

Case Study 2

```
dat <- read.csv("AB_NYC_2019.csv", header = TRUE)

# Split numbers and strings:
col_str <- c("name", "host_name", "neighbourhood_group", "neighbourhood", "room_type",
           "last_review")
col_num <- names(dat)[!names(dat) %in% col_str]

# Find column with missing values
colnames(dat)[colSums(is.na(dat)) > 0] # reviews_per_month has NAs

## [1] "reviews_per_month"

summary(dat$number_of_reviews[which(is.na(dat$reviews_per_month))])

##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
##          0        0        0        0        0        0

sum(dat$last_review[which(is.na(dat$reviews_per_month))] != "") 

## [1] 0

# all missing values in reviews_per_month correspond to 0 in number_of_reviews and blank in last_review
# fill in "reviews_per_month" with 0
dat$reviews_per_month[which(is.na(dat$reviews_per_month))] <- 0

# str(dat)

# Drop useless columns
drops <- c("id", "host_name", "host_id")
dat <- dat[, !(names(dat) %in% drops)]

# log transform price
dat <- dat %>%
  mutate(log_price = ifelse(price == 0, log(price + 5), log(price)))

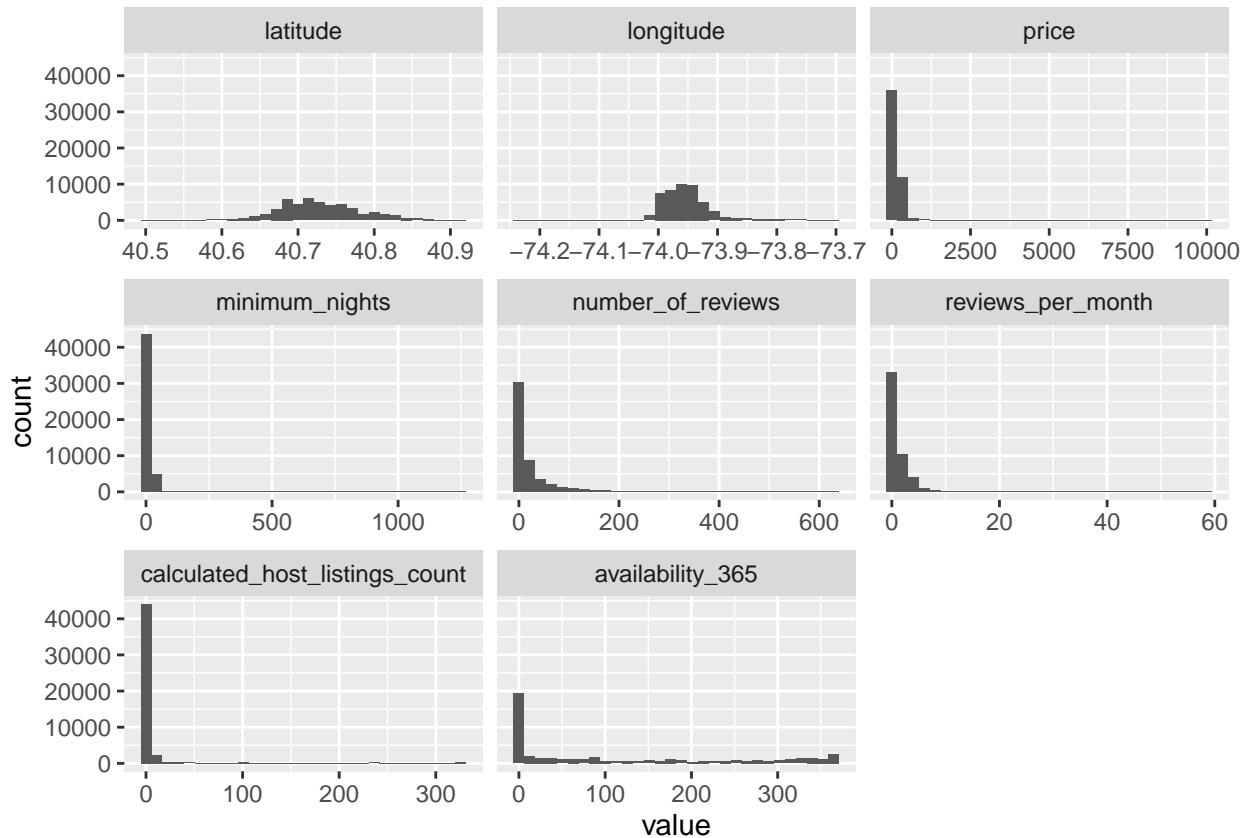
# Check unique values
# unique(dat$neighbourhood_group)
# unique(dat$neighbourhood)
# unique(dat$room_type)

# EDA
# Histograms for numeric variables
d <- melt(dat[, names(dat) %in% col_num])
```

```
## No id variables; using all as measure variables
```

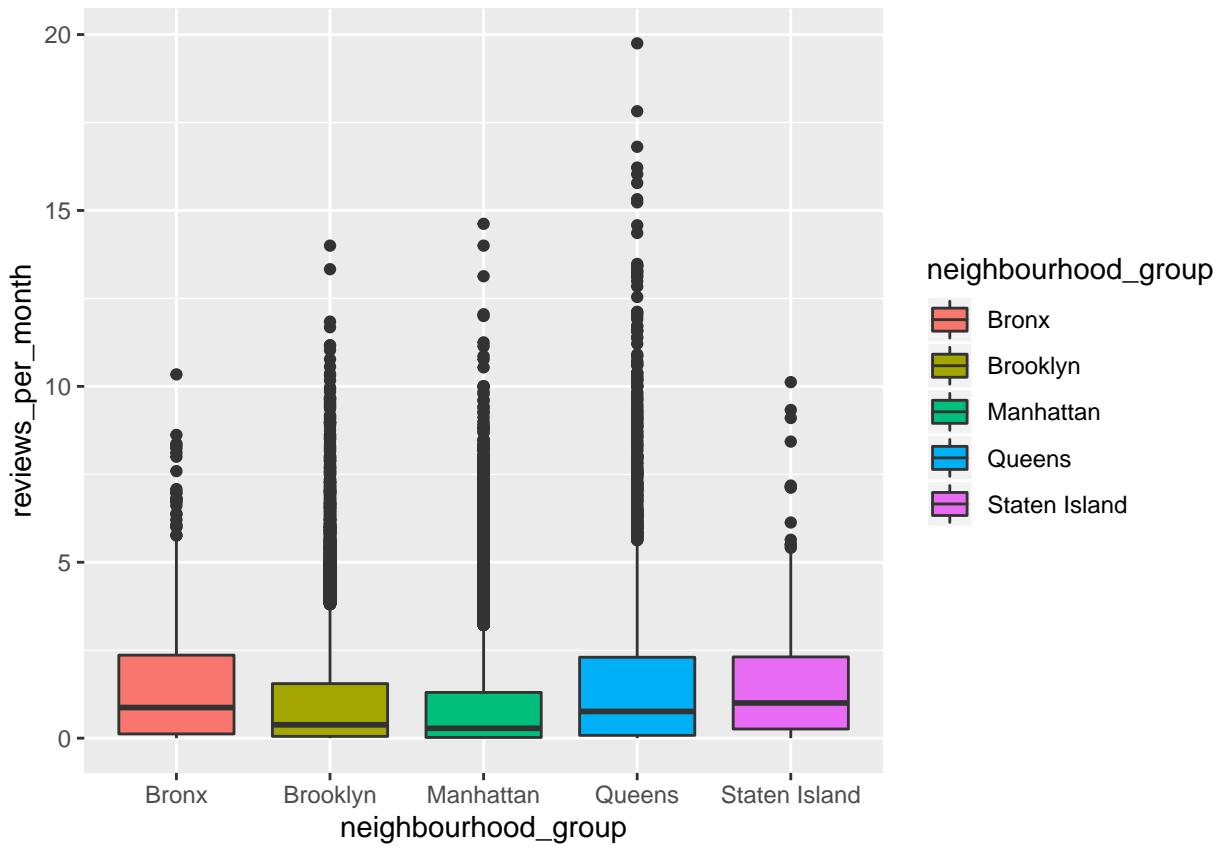
```
ggplot(d,aes(x = value)) +  
  facet_wrap(~variable,scales = "free_x") +  
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



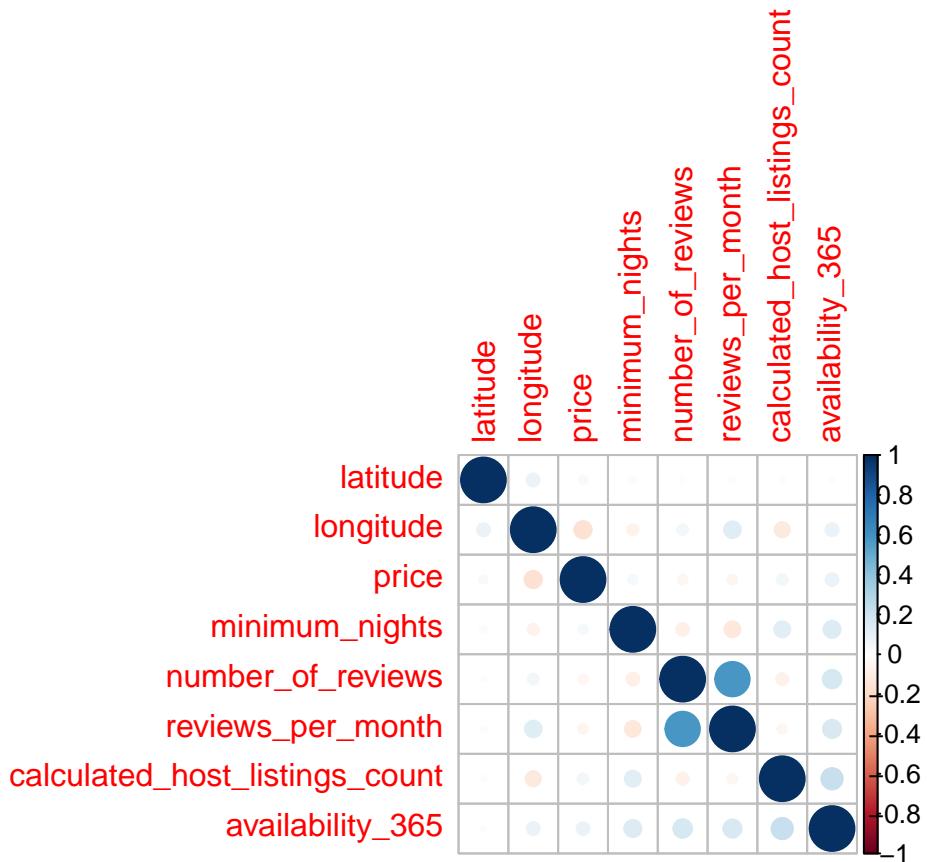
```
# Boxplots (remove extreme values to see distribution)
```

```
ggplot(dat[dat$reviews_per_month<=20,], aes(x=neighbourhood_group, y=reviews_per_month,  
fill=neighbourhood_group)) + geom_boxplot()
```



```
# Manhattan: highest price, fewest reviews
```

```
# Correlation plot
corrplot(cor(dat[, names(dat) %in% col_num]))
```



```
# only number_of_reviews & reviews_per_month have high correlation: expected

# Check if all listings under one host_id are in the same borough # but neighbourhood??
# table_host_borough <- table(dat$host_id, dat$neighbourhood_group)
# dat$host_id[which(apply(as.matrix(table_host_borough), 1, function(x) sum(x!=0))!=1)]

xtabs(~ neighbourhood + neighbourhood_group, data = dat)
```

## neighbourhood	neighbourhood_group					
	Bronx	Brooklyn	Manhattan	Queens	Staten Island	Island
## Allerton	42	0	0	0	0	0
## Arden Heights	0	0	0	0	4	
## Arrochar	0	0	0	0	21	
## Arverne	0	0	0	77	0	
## Astoria	0	0	0	900	0	
## Bath Beach	0	17	0	0	0	
## Battery Park City	0	0	70	0	0	
## Bay Ridge	0	141	0	0	0	
## Bay Terrace	0	0	0	6	0	
## Bay Terrace, Staten Island	0	0	0	0	2	
## Baychester	7	0	0	0	0	
## Bayside	0	0	0	39	0	
## Bayswater	0	0	0	17	0	
## Bedford-Stuyvesant	0	3714	0	0	0	
## Belle Harbor	0	0	0	8	0	
## Bellerose	0	0	0	14	0	

## Belmont	24	0	0	0	0
## Bensonhurst	0	75	0	0	0
## Bergen Beach	0	10	0	0	0
## Boerum Hill	0	177	0	0	0
## Borough Park	0	136	0	0	0
## Breezy Point	0	0	0	3	0
## Briarwood	0	0	0	56	0
## Brighton Beach	0	75	0	0	0
## Bronxdale	19	0	0	0	0
## Brooklyn Heights	0	154	0	0	0
## Brownsville	0	61	0	0	0
## Bull's Head	0	0	0	0	6
## Bushwick	0	2465	0	0	0
## Cambria Heights	0	0	0	26	0
## Canarsie	0	147	0	0	0
## Carroll Gardens	0	233	0	0	0
## Castle Hill	9	0	0	0	0
## Castleton Corners	0	0	0	0	4
## Chelsea	0	0	1113	0	0
## Chinatown	0	0	368	0	0
## City Island	18	0	0	0	0
## Civic Center	0	0	52	0	0
## Claremont Village	28	0	0	0	0
## Clason Point	21	0	0	0	0
## Clifton	0	0	0	0	15
## Clinton Hill	0	572	0	0	0
## Co-op City	2	0	0	0	0
## Cobble Hill	0	99	0	0	0
## College Point	0	0	0	19	0
## Columbia St	0	42	0	0	0
## Concord	0	0	0	0	26
## Concourse	50	0	0	0	0
## Concourse Village	32	0	0	0	0
## Coney Island	0	17	0	0	0
## Corona	0	0	0	64	0
## Crown Heights	0	1564	0	0	0
## Cypress Hills	0	135	0	0	0
## Ditmars Steinway	0	0	0	309	0
## Dongan Hills	0	0	0	0	7
## Douglaston	0	0	0	8	0
## Downtown Brooklyn	0	83	0	0	0
## DUMBO	0	36	0	0	0
## Dyker Heights	0	12	0	0	0
## East Elmhurst	0	0	0	185	0
## East Flatbush	0	500	0	0	0
## East Harlem	0	0	1117	0	0
## East Morrisania	10	0	0	0	0
## East New York	0	218	0	0	0
## East Village	0	0	1853	0	0
## Eastchester	13	0	0	0	0
## Edenwald	13	0	0	0	0
## Edgemere	0	0	0	11	0
## Elmhurst	0	0	0	237	0
## Eltingville	0	0	0	0	3

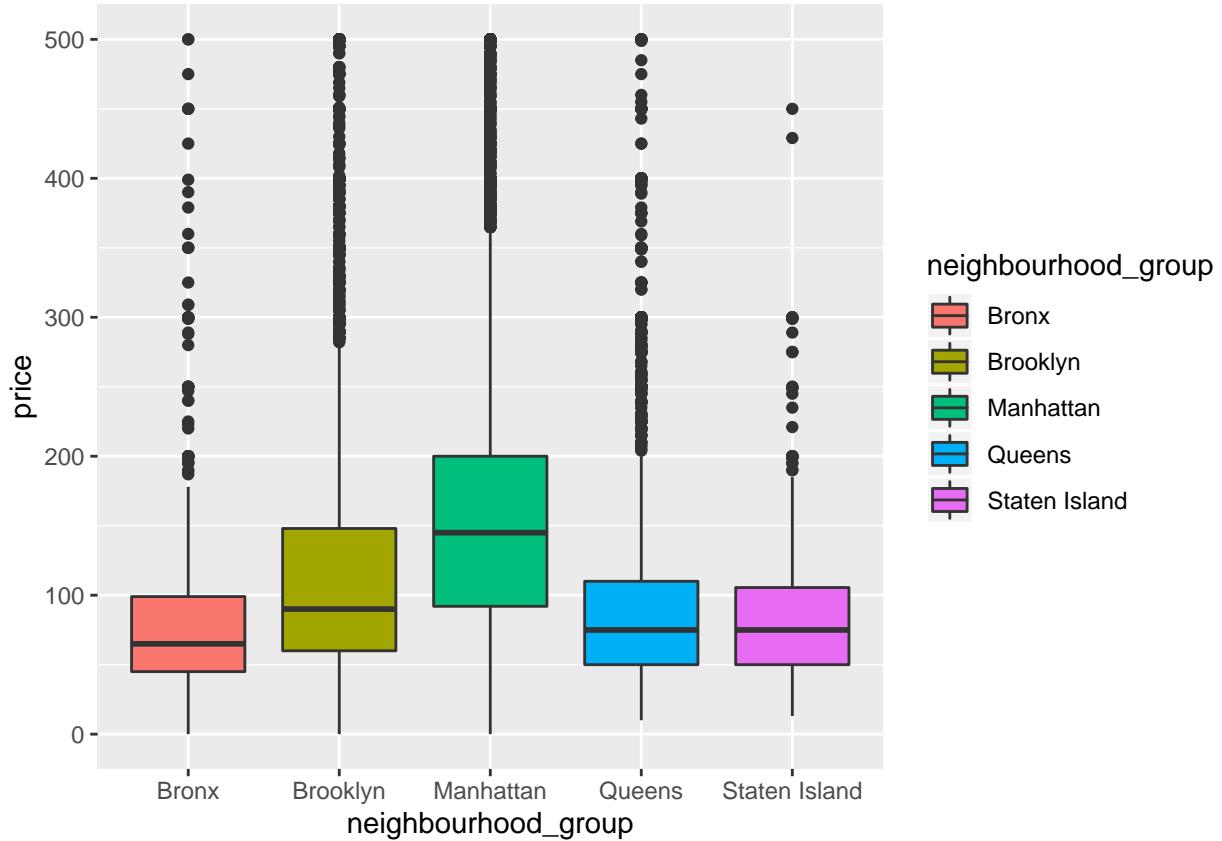
## Emerson Hill	0	0	0	0	5
## Far Rockaway	0	0	0	29	0
## Fieldston	12	0	0	0	0
## Financial District	0	0	744	0	0
## Flatbush	0	621	0	0	0
## Flatiron District	0	0	80	0	0
## Flatlands	0	83	0	0	0
## Flushing	0	0	0	426	0
## Fordham	63	0	0	0	0
## Forest Hills	0	0	0	144	0
## Fort Greene	0	489	0	0	0
## Fort Hamilton	0	55	0	0	0
## Fort Wadsworth	0	0	0	0	1
## Fresh Meadows	0	0	0	32	0
## Glendale	0	0	0	54	0
## Gowanus	0	247	0	0	0
## Gramercy	0	0	338	0	0
## Graniteville	0	0	0	0	3
## Grant City	0	0	0	0	6
## Gravesend	0	68	0	0	0
## Great Kills	0	0	0	0	10
## Greenpoint	0	1115	0	0	0
## Greenwich Village	0	0	392	0	0
## Grymes Hill	0	0	0	0	7
## Harlem	0	0	2658	0	0
## Hell's Kitchen	0	0	1958	0	0
## Highbridge	27	0	0	0	0
## Hollis	0	0	0	14	0
## Holliswood	0	0	0	4	0
## Howard Beach	0	0	0	20	0
## Howland Hook	0	0	0	0	2
## Huguenot	0	0	0	0	3
## Hunts Point	18	0	0	0	0
## Inwood	0	0	252	0	0
## Jackson Heights	0	0	0	186	0
## Jamaica	0	0	0	231	0
## Jamaica Estates	0	0	0	19	0
## Jamaica Hills	0	0	0	8	0
## Kensington	0	175	0	0	0
## Kew Gardens	0	0	0	32	0
## Kew Gardens Hills	0	0	0	26	0
## Kingsbridge	70	0	0	0	0
## Kips Bay	0	0	470	0	0
## Laurelton	0	0	0	18	0
## Lighthouse Hill	0	0	0	0	2
## Little Italy	0	0	121	0	0
## Little Neck	0	0	0	5	0
## Long Island City	0	0	0	537	0
## Longwood	62	0	0	0	0
## Lower East Side	0	0	911	0	0
## Manhattan Beach	0	8	0	0	0
## Marble Hill	0	0	12	0	0
## Mariners Harbor	0	0	0	0	8
## Maspeth	0	0	0	110	0

## Melrose	10	0	0	0	0
## Middle Village	0	0	0	31	0
## Midland Beach	0	0	0	0	6
## Midtown	0	0	1545	0	0
## Midwood	0	109	0	0	0
## Mill Basin	0	4	0	0	0
## Morningside Heights	0	0	346	0	0
## Morris Heights	17	0	0	0	0
## Morris Park	15	0	0	0	0
## Morrisania	18	0	0	0	0
## Mott Haven	60	0	0	0	0
## Mount Eden	6	0	0	0	0
## Mount Hope	20	0	0	0	0
## Murray Hill	0	0	485	0	0
## Navy Yard	0	14	0	0	0
## Neponsit	0	0	0	3	0
## New Brighton	0	0	0	0	5
## New Dorp	0	0	0	0	1
## New Dorp Beach	0	0	0	0	5
## New Springville	0	0	0	0	8
## NoHo	0	0	78	0	0
## Nolita	0	0	253	0	0
## North Riverdale	10	0	0	0	0
## Norwood	31	0	0	0	0
## Oakwood	0	0	0	0	5
## Olinville	4	0	0	0	0
## Ozone Park	0	0	0	62	0
## Park Slope	0	506	0	0	0
## Parkchester	39	0	0	0	0
## Pelham Bay	17	0	0	0	0
## Pelham Gardens	28	0	0	0	0
## Port Morris	46	0	0	0	0
## Port Richmond	0	0	0	0	9
## Prince's Bay	0	0	0	0	4
## Prospect Heights	0	357	0	0	0
## Prospect-Lefferts Gardens	0	535	0	0	0
## Queens Village	0	0	0	60	0
## Randall Manor	0	0	0	0	19
## Red Hook	0	79	0	0	0
## Rego Park	0	0	0	106	0
## Richmond Hill	0	0	0	94	0
## Richmondtown	0	0	0	0	1
## Ridgewood	0	0	0	423	0
## Riverdale	11	0	0	0	0
## Rockaway Beach	0	0	0	56	0
## Roosevelt Island	0	0	77	0	0
## Rosebank	0	0	0	0	7
## Rosedale	0	0	0	59	0
## Rossville	0	0	0	0	1
## Schuylerville	13	0	0	0	0
## Sea Gate	0	7	0	0	0
## Sheepshead Bay	0	164	0	0	0
## Shore Acres	0	0	0	0	7
## Silver Lake	0	0	0	0	2

## SoHo	0	0	358	0	0
## Soundview	15	0	0	0	0
## South Beach	0	0	0	0	8
## South Ozone Park	0	0	0	40	0
## South Slope	0	284	0	0	0
## Springfield Gardens	0	0	0	85	0
## Spuyten Duyvil	4	0	0	0	0
## St. Albans	0	0	0	76	0
## St. George	0	0	0	0	48
## Stapleton	0	0	0	0	27
## Stuyvesant Town	0	0	37	0	0
## Sunnyside	0	0	0	363	0
## Sunset Park	0	390	0	0	0
## Theater District	0	0	288	0	0
## Throgs Neck	24	0	0	0	0
## Todt Hill	0	0	0	0	4
## Tompkinsville	0	0	0	0	42
## Tottenville	0	0	0	0	7
## Tremont	11	0	0	0	0
## Tribeca	0	0	177	0	0
## Two Bridges	0	0	72	0	0
## Unionport	7	0	0	0	0
## University Heights	21	0	0	0	0
## Upper East Side	0	0	1798	0	0
## Upper West Side	0	0	1971	0	0
## Van Nest	11	0	0	0	0
## Vinegar Hill	0	34	0	0	0
## Wakefield	50	0	0	0	0
## Washington Heights	0	0	899	0	0
## West Brighton	0	0	0	0	18
## West Farms	2	0	0	0	0
## West Village	0	0	768	0	0
## Westchester Square	10	0	0	0	0
## Westerleigh	0	0	0	0	2
## Whitestone	0	0	0	11	0
## Williamsbridge	40	0	0	0	0
## Williamsburg	0	3920	0	0	0
## Willowbrook	0	0	0	0	1
## Windsor Terrace	0	157	0	0	0
## Woodhaven	0	0	0	88	0
## Woodlawn	11	0	0	0	0
## Woodrow	0	0	0	0	1
## Woodside	0	0	0	235	0

```
# neighbourhood_group & neighbourhood are nested
```

```
# Boxplots - price by borough (remove extreme values)
ggplot(dat[dat$price<=500,], aes(x=neighbourhood_group, y=price,
fill=neighbourhood_group)) + geom_boxplot()
```



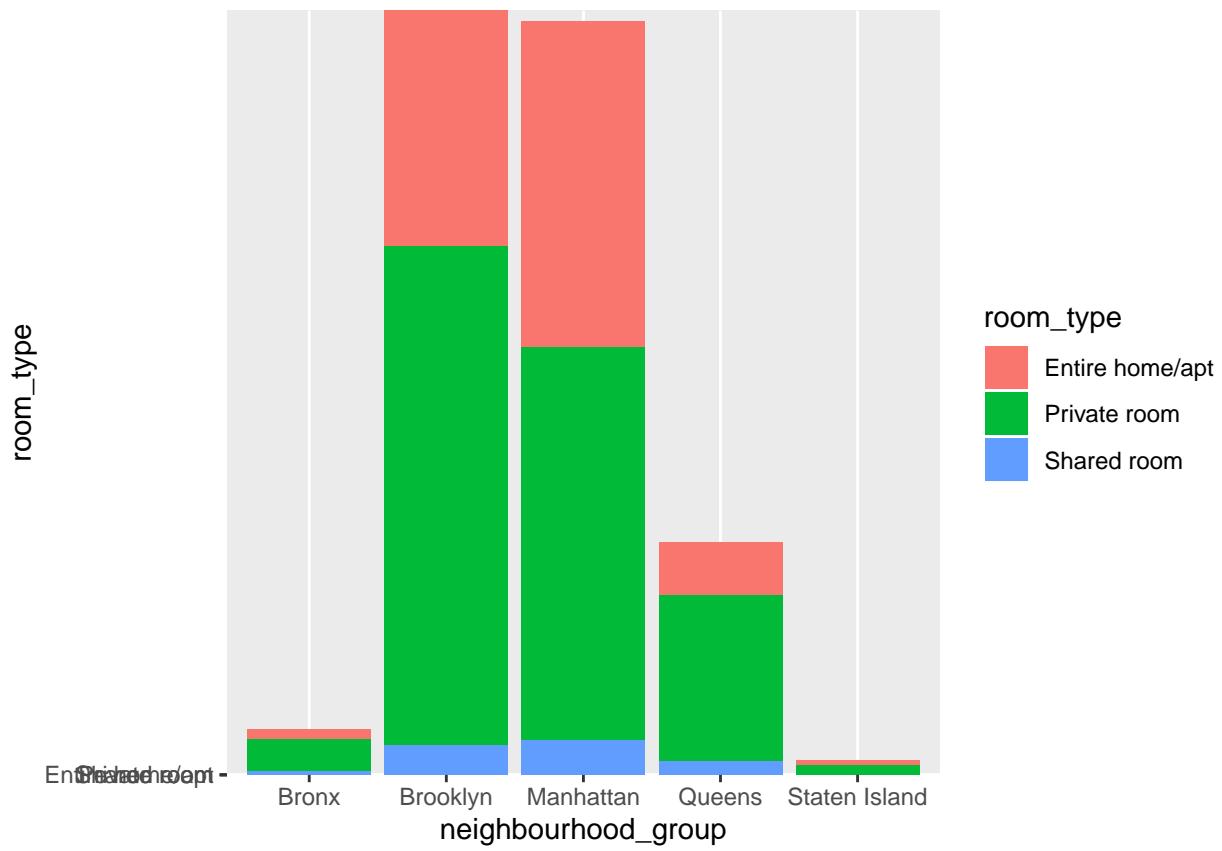
```
anova(lm(price~neighbourhood,data=dat))
```

```
## Analysis of Variance Table
##
## Response: price
##              Df     Sum Sq Mean Sq F value    Pr(>F)
## neighbourhood  220  189750135  862501   15.961 < 2.2e-16 ***
## Residuals      48674 2630163656   54036
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

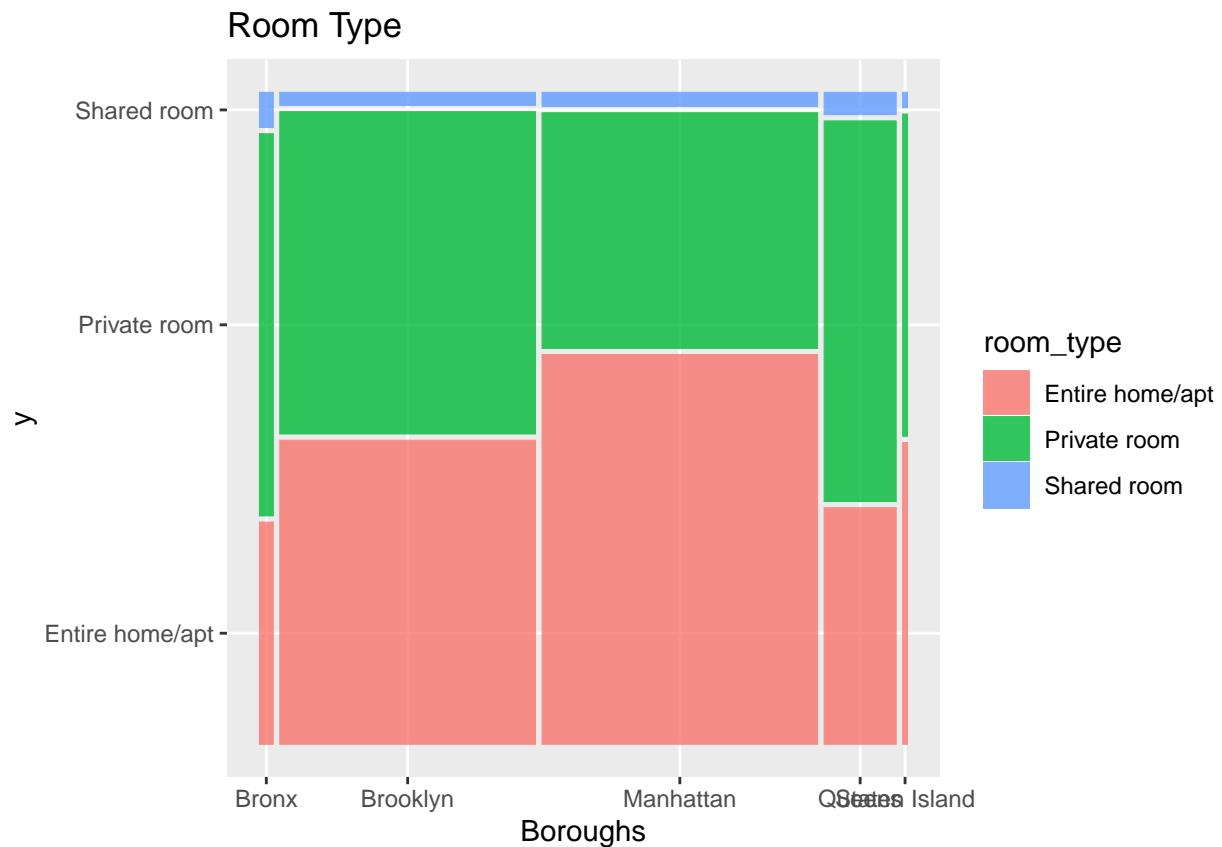
```
anova(lm(price~neighbourhood_group,data=dat))
```

```
## Analysis of Variance Table
##
## Response: price
##              Df     Sum Sq Mean Sq F value    Pr(>F)
## neighbourhood_group      4    79590956 19897739   354.99 < 2.2e-16 ***
## Residuals                  48890 2740322834   56051
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

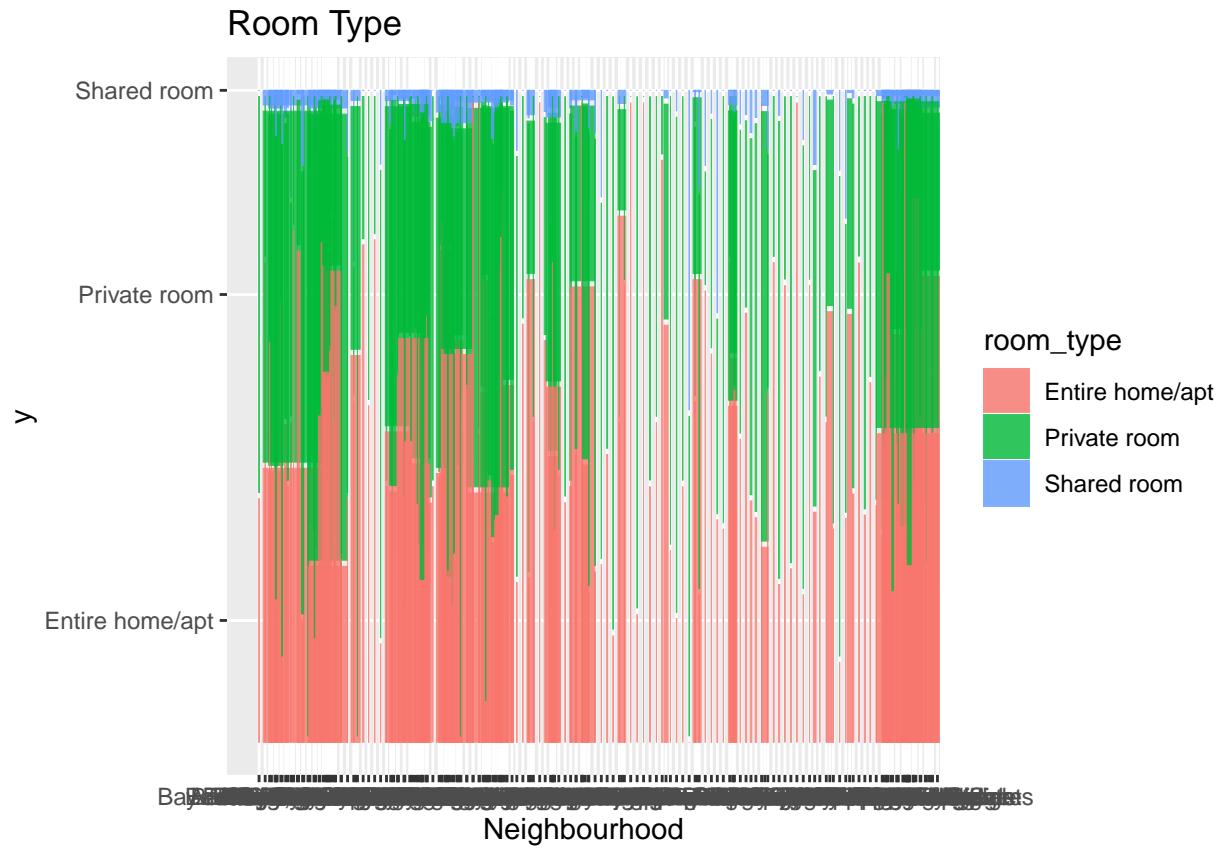
```
# Barplots & Mosaic plot - room type by borough
ggplot(data=dat, aes(x = neighbourhood_group, y = room_type, fill = room_type)) +
  geom_bar(stat="identity")
```



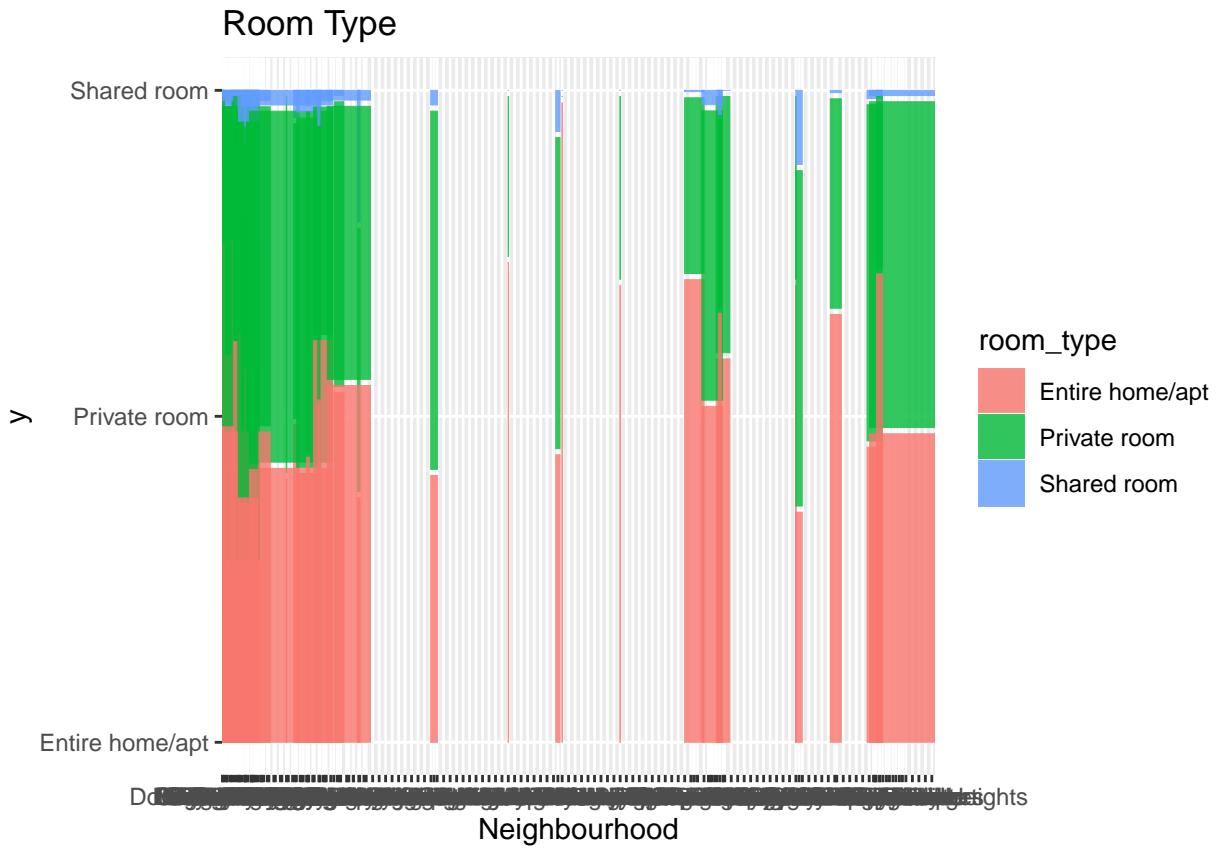
```
# boroughs
ggplot(data = dat) +
  geom_mosaic(aes(x = product(room_type, neighbourhood_group), fill=room_type), na.rm=TRUE) +
  labs(x="Boroughs", title='Room Type')
```



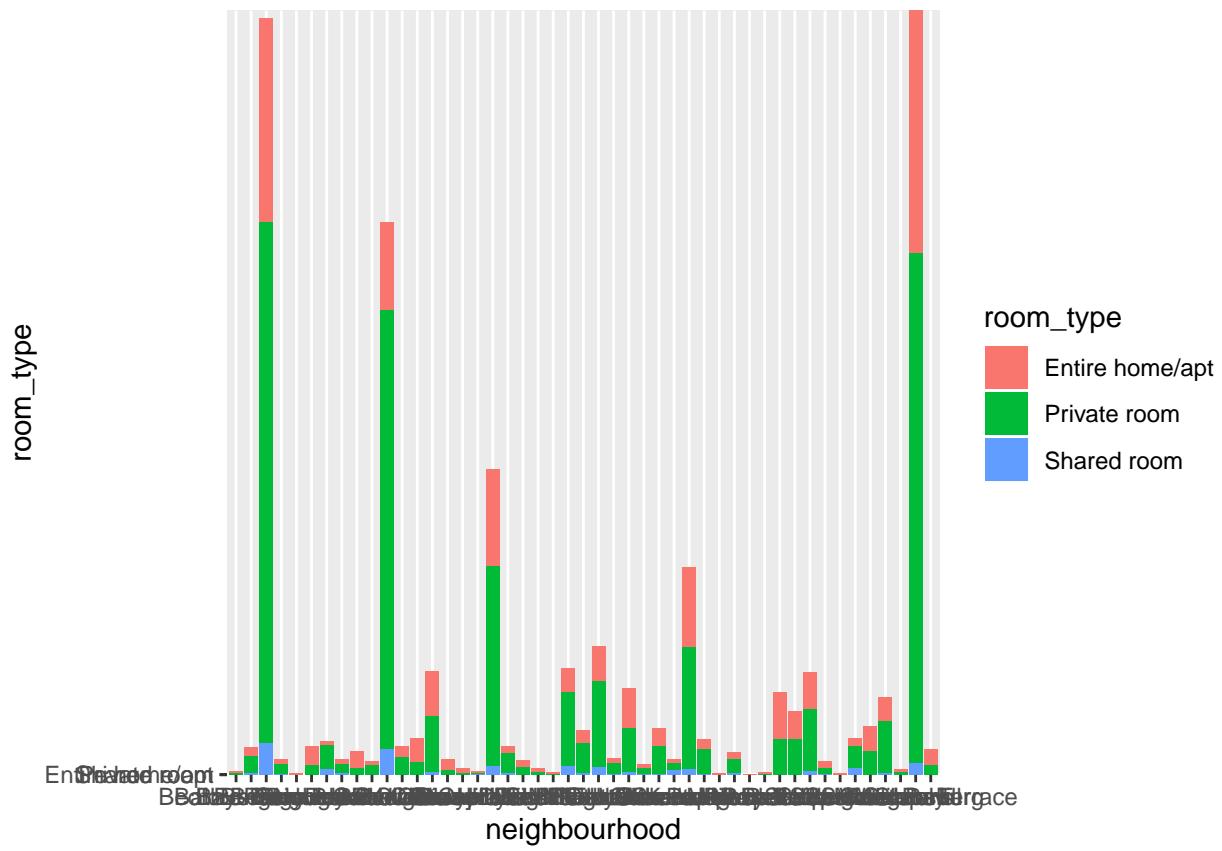
```
# all neighbourhood -> difficult to see
ggplot(data = dat) +
  geom_mosaic(aes(x = product(room_type, neighbourhood), fill=room_type), na.rm=TRUE) +
  labs(x="Neighbourhood", title='Room Type')
```



```
# plot by boroughs
ggplot(data = dat %>%
  filter(neighbourhood_group=="Brooklyn")) +
  geom_mosaic(aes(x = product(room_type, neighbourhood), fill=room_type), na.rm=TRUE) +
  labs(x="Neighbourhood", title='Room Type')
```



```
ggplot(data = dat %>%
  filter(neighbourhood_group=="Brooklyn"),
  aes(x = neighbourhood, y = room_type, fill = room_type)) +
  geom_bar(stat="identity")
```



```

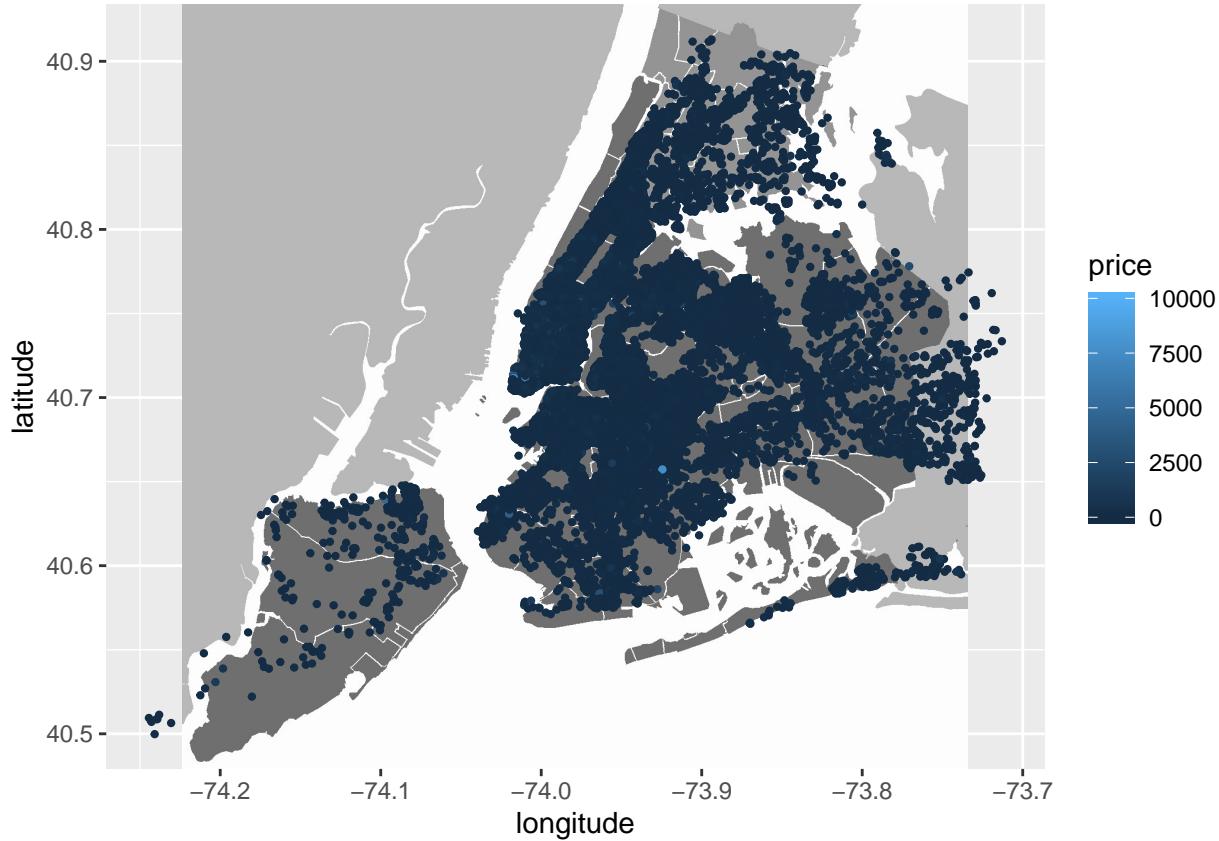
library(magick)

## Linking to ImageMagick 6.9.9.39
## Enabled features: cairo, fontconfig, freetype, lcms, pango, rsvg, webp
## Disabled features: fftw, ghostscript, x11

library(grid)
img <- image_read("New_York_City_.png")
g <- rasterGrob(img)

ggplot() +
  annotation_custom(g) +
  geom_point(data = dat, mapping = aes(x = longitude, y = latitude, colour = price), size=0.8)

```



```
#ggmap
```

```
availability <- dat$availability_365==0
last_review_year <- as.numeric(substr(dat$last_review, start = 1, stop = 4))
review_per_month <- dat$reviews_per_month
# table(last_review_year, availability)
# table(review_per_month, availability)

lm_1 <- lmer(log_price ~ latitude + longitude + room_type + reviews_per_month +
               calculated_host_listings_count + availability_365 +
               (1 | neighbourhood_group) + (1 | neighbourhood), data = dat,
               REML=TRUE)
```

```
## Warning: Some predictor variables are on very different scales: consider
## rescaling
```

```
## Warning: Some predictor variables are on very different scales: consider
## rescaling
```

```
summary(lm_1)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: log_price ~ latitude + longitude + room_type + reviews_per_month +
```

```

##      calculated_host_listings_count + availability_365 + (1 |
##      neighbourhood_group) + (1 | neighbourhood)
## Data: dat
##
## REML criterion at convergence: 66794
##
## Scaled residuals:
##      Min     1Q Median     3Q    Max
## -7.7388 -0.6166 -0.1089  0.4695 10.7041
##
## Random effects:
## Groups           Name        Variance Std.Dev.
## neighbourhood   (Intercept) 0.04578  0.2140
## neighbourhood_group (Intercept) 0.20555  0.4534
## Residual          0.22666  0.4761
## Number of obs: 48895, groups: neighbourhood, 221; neighbourhood_group, 5
##
## Fixed effects:
##                               Estimate Std. Error      df t value
## (Intercept)            -2.196e+02  2.131e+01 8.950e+02 -10.306
## latitude                -4.098e-01  2.746e-01 6.356e+02  -1.492
## longitude               -3.261e+00  2.364e-01 1.131e+03 -13.794
## room_typePrivate room  -7.063e-01  4.611e-03 4.882e+04 -153.186
## room_typeShared room   -1.082e+00  1.456e-02 4.885e+04  -74.271
## reviews_per_month       -1.116e-02  1.411e-03 4.884e+04  -7.904
## calculated_host_listings_count -6.268e-04  7.623e-05 4.884e+04  -8.223
## availability_365        7.140e-04  1.756e-05 4.878e+04  40.654
##                               Pr(>|t|)
## (Intercept) < 2e-16 ***
## latitude      0.136
## longitude    < 2e-16 ***
## room_typePrivate room < 2e-16 ***
## room_typeShared room < 2e-16 ***
## reviews_per_month    2.76e-15 ***
## calculated_host_listings_count < 2e-16 ***
## availability_365      < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##              (Intr) latitd longtd rm_tPr rm_tSr rvws__ clc__
## latitude      -0.574
## longitude      0.852 -0.060
## rm_tPrvtr   -0.010 -0.029 -0.031
## rm_tPShrdr  -0.007 -0.003 -0.010  0.161
## rvws_pr_mnt -0.039  0.018 -0.036  0.007  0.021
## clcltd_hs__  0.004  0.008  0.011  0.066  0.030  0.074
## avlblty_365 -0.035  0.019 -0.030 -0.005 -0.051 -0.148 -0.212
## fit warnings:
## Some predictor variables are on very different scales: consider rescaling

anova(lm_1) # test fixed effect

```

```
## Type III Analysis of Variance Table with Satterthwaite's method
```

```

##                                     Sum Sq Mean Sq NumDF DenDF    F value    Pr(>F)
## latitude                           0.5   0.50     1   636   2.2273   0.1361
## longitude                          43.1  43.13     1  1131 190.2773 < 2.2e-16
## room_type                          5892.0 2946.00     2 48837 12997.5413 < 2.2e-16
## reviews_per_month                  14.2   14.16     1 48842   62.4705 2.761e-15
## calculated_host_listings_count   15.3   15.32     1 48836   67.6126 < 2.2e-16
## availability_365                  374.6  374.60     1 48785 1652.7086 < 2.2e-16
##
## latitude                           ***
## longitude                          ***
## room_type                          ***
## reviews_per_month                  ***
## calculated_host_listings_count   ***
## availability_365                  ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

rand(lm_1) # test random effect

## Warning: Some predictor variables are on very different scales: consider
## rescaling

## Warning: Some predictor variables are on very different scales: consider
## rescaling

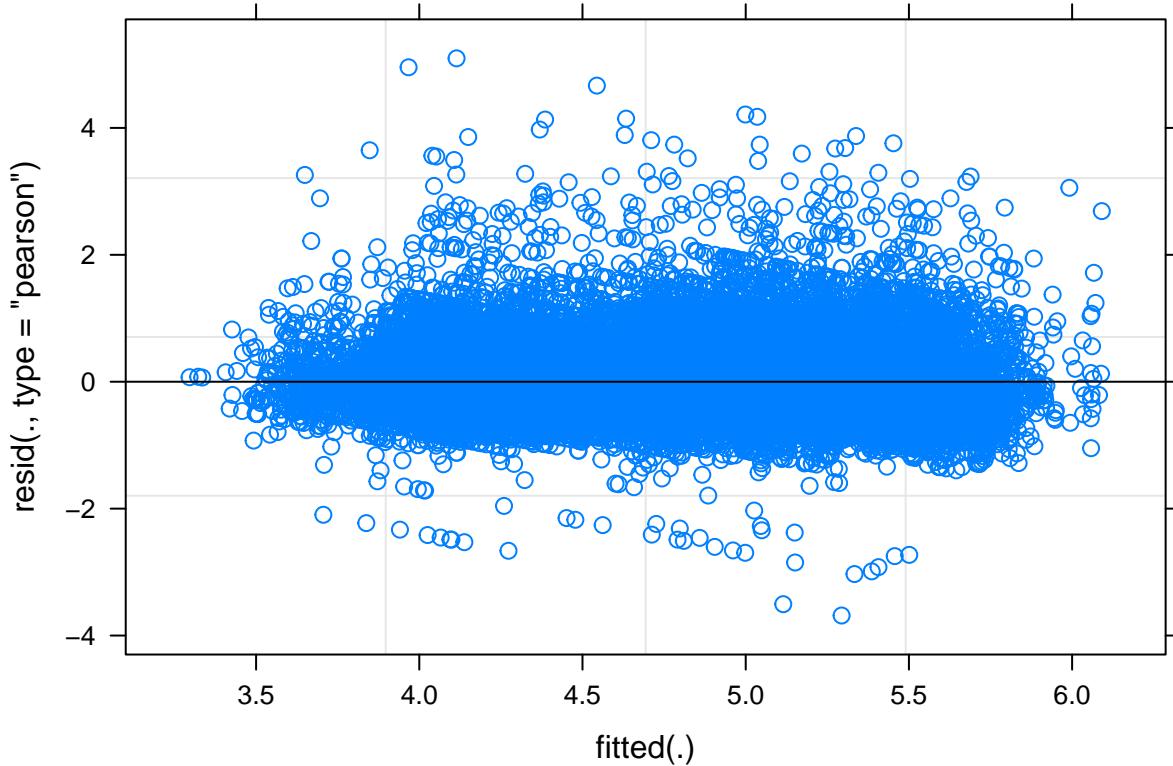
## Warning: Some predictor variables are on very different scales: consider
## rescaling

## Warning: Some predictor variables are on very different scales: consider
## rescaling

## ANOVA-like table for random-effects: Single term deletions
##
## Model:
## log_price ~ latitude + longitude + room_type + reviews_per_month +
##           calculated_host_listings_count + availability_365 + (1 |
##           neighbourhood_group) + (1 | neighbourhood)
##           npar logLik   AIC      LRT Df Pr(>Chisq)
## <none>            11 -33397 66816
## (1 | neighbourhood_group) 10 -33490 67000 186.2 1 < 2.2e-16 ***
## (1 | neighbourhood)    10 -35647 71315 4500.6 1 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

plot(lm_1) # residual plot

```



```

library(tidytext)
data(stop_words)

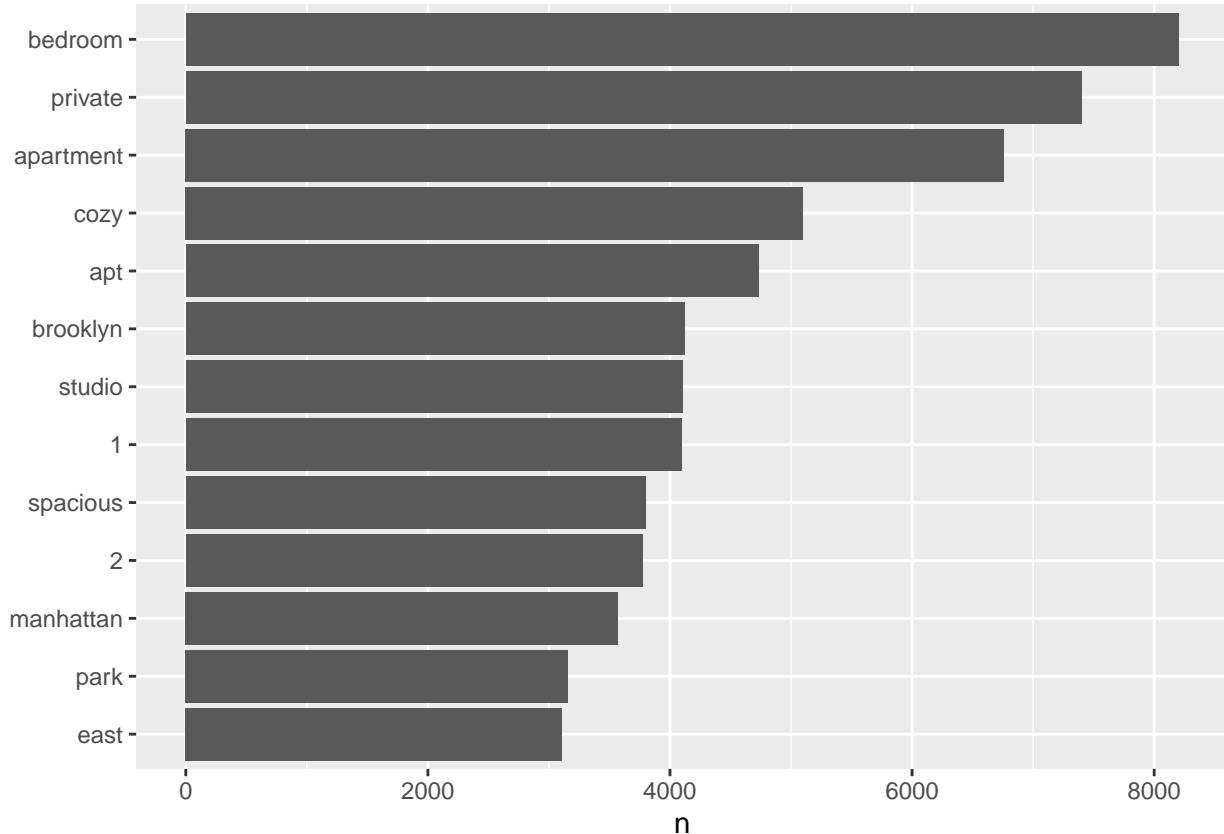
dat$name <- as.character(dat$name)
name_df <- tibble(line = 1:nrow(dat), dat$name)

tokens <- name_df %>%
  unnest_tokens(word, dat$name) %>%
  anti_join(stop_words)

## Joining, by = "word"

tokens %>%
  count(word, sort = TRUE) %>%
  filter(n > 3000) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_col() +
  xlab(NULL) +
  coord_flip()

```



```

# library(caret)
# set.seed(1)
# train_index <- sample(1:nrow(dat), 0.9 * nrow(dat))
# dat_train <- dat[train_index, ]
# dat_test <- dat[-train_index, ]
# # fit a random forest model (using ranger)
# rf_fit <- train(log_price ~ latitude + longitude + room_type + reviews_per_month +
# #                               calculated_host_listings_count + availability_365 +
# #                               neighbourhood_group + neighbourhood, data = dat_train,
# #                               method = "ranger")
# rf_fit
# rf_pred <- predict(rf_fit, dat_test)
# dat_VarImp = varImp(rf_fit)

library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
## 
##     combine

```

```

## The following object is masked from 'package:ggplot2':
##
##      margin

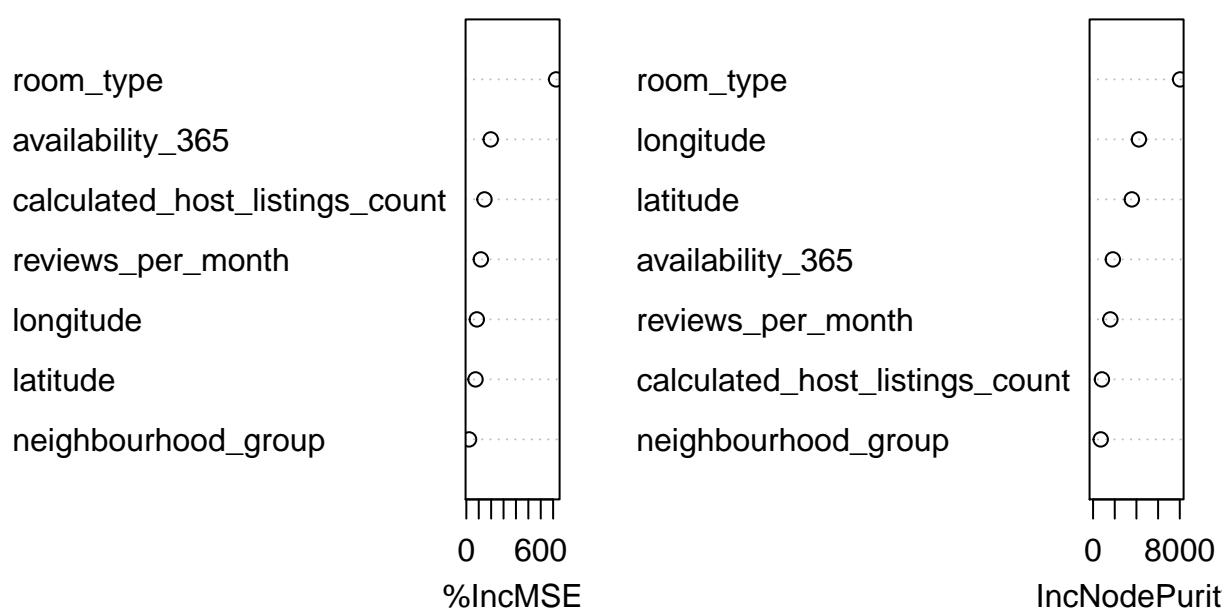
set.seed(1)
train_index <- sample(1:nrow(dat), 0.9 * nrow(dat))
dat_train <- dat[train_index, ]
dat_test <- dat[-train_index, ]
# fit a random forest model
rf_fit <- randomForest(log_price ~ latitude + longitude + room_type + reviews_per_month +
                         calculated_host_listings_count + availability_365 +
                         neighbourhood_group, data = dat_train,
                         ntree = 500, mtry = 6,
                         importance = TRUE) # cannot use neighbourhood: # levels > 53
importance(rf_fit)

##                                     %IncMSE IncNodePurity
## latitude                         74.08658   3575.9157
## longitude                        84.49337   4230.5005
## room_type                         724.07856   8030.9396
## reviews_per_month                  116.70760   1594.1227
## calculated_host_listings_count    146.23027   816.5953
## availability_365                  196.84354   1825.2299
## neighbourhood_group                21.93242   705.7680

varImpPlot(rf_fit)

```

rf_fit



```
rf_pred <- predict(rf_fit, dat_test)
plot(dat_test$log_price, rf_pred)
abline(0,1,col="red")
```

