0.651	1.370	-2.56655	-3.25	5.00	5.02	0.3185
203536	0.0244	0.0155	0.686	1.000	0.512	1.0000	0.000	0.000	0.405	-3.52676	-3.89	4.86	4.91	-0.8917
76012	0.3757	0.0023	0.870	0.908	0.968	1.0000	0.000	0.000	0.592	-0.96574	-4.92	7.17	7.48	-0.5153
46329	0.0000	0.0000	0.000	0.000	0.000	0.0000	0.000	0.000	0.443	-5.29832	-5.30	-5.30	-5.30	-0.8036
137859	0.0754	0.0076	0.695	0.923	0.962	0.0227	0.301	0.676	0.558	-2.52074	-4.37	6.49	6.60	-0.5739
195741	0.0204	0.0632	0.821	0.979	0.979	0.1684	0.210	0.621	0.870	-3.67301	-2.69	4.55	5.62	-0.1336
14848	0.0000	0.0000	0.000	0.000	0.000	0.0000	0.000	0.000	0.338	-5.29832	-5.30	-5.30	-5.30	-1.0713
2887	0.0000	0.0000	0.000	0.000	0.000	0.0000	0.000	0.000	0.422	-5.29832	-5.30	-5.30	-5.30	-0.8502
171623	0.6508	0.0061	0.472	0.892	0.831	1.0000	0.000	0.000	3.217	-0.42190	-4.50	8.04	8.13	1.1699
112560	0.0249	0.0392	1.000	0.941	0.941	1.0000	0.000	0.000	0.139	-3.50990	-3.12	3.93	5.69	-1.9354
181313	0.3972	0.0015	0.440	1.000	0.663	0.0740	0.180	0.752	0.681	-0.91081	-5.04	7.22	7.22	-0.3762
185138	0.0237	0.0451	0.861	0.645	1.000	0.0132	0.314	0.673	0.319	-3.55086	-2.99	6.28	7.40	-1.1261
124134	0.0000	0.0000	0.000	0.000	0.000	0.0000	0.000	0.000	0.301	-5.29832	-5.30	-5.30	-5.30	-1.1835
214049	1.0019	0.0039	0.906	0.698	0.985	1.0000	0.000	0.000	1.310	0.00688	-4.72	6.65	7.64	0.2742
66491	0.0000	0.0000	0.000	0.000	0.000	0.0000	0.000	0.000	0.570	-5.29832	-5.30	-5.30	-5.30	-0.5536
166485	0.0000	0.0000	0.000	0.000	0.000	0.0000	0.000	0.000	1.526	-5.29832	-5.30	-5.30	-5.30	0.4261
180252	0.0072	0.3333	1.000	1.000	1.000	0.0000	0.200	0.800	2.477	-4.40632	-1.08	2.71	2.77	0.9092
72433	0.0000	0.0000	0.000	0.000	0.000	0.0000	0.000	0.000	1.067	-5.29832	-5.30	-5.30	-5.30	0.0692
63625	0.1155	0.0102	0.884	0.898	0.990	0.0000	0.122	0.857	0.835	-2.11611	-4.19	4.59	4.85	-0.1741

```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))
```

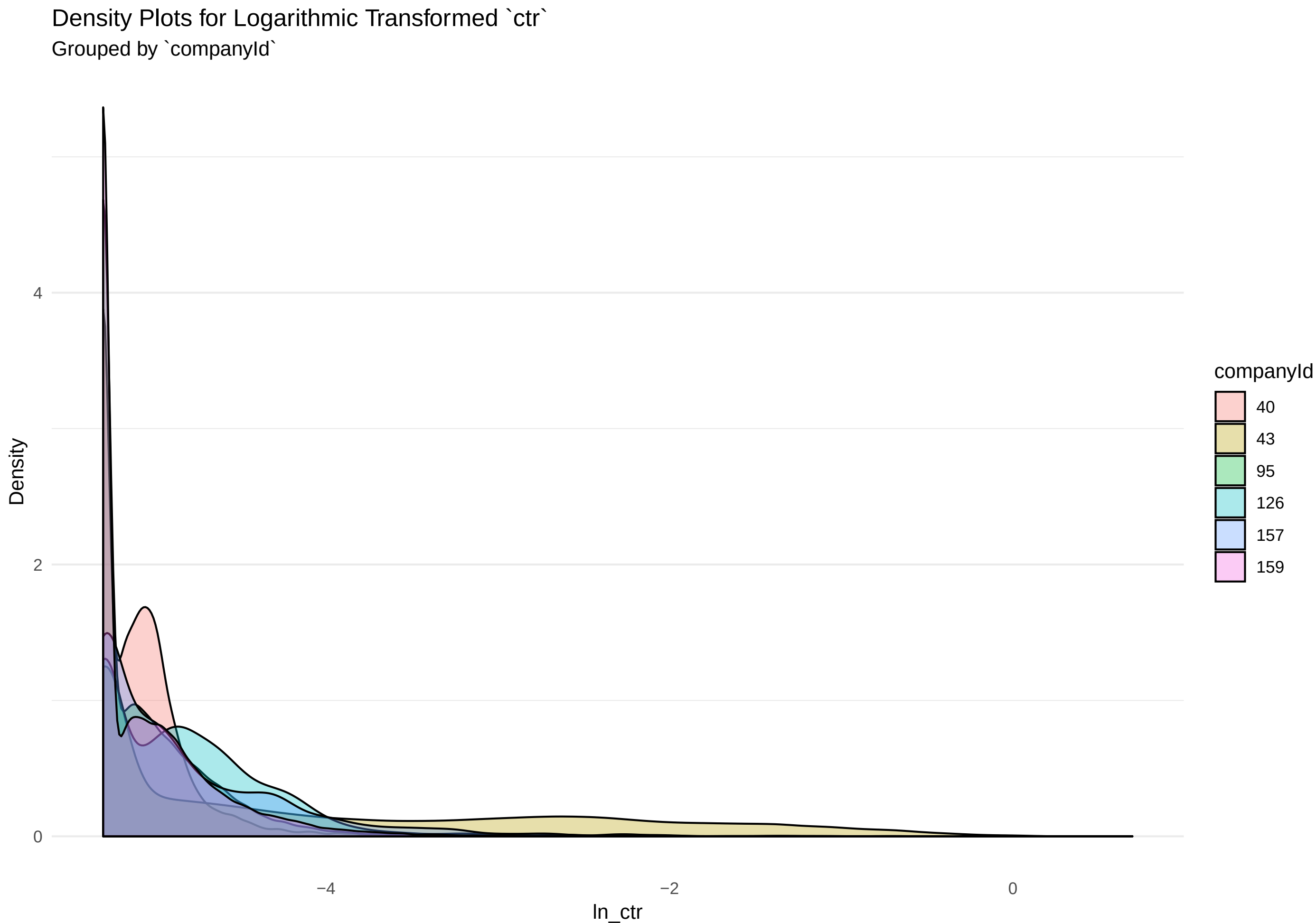


The natural logarithmic transformations of `impression` and `requests` clearly approached a normal distribution. The transformed `y` target feature somewhat resembled a normal distribution, albeit less closely as compared to `impression`. Both `cpc` and `ctr` appeared to be bimodal distributions after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

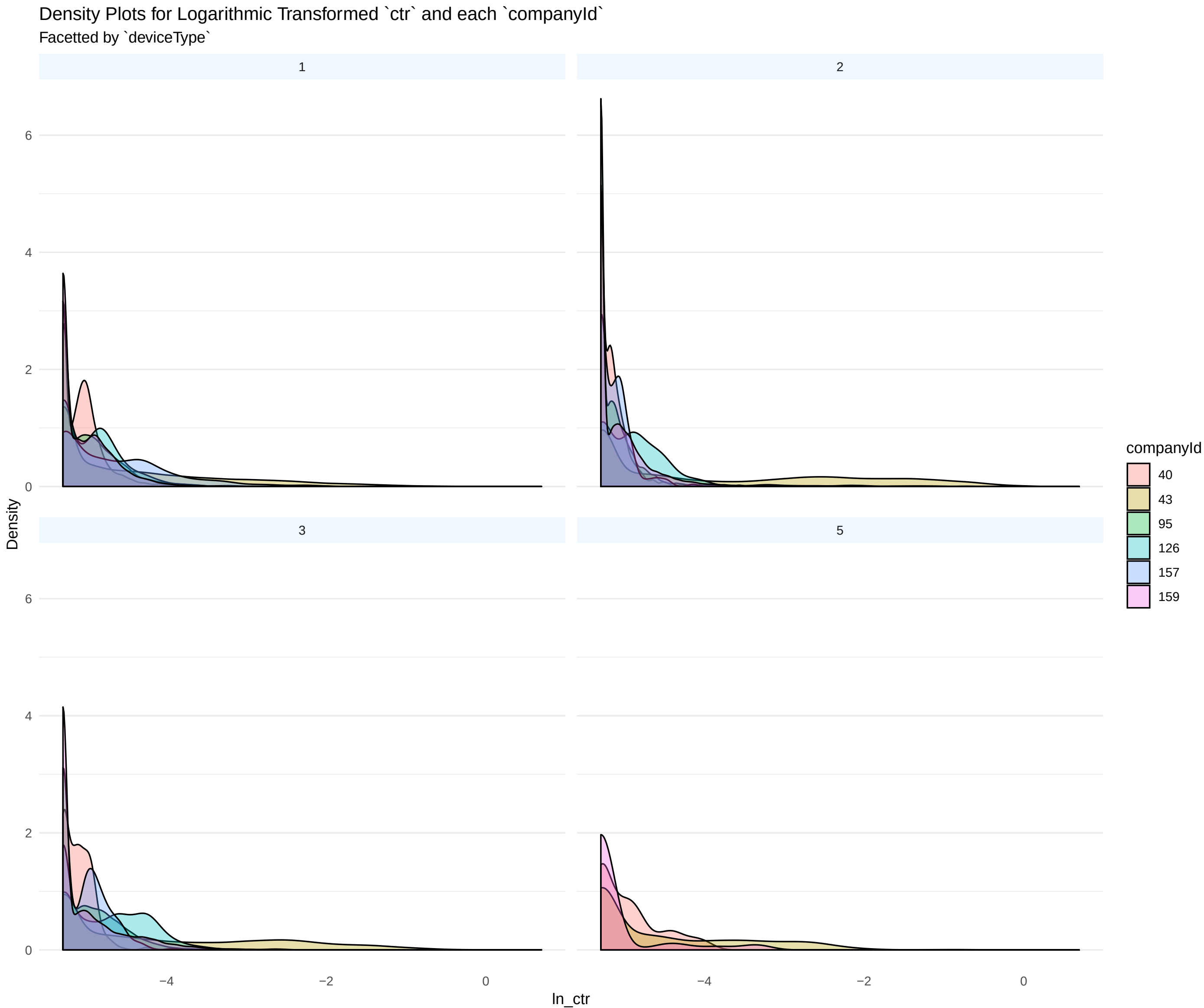
```r
ggplot(advertising_train) +
  geom_density(aes(x = ln_ctr, fill = companyId),
               alpha = 1/3) +
  labs(title = "Density Plots for Logarithmic Transformed \`ctr\`",
       subtitle = "Grouped by \`companyId\`",
       y = "Density") +
  theme_minimal() +
  theme(panel.grid.major.x = element_blank(),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

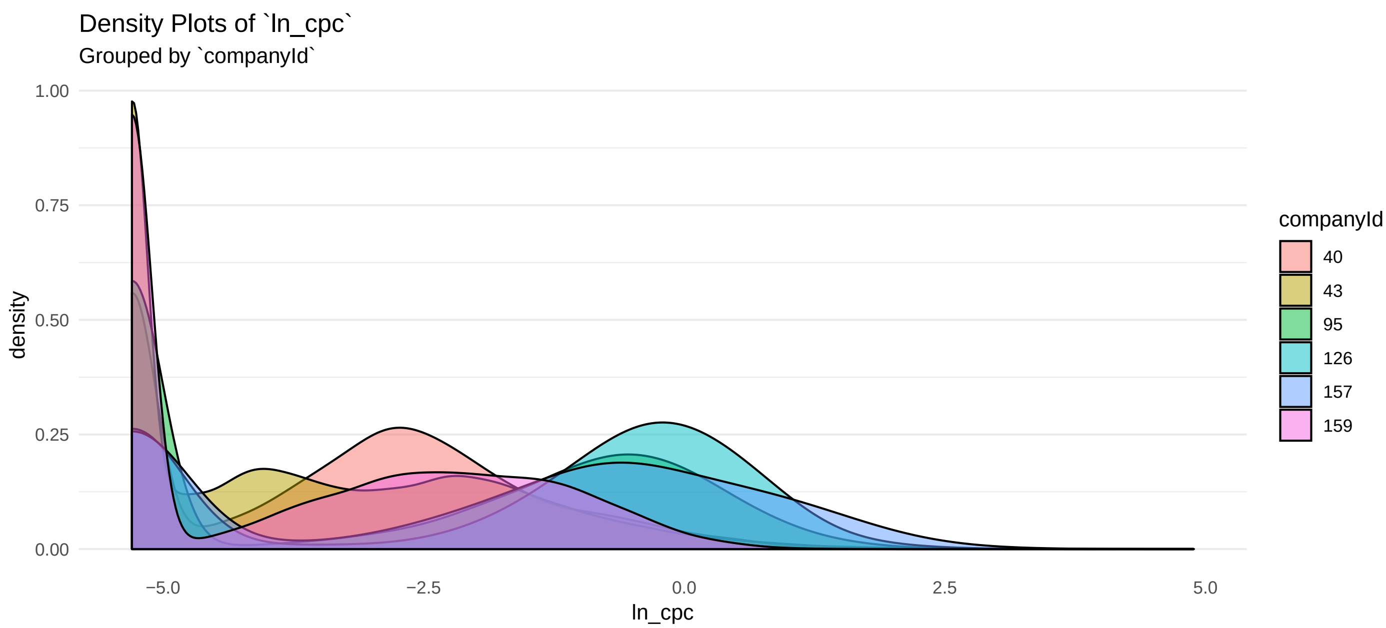
```r
ggplot(advertising_train) +
  geom_density(aes(x = ln_ctr, fill = companyId),
               alpha = 1/3) +
  facet_rep_wrap(~deviceType) +
  labs(title = "Density Plots for Logarithmic Transformed \`ctr\` and each \`companyId\`",
       subtitle = "Facetted by \`deviceType\`",
       y = "Density")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

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

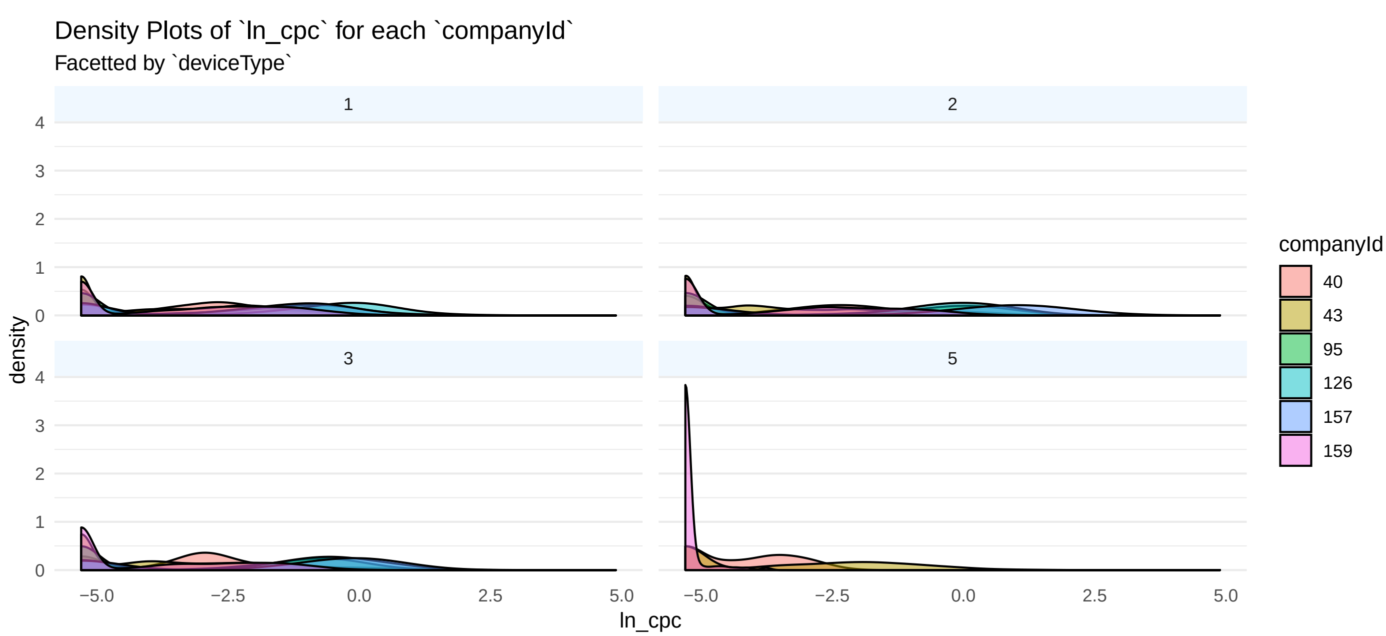


As above for `ln_ctr`, grouping by `companyId` and facetting by `deviceType` revealed a multivariate relationship between aforementioned descriptive features and the transformed `ln_cpc`.

```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))
```

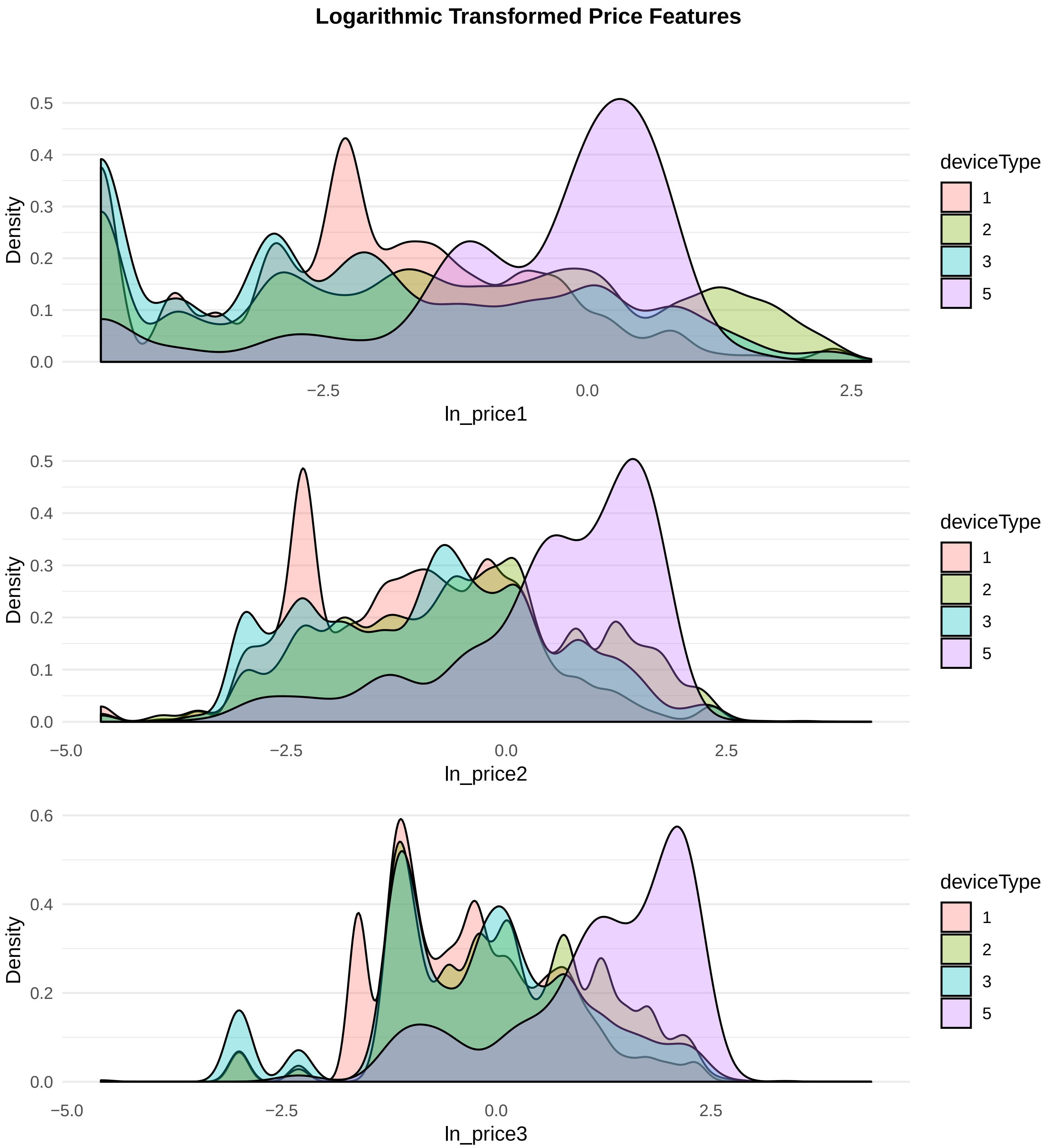


Each of the pricing features, (`price1`, `price2`, `price3`) were not suitably transformed by either logarithmic, square root, or cube root. Logarithmic transformations appeared to spread the data the most, but these transformations considerably diverged from a symmetrical normal distribution. Further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



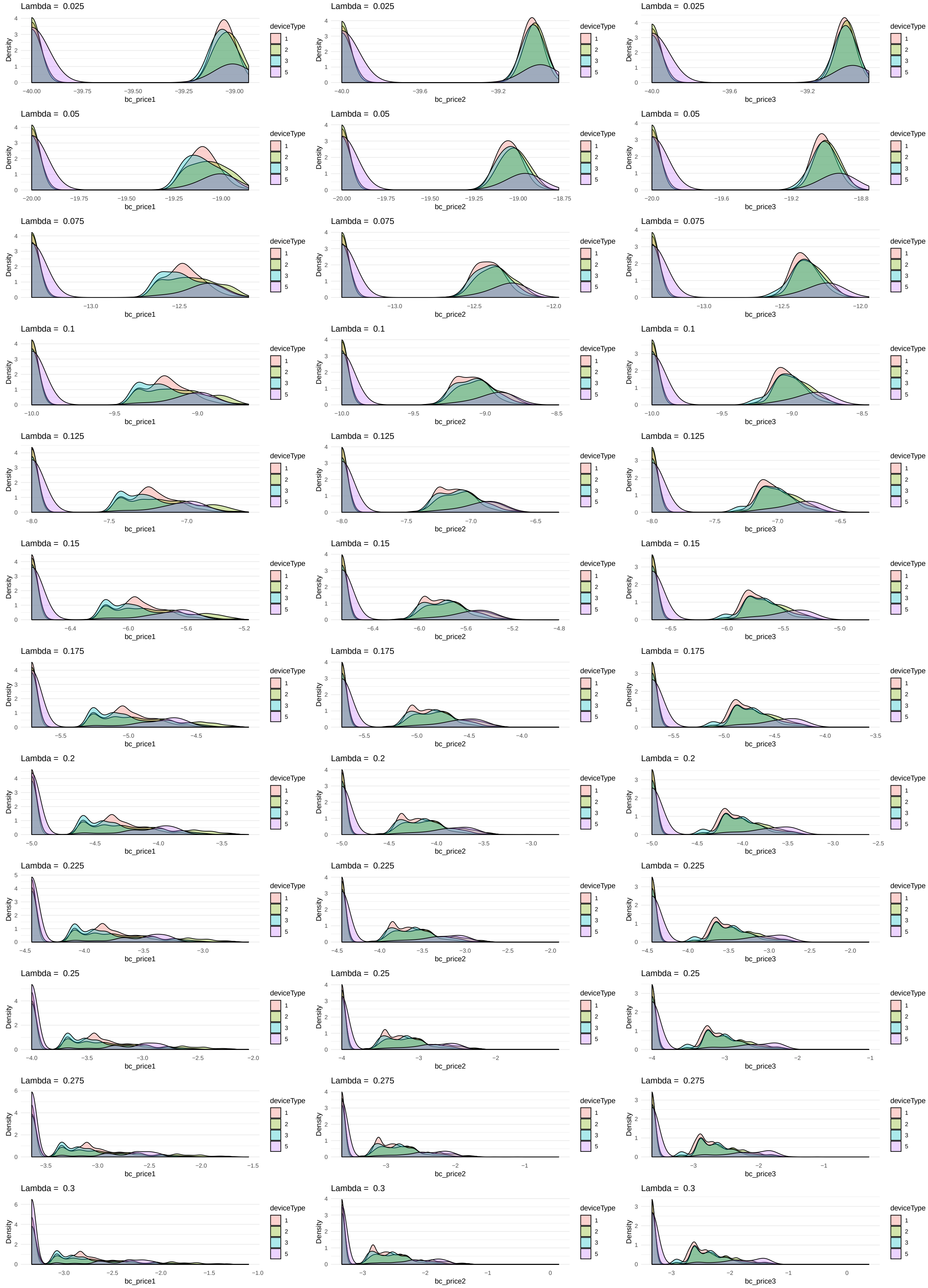
Box-Cox transformations with a range of lamda values also did not convert the price features into distributions that resembled a normal curve.

```
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



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
78,286	43	226	2	12	Wednesday	0.14	0.23	0.45	7.5000	0.83	3,676	1,110	0.1674	0.0018	0.63
78,213	43	20	2	12	Wednesday	0.08	0.14	0.42	6.5520	0.12	0	0	0.0000	0.0000	0.00
173,079	43	234	2	25	Tuesday	4.54	4.54	4.54	0.0001	1.00	126	19	0.0741	0.1579	0.73
156	43	234	1	1	Saturday	0.00	0.00	0.00	9.0000	1.00	0	0	0.0000	0.0000	0.00
8,506	43	68	1	2	Sunday	0.00	0.00	0.00	0.0001	1.00	0	0	0.0000	0.0000	0.00
7,048	43	56	5	2	Sunday	0.00	0.00	0.00	18.0000	2.00	0	0	0.0000	0.0000	0.00
144,872	159	75	3	21	Friday	0.01	0.15	0.29	24.2500	0.26	0	0	0.0000	0.0000	0.00
56,618	43	31	1	9	Sunday	0.57	1.25	2.51	7.5000	0.83	20,390	2,414	0.2713	0.0037	0.52
149,353	43	191	2	22	Saturday	0.20	0.40	0.79	7.9200	0.18	4,456	2,826	0.0956	0.0092	0.91
95,071	43	56	3	14	Friday	2.26	2.26	2.26	0.0001	1.00	6,054	3,965	0.0268	0.0398	0.69
104,320	43	100	2	16	Sunday	0.00	0.00	0.00	24.2500	0.26	0	0	0.0000	0.0000	0.00
67,393	43	227	2	11	Tuesday	0.25	0.69	1.37	0.0001	1.00	253	203	0.0108	0.0739	1.00
86,378	43	57	5	13	Thursday	0.00	0.00	0.00	7.5000	0.83	0	0	0.0000	0.0000	0.00
79,000	95	77	1	12	Wednesday	0.03	0.10	0.30	7.5000	0.83	0	0	0.0000	0.0000	0.00
156,217	159	57	1	22	Saturday	0.14	0.30	0.58	0.0001	1.00	17,138	11,237	0.3059	0.0016	0.20
10,833	95	13	2	3	Monday	0.05	0.05	0.05	18.0000	2.00	0	0	0.0000	0.0000	0.00
181,503	43	202	3	26	Wednesday	0.07	0.12	0.38	18.0000	2.00	448	445	0.2331	0.0022	0.48
19,102	43	49	1	4	Tuesday	0.09	0.19	0.38	0.0001	1.00	31	18	0.0028	0.1111	0.86
11,742	40	100	1	3	Monday	0.00	0.00	0.00	0.0001	1.00	3,106	2,843	0.0241	0.0007	0.32
176,301	43	57	2	25	Tuesday	3.40	3.40	3.40	0.0001	1.00	25	23	0.1804	0.0435	0.83

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
78,286	0.77	0.46	1.0000	0.000	0.00	0.091	-1.8	-5.0	7.0	8.2	-2.347	0.36560	0.379	0.00953	0.00364	0.00571
78,213	0.00	0.00	0.0000	0.000	0.00	0.057	-5.3	-5.3	-5.3	-5.3	-2.779	0.36526	0.379	0.00545	0.00222	0.00529
173,079	1.00	0.37	1.0000	0.000	0.00	0.376	-2.5	-1.8	2.9	4.8	-0.965	0.80830	0.828	0.30905	0.07193	0.05752
156	0.00	0.00	0.0000	0.000	0.00	0.481	-5.3	-5.3	-5.3	-5.3	-0.722	0.00072	0.000	0.00000	0.00000	0.00000
8,506	0.00	0.00	0.0000	0.000	0.00	0.970	-5.3	-5.3	-5.3	-5.3	-0.025	0.03972	0.034	0.00000	0.00000	0.00000
7,048	0.00	0.00	0.0000	0.000	0.00	2.235	-5.3	-5.3	-5.3	-5.3	0.807	0.03291	0.034	0.00000	0.00000	0.00000
144,872	0.00	0.00	0.0000	0.000	0.00	0.686	-5.3	-5.3	-5.3	-5.3	-0.369	0.67657	0.690	0.00068	0.00238	0.00361
56,618	0.57	0.64	0.1292	0.123	0.75	0.107	-1.3	-4.7	7.8	9.9	-2.192	0.26441	0.276	0.03880	0.01980	0.03176
149,353	0.72	0.99	1.0000	0.000	0.00	0.538	-2.3	-4.3	7.9	8.4	-0.610	0.69749	0.724	0.01361	0.00634	0.01007
95,071	0.90	0.42	0.0053	0.099	0.90	0.479	-3.4	-3.1	8.3	8.7	-0.726	0.44399	0.448	0.15385	0.03580	0.02869
104,320	0.00	0.00	0.0000	0.000	0.00	0.218	-5.3	-5.3	-5.3	-5.3	-1.498	0.48718	0.517	0.00000	0.00000	0.00000
67,393	0.57	0.98	1.0000	0.000	0.00	0.639	-4.1	-2.5	5.3	5.5	-0.440	0.31473	0.345	0.01702	0.01093	0.01739
86,378	0.00	0.00	0.0000	0.000	0.00	0.036	-5.3	-5.3	-5.3	-5.3	-3.199	0.40339	0.414	0.00000	0.00000	0.00000
79,000	0.00	0.00	0.0000	0.000	0.00	0.121	-5.3	-5.3	-5.3	-5.3	-2.073	0.36894	0.379	0.00204	0.00158	0.00380
156,217	0.69	0.77	0.0667	0.179	0.75	0.313	-1.2	-5.0	9.3	9.7	-1.147	0.72955	0.724	0.00953	0.00475	0.00733
10,833	0.00	0.00	0.0000	0.000	0.00	6.433	-5.3	-5.3	-5.3	-5.3	1.862	0.05059	0.069	0.00340	0.00079	0.00063
181,503	0.96	0.79	0.0427	0.315	0.65	0.556	-1.4	-4.9	6.1	6.1	-0.579	0.84764	0.862	0.00477	0.00190	0.00487
19,102	0.78	0.83	0.0000	0.389	0.56	0.154	-4.9	-2.2	2.9	3.4	-1.840	0.08920	0.103	0.00613	0.00301	0.00477
11,742	1.00	0.99	0.0675	0.476	0.46	0.016	-3.5	-5.2	8.0	8.0	-3.843	0.05483	0.069	0.00000	0.00000	0.00000
176,301	1.00	0.52	1.0000	0.000	0.00	6.067	-1.7	-3.0	3.1	3.2	1.804	0.82334	0.828	0.23145	0.05387	0.04314

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

```
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)
```

Table 8: Sample of advertising_train Data Frame with Normalised Numeric Features (3/3)

case_id	norm_ad_area	norm_ad_ratio	norm_requests	norm_impression	norm_cpc	norm_ctr	norm_viewability	norm_ratio1	norm_ratio2	norm_ratio3	norm_ratio4	norm_ratio5	norm_y	norm_ln_cpc	norm_ln_ctr	norm_ln_impr
78,286	0.21	0.1525	0.0005485	0.0001820	0.001263	0.00090	0.090	0.77	0.45	0.6667	0.000	0.00	0.00193	0.348	0.051	0.59
78,213	0.18	0.0082	0.0000000	0.0000000	0.000000	0.00000	0.000	0.00	0.00	0.0000	0.000	0.00	0.00121	0.000	0.000	0.00
173,079	0.00	0.1864	0.0000188	0.0000031	0.000559	0.07895	0.105	1.00	0.36	0.6667	0.000	0.00	0.00799	0.271	0.581	0.39
156	0.25	0.1864	0.0000000	0.0000000	0.000000	0.00000	0.000	0.00	0.00	0.0000	0.000	0.00	0.01021	0.000	0.000	0.00
8,506	0.00	0.1864	0.0000000	0.0000000	0.000000	0.00000	0.000	0.00	0.00	0.0000	0.000	0.00	0.02061	0.000	0.000	0.00
7,048	0.50	0.3898	0.0000000	0.0000000	0.000000	0.00000	0.000	0.00	0.00	0.0000	0.000	0.00	0.04750	0.000	0.000	0.00
144,872	0.67	0.0355	0.0000000	0.0000000	0.000000	0.00000	0.000	0.00	0.00	0.0000	0.000	0.00	0.01458	0.000	0.000	0.00
56,618	0.21	0.1525	0.0030424	0.0003957	0.002047	0.00185	0.075	0.57	0.62	0.0861	0.114	0.62	0.00227	0.394	0.092	0.63
149,353	0.22	0.0200	0.0006649	0.0004633	0.000721	0.00460	0.130	0.72	0.96	0.6667	0.000	0.00	0.01144	0.295	0.174	0.63
95,071	0.00	0.1864	0.0009033	0.0006500	0.000202	0.01990	0.099	0.90	0.41	0.0035	0.092	0.75	0.01018	0.182	0.366	0.65
104,320	0.67	0.0355	0.0000000	0.0000000	0.000000	0.00000	0.000	0.00	0.00	0.0000	0.000	0.00	0.00464	0.000	0.000	0.00
67,393	0.00	0.1864	0.0000378	0.0000333	0.000082	0.03695	0.143	0.57	0.95	0.6667	0.000	0.00	0.01357	0.113	0.460	0.51
86,378	0.21	0.1525	0.0000000	0.0000000	0.000000	0.00000	0.000	0.00	0.00	0.0000	0.000	0.00	0.00076	0.000	0.000	0.00
79,000	0.21	0.1525	0.0000000	0.0000000	0.000000	0.00000	0.000	0.00	0.00	0.0000	0.000	0.00	0.00256	0.000	0.000	0.00
156,217	0.00	0.1864	0.0025572	0.0018420	0.002308	0.00080	0.029	0.69	0.75	0.0445	0.166	0.63	0.00664	0.405	0.046	0.70
10,833	0.50	0.3898	0.0000000	0.0000000	0.000000	0.00000	0.000	0.00	0.00	0.0000	0.000	0.00	0.13670	0.000	0.000	0.00
181,503	0.50	0.3898	0.0000668	0.0000729	0.001759	0.00110	0.069	0.96	0.77	0.0285	0.292	0.54	0.01181	0.379	0.061	0.54
19,102	0.00	0.1864	0.0000046	0.0000030	0.000021	0.05555	0.122	0.78	0.81	0.0000	0.361	0.46	0.00327	0.044	0.525	0.39
11,742	0.00	0.1864	0.0004634	0.0004660	0.000182	0.00035	0.046	1.00	0.97	0.0450	0.442	0.38	0.00035	0.173	0.022	0.63
176,301	0.00	0.1864	0.0000037	0.0000038	0.001361	0.02175	0.119	1.00	0.51	0.6667	0.000	0.00	0.12891	0.355	0.379	0.40