Final project

Import all library

Import data

```
final_data = as_tibble(read.csv("./input/all_data_v3.csv"))
join_tb = as_tibble(read.csv("./input/all_jointed_data.csv"))
```

4.1 Check missing values

```
# Data overview
data_dict(final_data, print_table = 'Yes')
```

ableType	MissingValues	n	mean	sd	median	se	n
nteger	0	8576	216.32	122.07	235.5	1.32	
umeric	0	8576	3.52	0.46	3.54	0.01	
nteger	0	8576	302.86	347.38	192	3.75	
umeric	0	8576	2.22	0.2	2.23	0	
nteger	0	8576	26076432.06	14122953.66	27776148	152504.67	27
umeric	0	8576	34.07	0.07	34.07	0	33
umeric	0	8576	−118.37	0.09	−118.36	0	−11
aracter	0	8576					
aracter	0	8576					
aracter	0	8576					
nteger	0	8576	190.77	261.88	120	2.83	1
aracter	0	8576					

4.2

```
distinct(final_data, room_type)
```

```
## # A tibble: 4 x 1
##   room_type
```

```
##   <chr>
## 1 Private room
## 2 Entire home/apt
## 3 Shared room
## 4 Hotel room
```

```
distinct(final_data, neighbourhood_group)
```

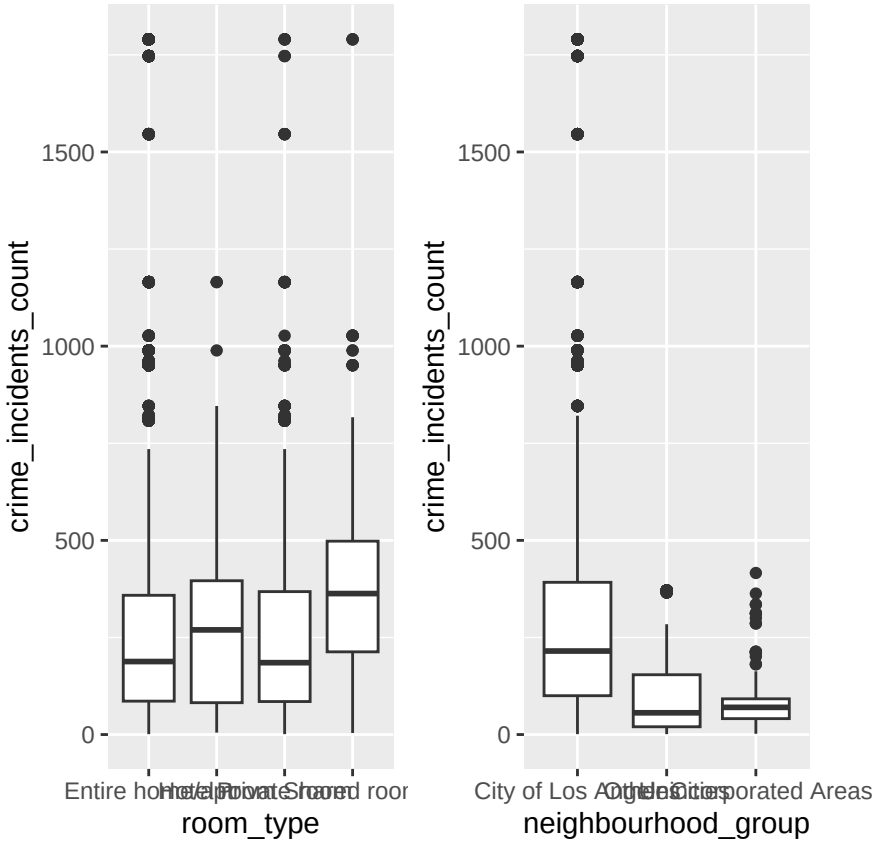
```
## # A tibble: 3 x 1
##   neighbourhood_group
##   <chr>
## 1 City of Los Angeles
## 2 Other Cities
## 3 Unincorporated Areas
```

```
distinct(final_data, neighbourhood)
```

```
## # A tibble: 141 x 1
##    neighbourhood
##    <chr>
##  1 Hollywood
##  2 Hollywood Hills
##  3 Hollywood Hills West
##  4 Del Rey
##  5 Culver City
##  6 Atwater Village
##  7 Glendale
##  8 Venice
##  9 Marina del Rey
## 10 Santa Monica
## # i 131 more rows
```

crime incidnet

```
n1 = ggplot(final_data) + geom_boxplot(aes(x=room_type, y=crime_incidents_count))
n2 = ggplot(final_data) + geom_boxplot(aes(x=neighbourhood_group, y=crime_incidents_count))
ggarrange(n1, n2, ncol=3)
```

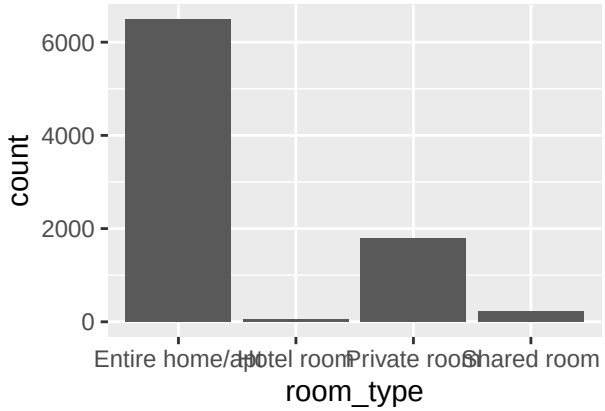


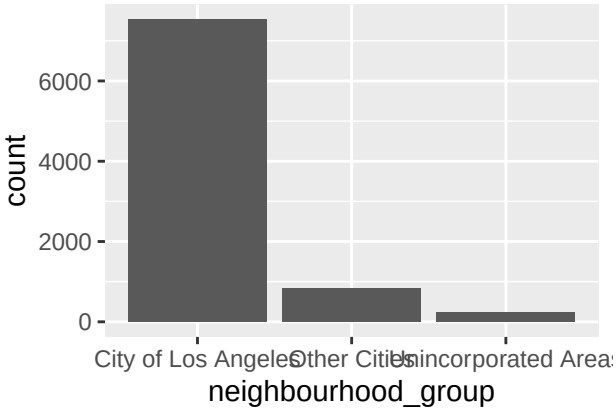
4.3.1. Univariate Summary of Factors

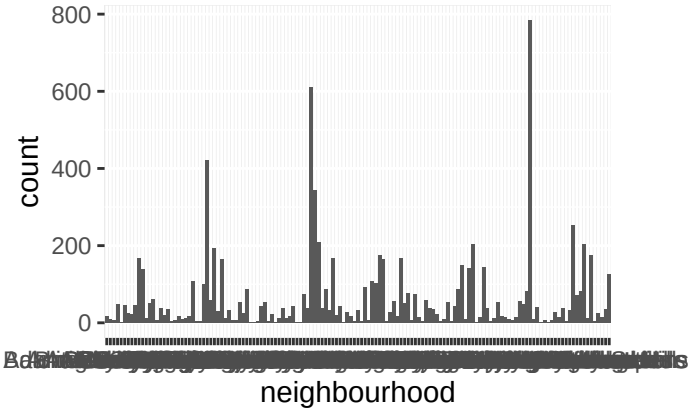
```
# summary(final_data)
colnames(final_data %>% select_if((is.character)))
```

```
## [1] "name"                "neighbourhood_group" "neighbourhood"
## [4] "room_type"
```

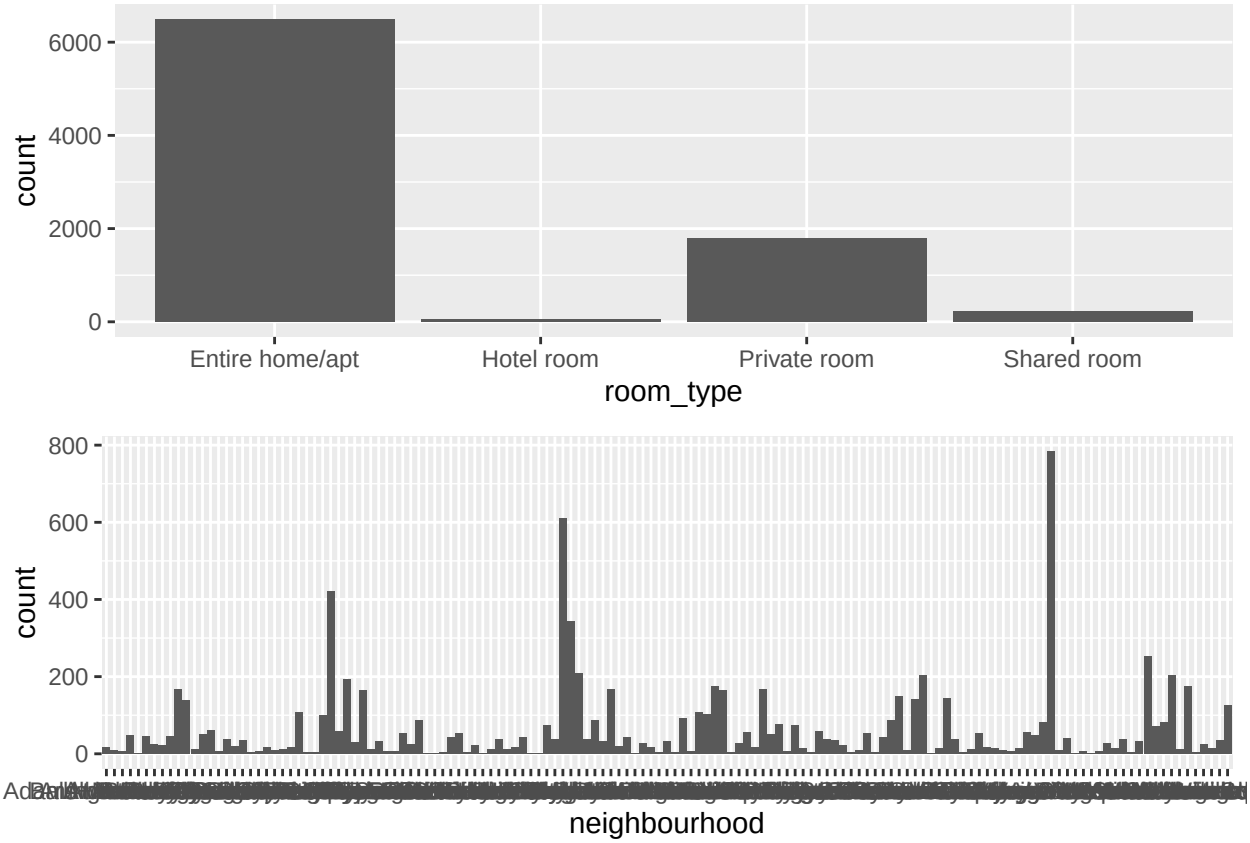
```
g1 = ggplot(final_data) + geom_bar(aes(x=room_type))
g2 = ggplot(final_data) + geom_bar(aes(x=neighbourhood_group))
g3 = ggplot(final_data) + geom_bar(aes(x=neighbourhood))
ggarrange(g1, g2, g3)
```








```
ggarrange(g1, g3, nrow=2)
```



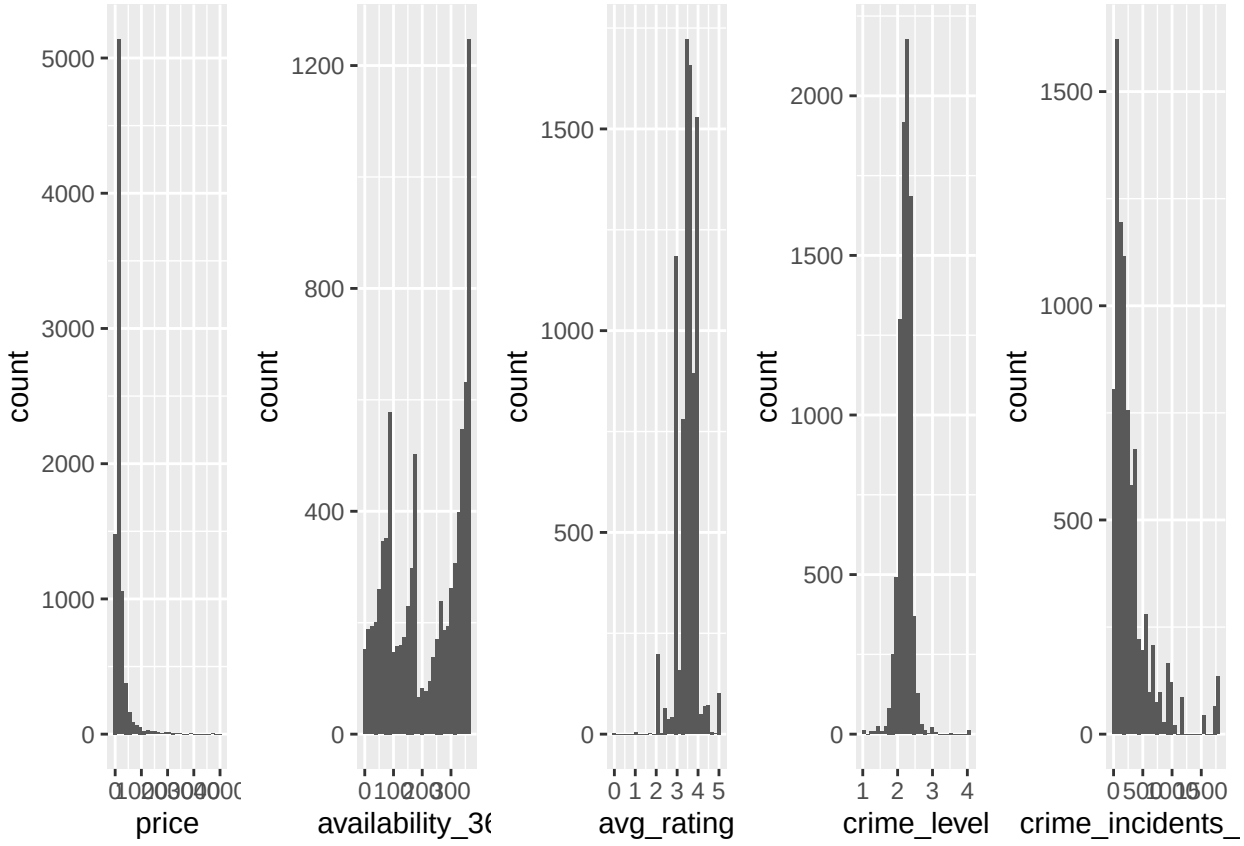
4.3.2. Explore Numeric Attributes

```
####### plot all #########
colnames(final_data %>% select_if((is.numeric)))
```

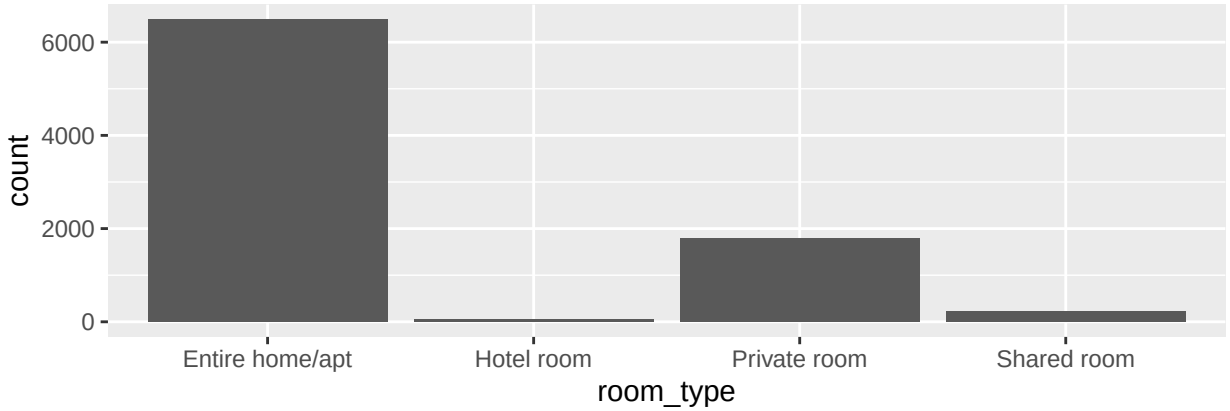
```
## [1] "id"                    "latitude"              "longitude"
## [4] "price"                 "availability_365"      "avg_rating"
## [7] "crime_level"           "crime_incidents_count"
```

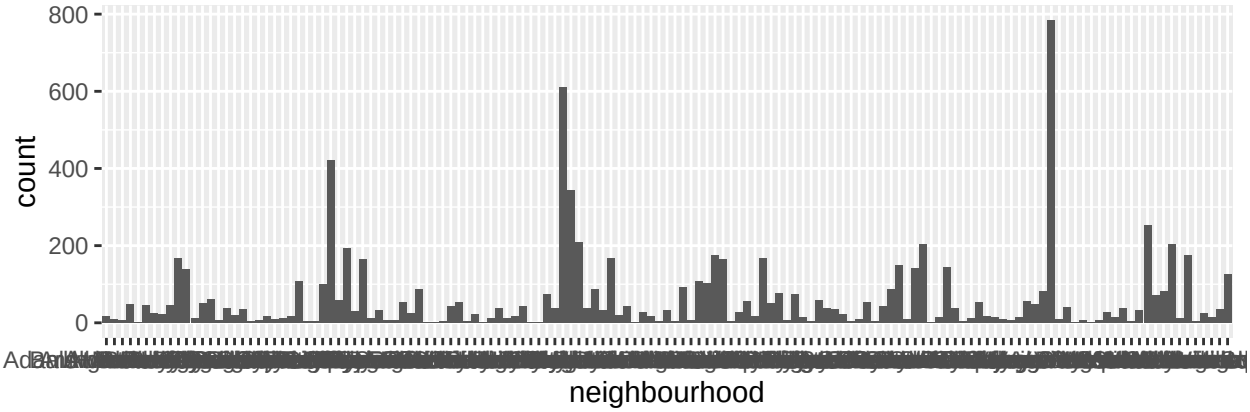
```
n1 = ggplot(final_data) + geom_histogram(aes(x=price))
n2 = ggplot(final_data) + geom_histogram(aes(x=availability_365))
n3 = ggplot(final_data) + geom_histogram(aes(x=avg_rating))
n4 = ggplot(final_data) + geom_histogram(aes(x=crime_level))
n5 = ggplot(final_data) + geom_histogram(aes(x=crime_incidents_count))
ggarrange(n1, n2, n3, n4, n5, ncol=5, nrow=1)
```

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




```
ggarrange(g1, g3, nrow=2)
```






```
####### mean min max SD #########
numeric_data = final_data %>% select(price, availability_365, avg_rating, crime_level, crime_incidents_
summarize_numeric(numeric_data)
```

```
##               Attribute Missing Values Unique Values       Mean Min  Max
## 1                 price              0           712 190.769007  10 4000
## 2      availability_365              0           365 216.315182   1  365
## 3            avg_rating              0           471   3.523852   0    5
## 4           crime_level              0           664   2.224057   1    4
## 5 crime_incidents_count              0           344 302.859492   1 1790
##            SD
## 1 261.8774621
## 2 122.0673375
## 3   0.4638161
## 4   0.1982071
## 5 347.3810503
```

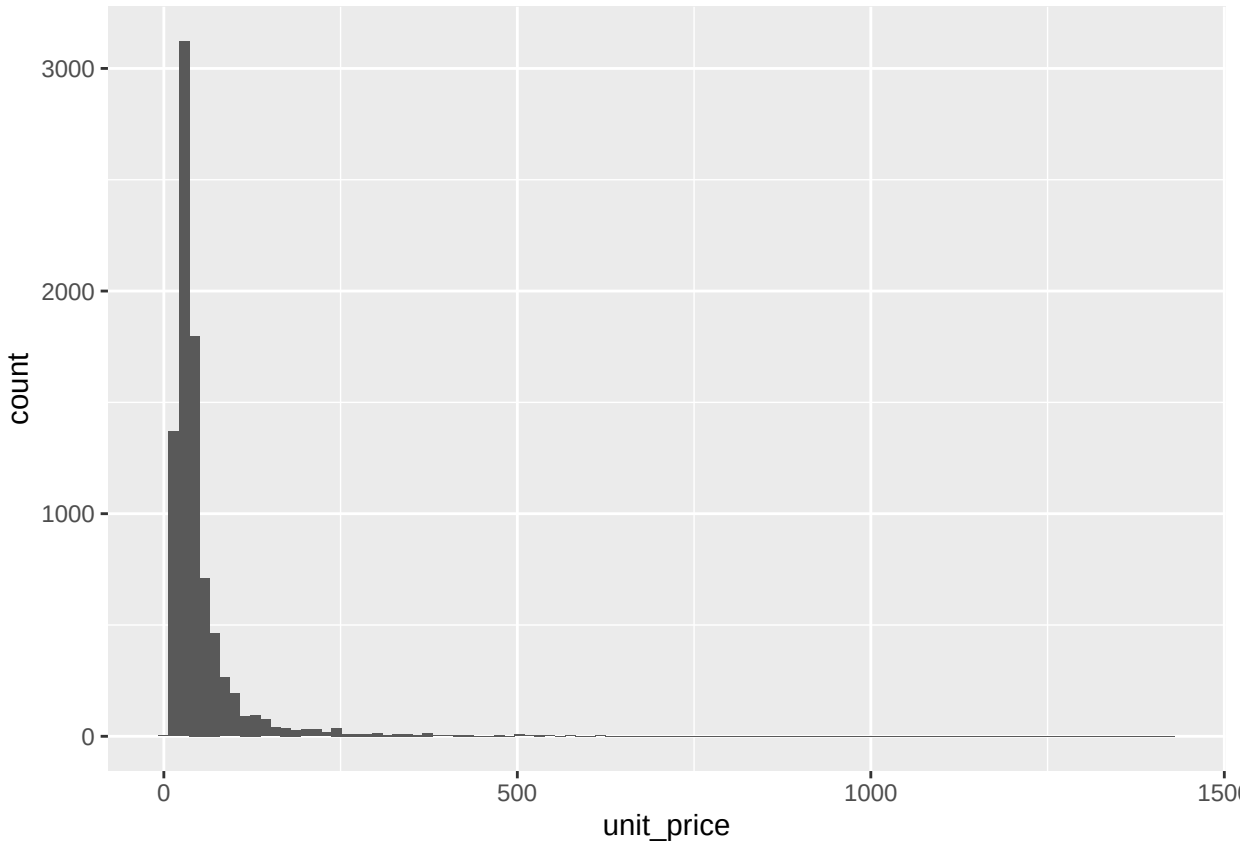
Summary of Attributes After Initial Analysis convert price into unit price (price/room_type)

```
# 4, 2, 1, 1
# final_data = final_data %>%
#   mutate(unit_price = ifelse(room_type=='Entire home/apt',price/4,ifelse(room_type=='Hotel room',pric
# 4, 2, 2, 1
final_data = final_data %>%
  mutate(unit_price = ifelse(room_type=='Entire home/apt',price/4,ifelse(room_type=='Shared room',price
```

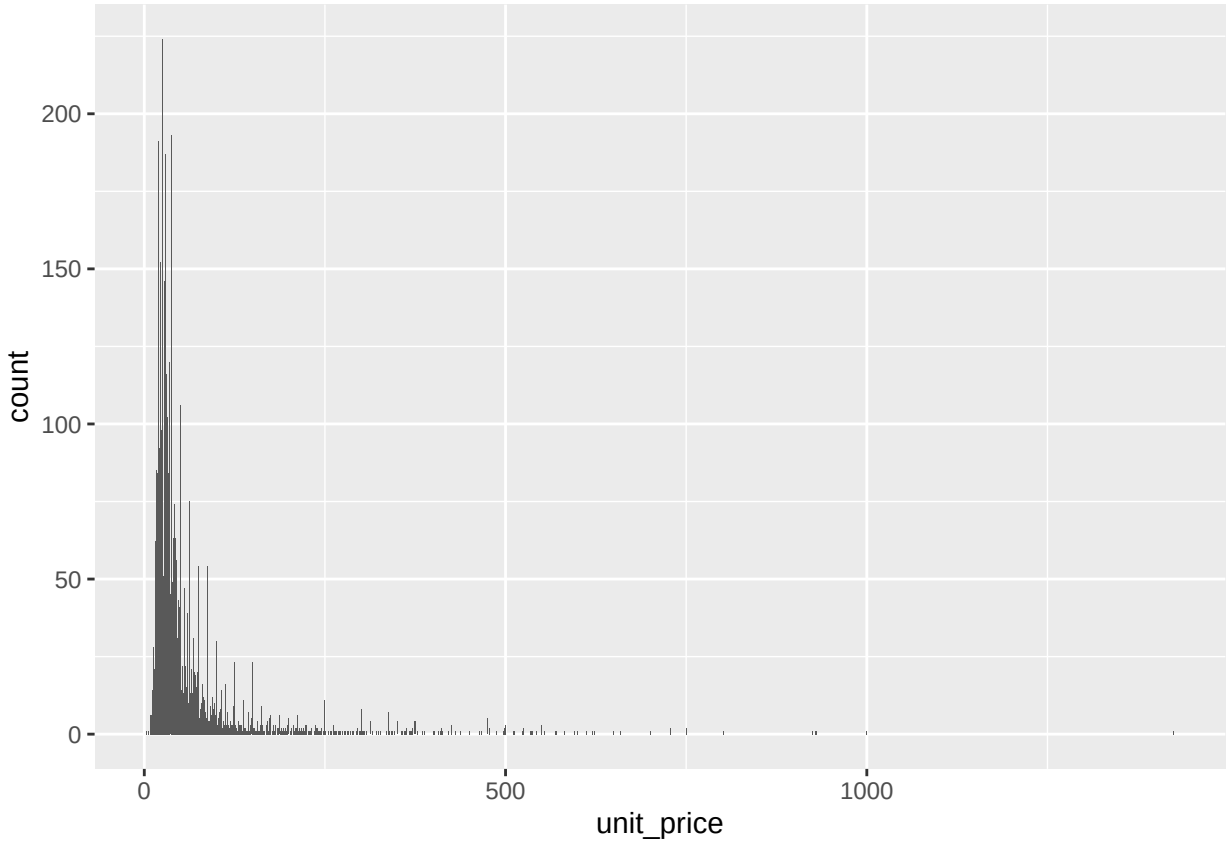
plot unit price: EDA - Bivariate Analysis (Measures)

```
####### plot all #########

ggplot(final_data) + geom_histogram(aes(x=unit_price), bins=100)
```




```
ggplot(final_data) + geom_bar(aes(x=unit_price))
```




```
####### mean min max SD #########
numeric_data = final_data %>% select(unit_price, avg_rating, crime_level, crime_incidents_count)
summarize_numeric(numeric_data)
```

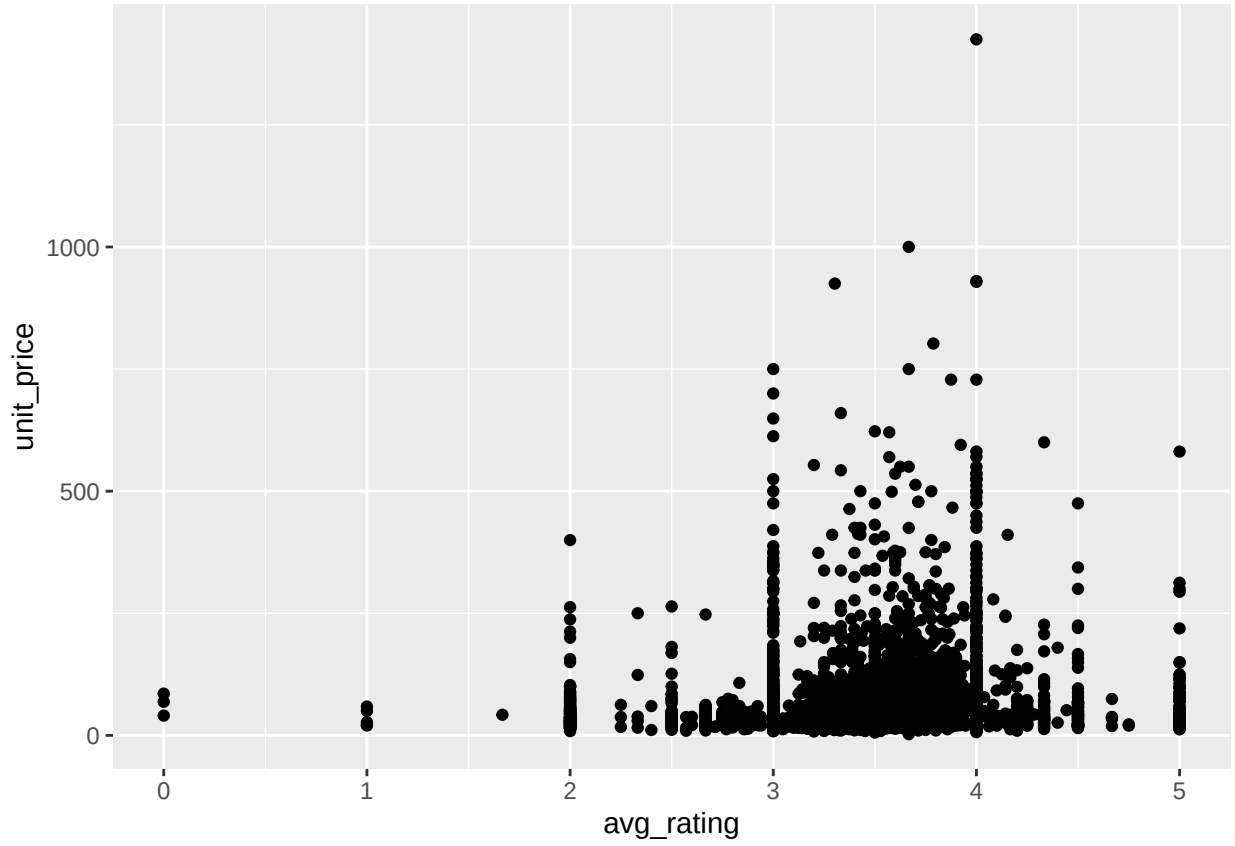
```
##               Attribute Missing Values Unique Values       Mean Min  Max
## 1            unit_price              0           716  52.902781 2.5 1425
## 2            avg_rating              0           471   3.523852 0.0    5
## 3           crime_level              0           664   2.224057 1.0    4
## 4 crime_incidents_count              0           344 302.859492 1.0 1790
##            SD
## 1  66.7841040
## 2   0.4638161
## 3   0.1982071
## 4 347.3810503
```

Bivariate Analysis - Correlation matrix

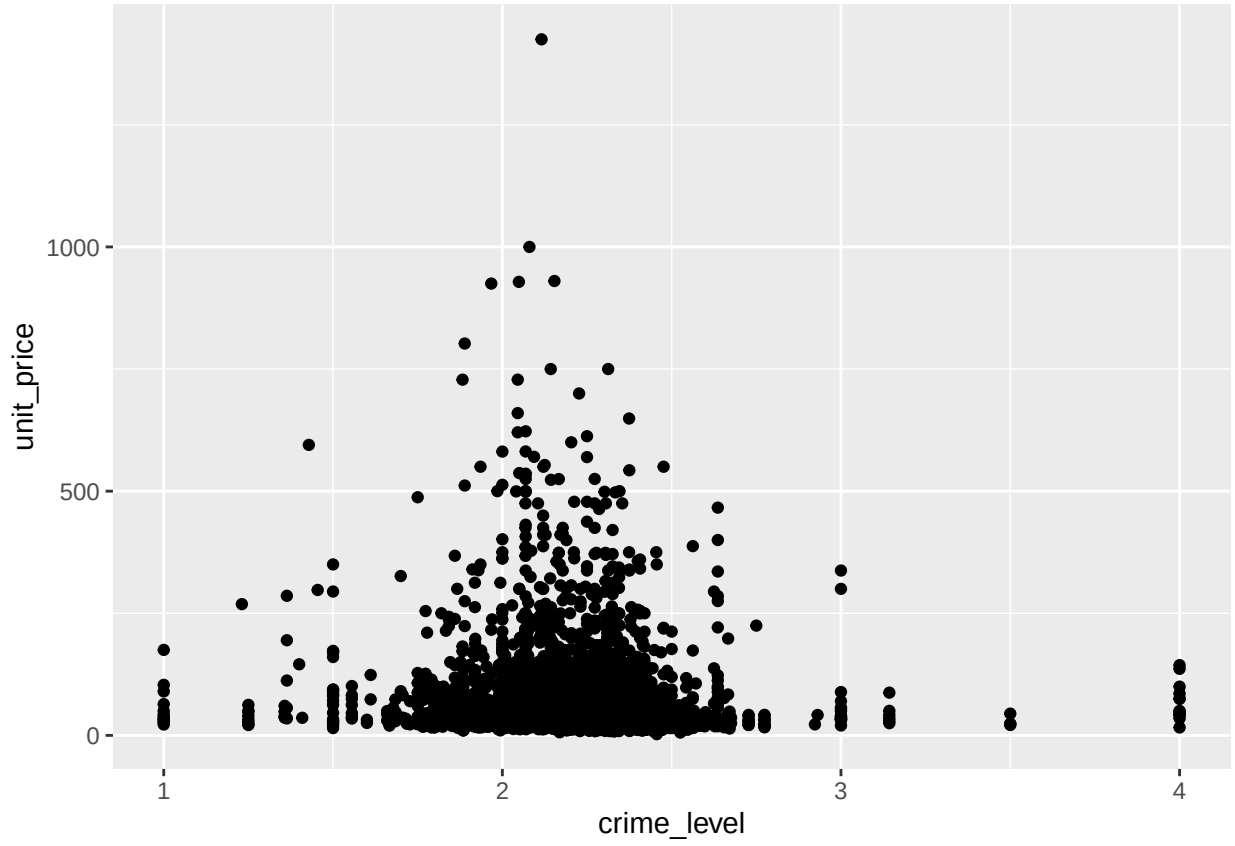
```
cor_data = select(final_data, c(unit_price, avg_rating, crime_level, crime_incidents_count))
fullCorrMatrix = round(cor(cor_data %>% select_if(is.numeric)), 2)
ggcorrplot(fullCorrMatrix, lab = TRUE)
```

EDA - Bivariate Analysis (Measures)

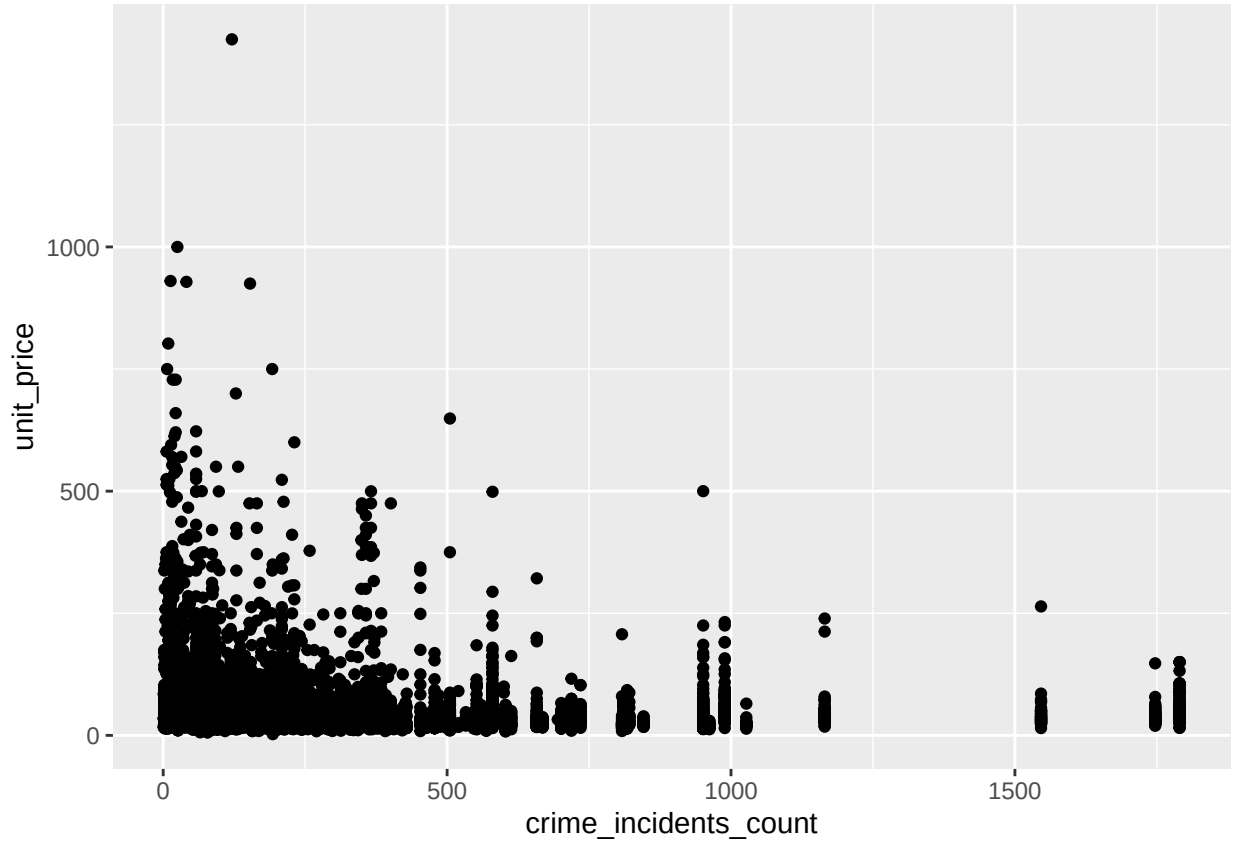
```
# unit_price vs avg_rating
ggplot(final_data) + geom_point(aes(x=avg_rating, y=unit_price))
```



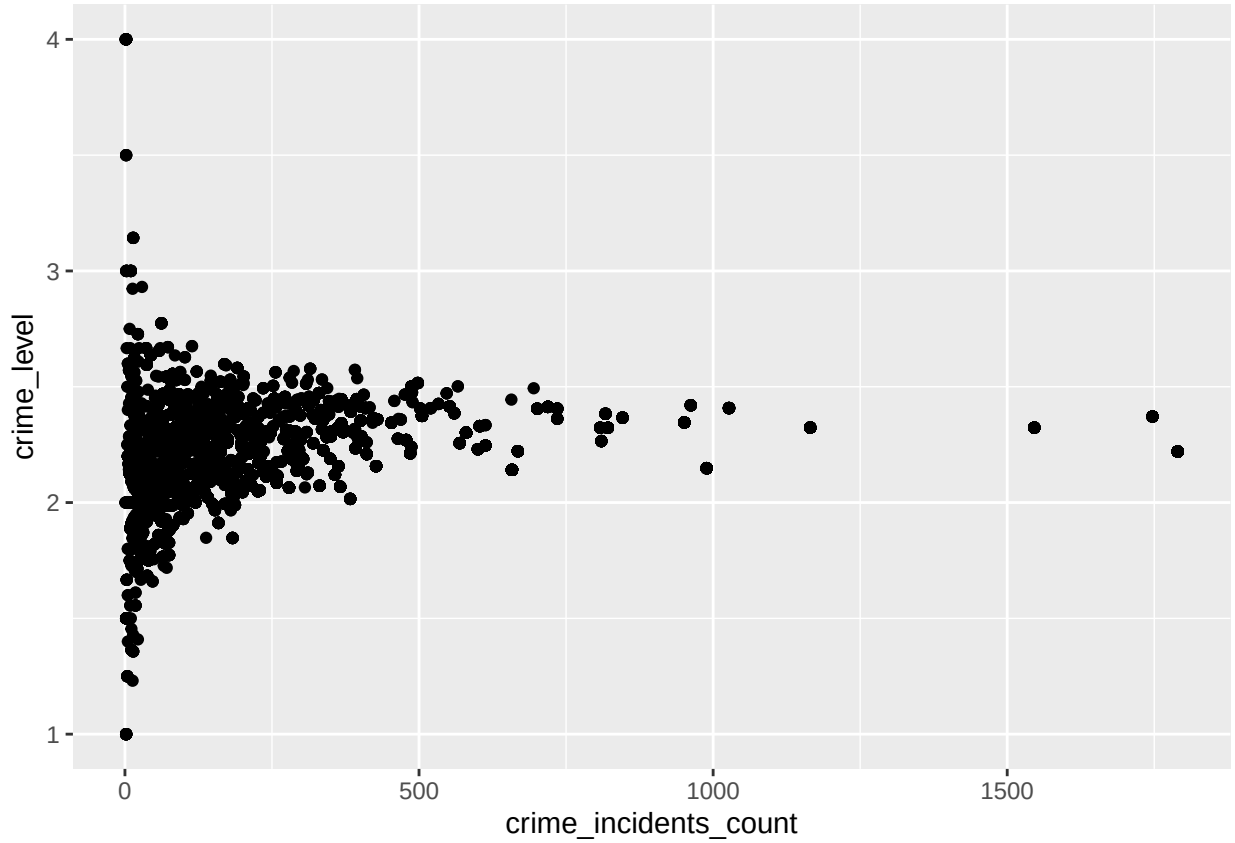

```
# unit_price vs crime_level
ggplot(final_data) + geom_point(aes(x=crime_level, y=unit_price))
```




```
# unit_price vs crime_incidents_count
ggplot(final_data) + geom_point(aes(x=crime_incidents_count, y=unit_price))
```

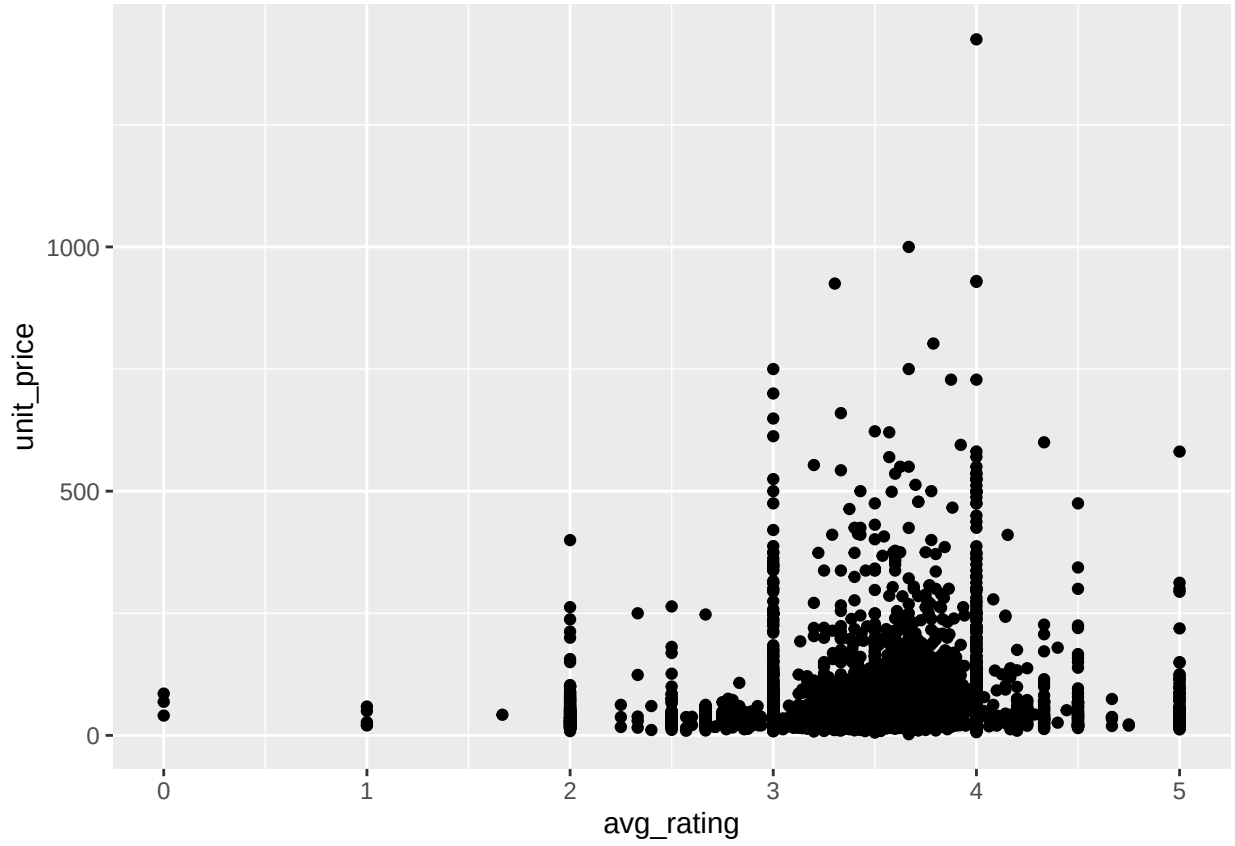



```
# crime_incidents_count vs crime_level
ggplot(final_data) + geom_point(aes(x=crime_incidents_count, y=crime_level))
```



EDA - Bivariate Analysis (Explore the data points with high rating but low price)

```
# unit_price vs avg_rating
ggplot(final_data) + geom_point(aes(x=avg_rating, y=unit_price))
```

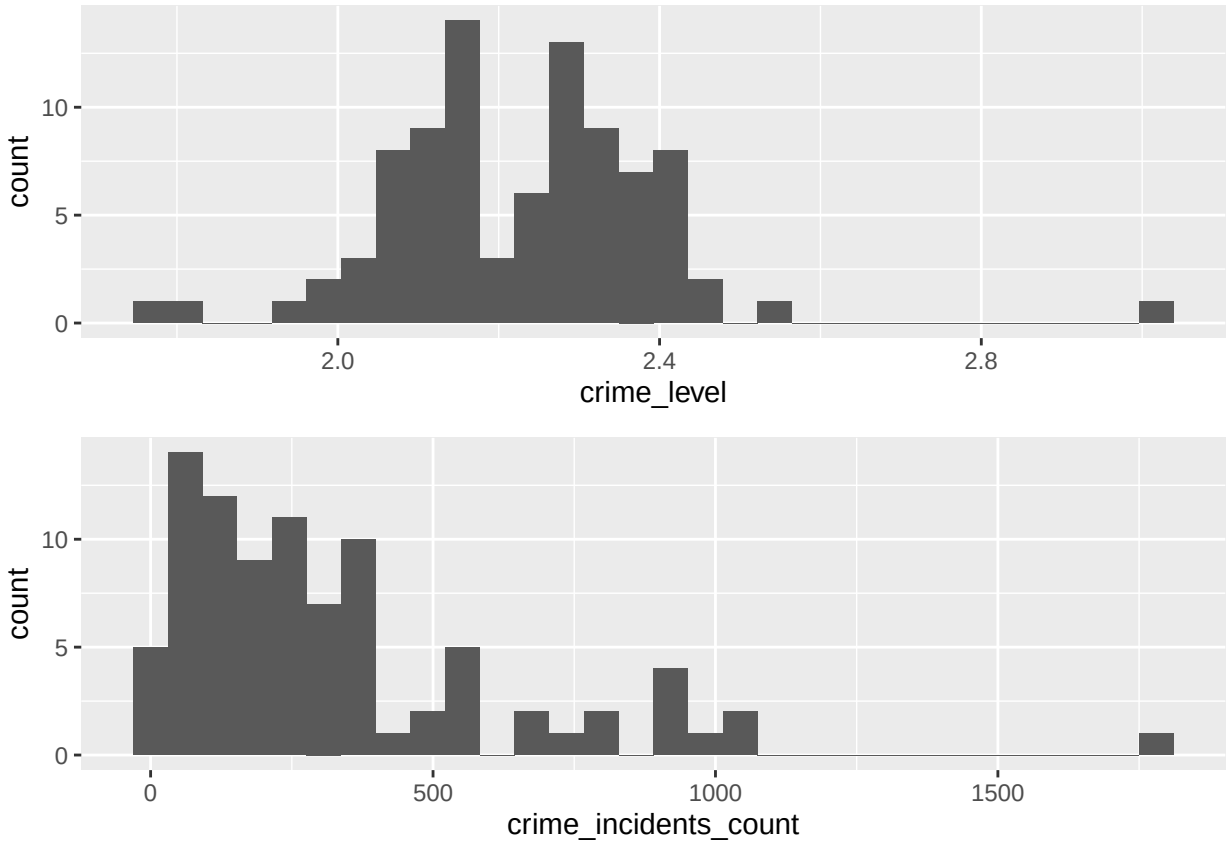



```
# filter out low price and high rating
LH_data = final_data %>%
  filter(unit_price<=100) %>%
  filter(avg_rating>=5)
summarize_numeric(LH_data)
```

```
##               Attribute Missing Values Unique Values            Mean       Min
## 1                    id              0             89 28073806.730337 131557.00
## 2              latitude              0             28       34.073258     33.81
## 3             longitude              0             32     -118.370899   -118.63
## 4                 price              0             65      140.404494     21.00
## 5      availability_365              0             62      242.910112      7.00
## 6            avg_rating              0              1        5.000000      5.00
## 7          crime_level              0             74        2.231017      1.75
## 8 crime_incidents_count              0             72      325.898876     10.00
## 9            unit_price              0             64       39.957865     12.25
##           Max                SD
## 1 45642741.00 15702519.69091159
## 2       34.26        0.07194972
## 3     -118.18        0.09026024
## 4      399.00       94.24963457
## 5      365.00      119.30534169
## 6        5.00        0.00000000
## 7        3.00        0.16848303
## 8     1790.00      313.03274752
## 9       99.75       22.67511514
```

```
##### visulaization #####
LH1 = ggplot(LH_data) + geom_histogram(aes(x=crime_level))
LH2 = ggplot(LH_data) + geom_histogram(aes(x=crime_incidents_count))
ggarrange(LH1, LH2, nrow=2)
```

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

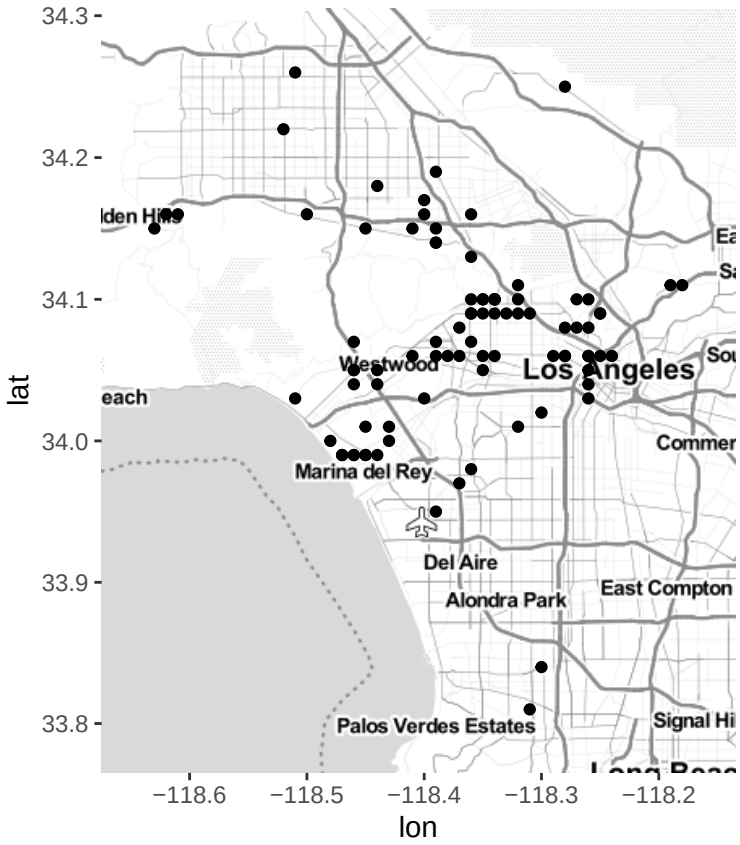



```
height <- max(LH_data$latitude) - min(LH_data$latitude)
width <- max(LH_data$longitude) - min(LH_data$longitude)
LA_borders <- c(bottom  = min(LH_data$latitude)  - 0.1 * height,
                top     = max(LH_data$latitude)  + 0.1 * height,
                left    = min(LH_data$longitude) - 0.1 * width,
                right   = max(LH_data$longitude) + 0.1 * width)

map <- get_stamenmap(LA_borders, zoom = 10, maptype = "toner-lite")
```

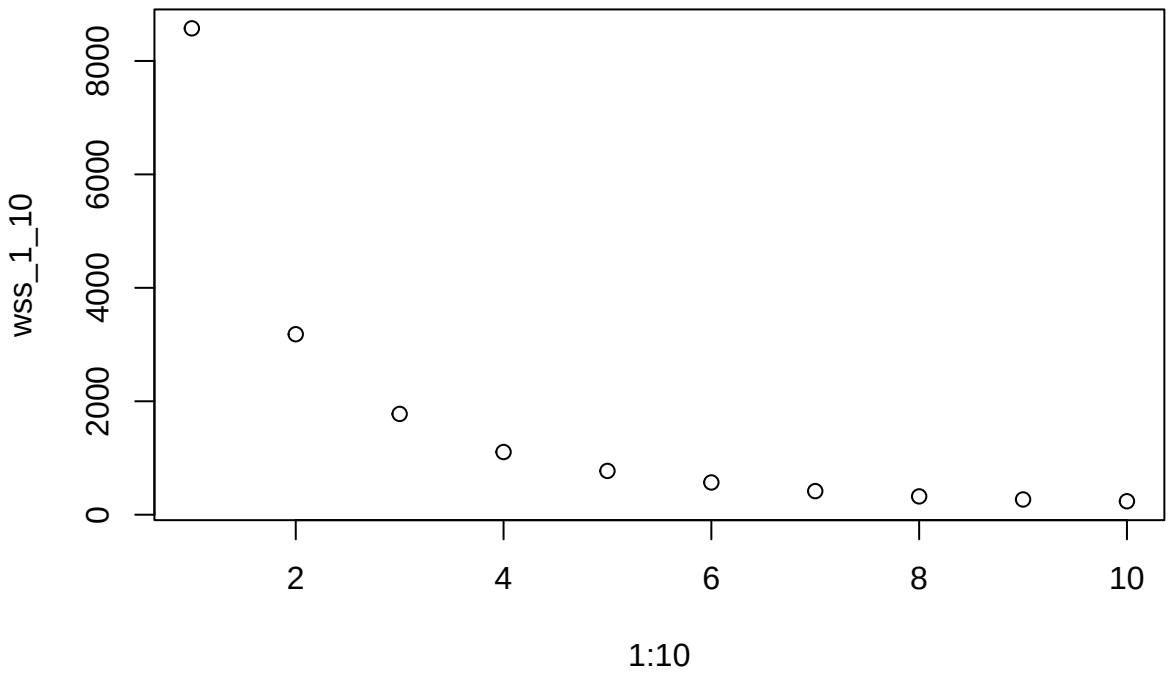
```
## i Map tiles by Stamen Design, under CC BY 3.0. Data by OpenStreetMap, under ODbL.
```

```
ggmap(map) +
    geom_point(data = LH_data, mapping = aes(x = longitude, y = latitude))
```

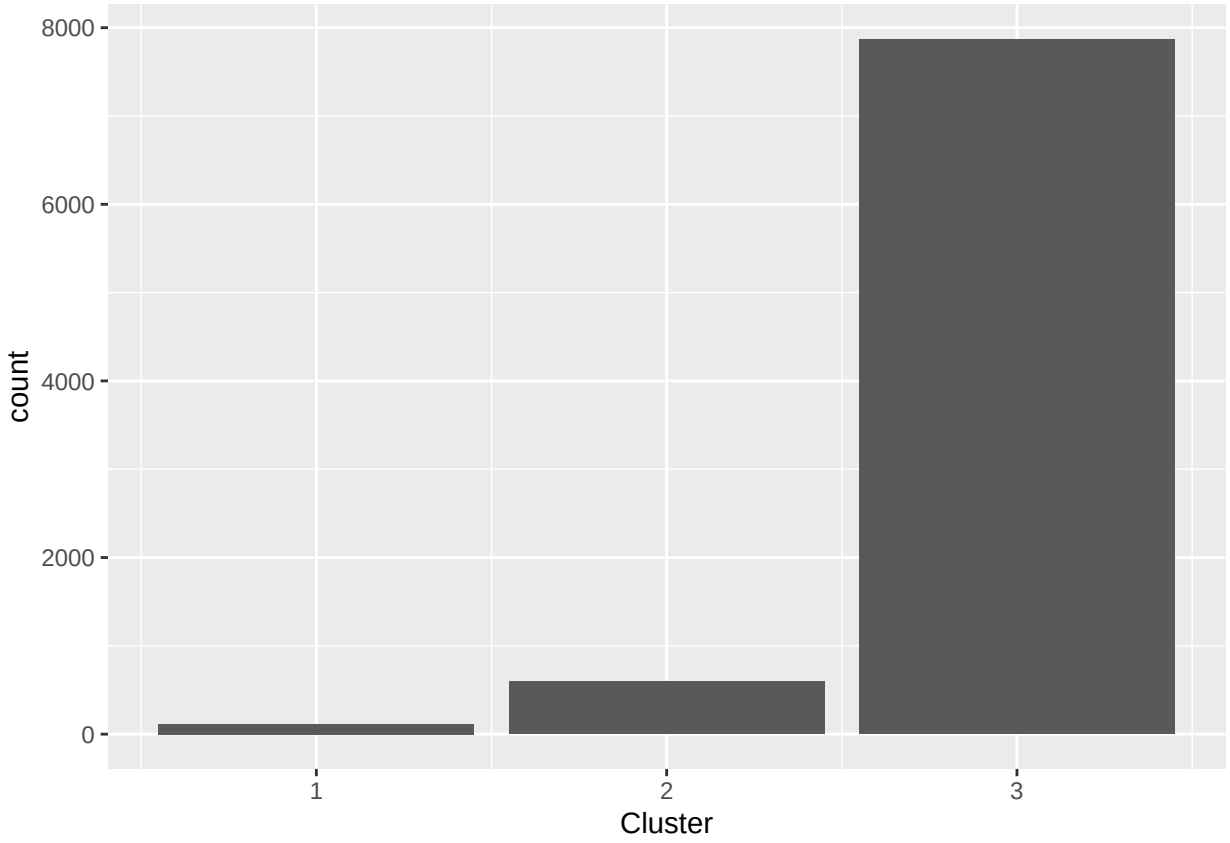


Do clustering based on unit_price

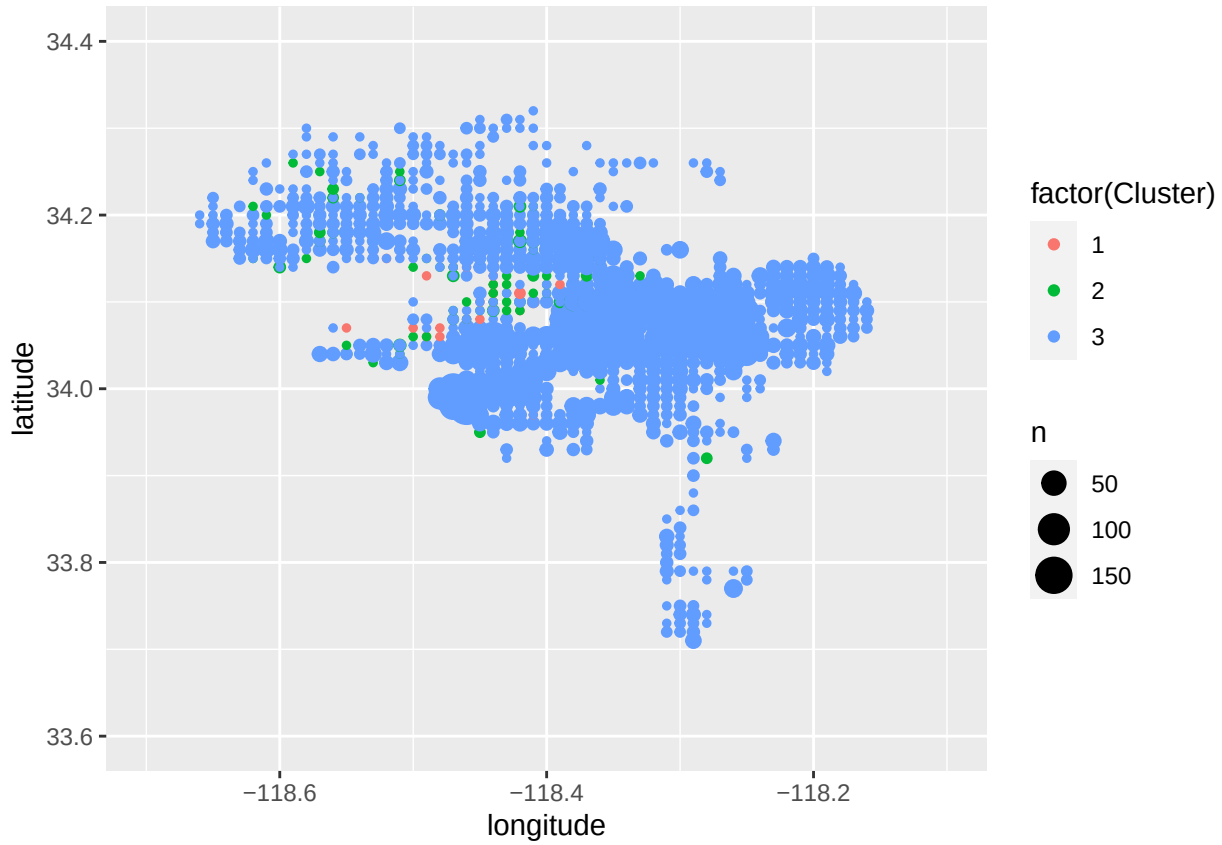
```
train_data = select(final_data, unit_price)
##### elbow diagram #####
wss = function (k) {kmeans(scale(train_data), k, nstart=10)$tot.withinss}
wss_1_10 = map_dbl(1:10, wss) # with k from 1 to 10, means call wss function 10 times to do kmeans
plot(1:10, wss_1_10)
```



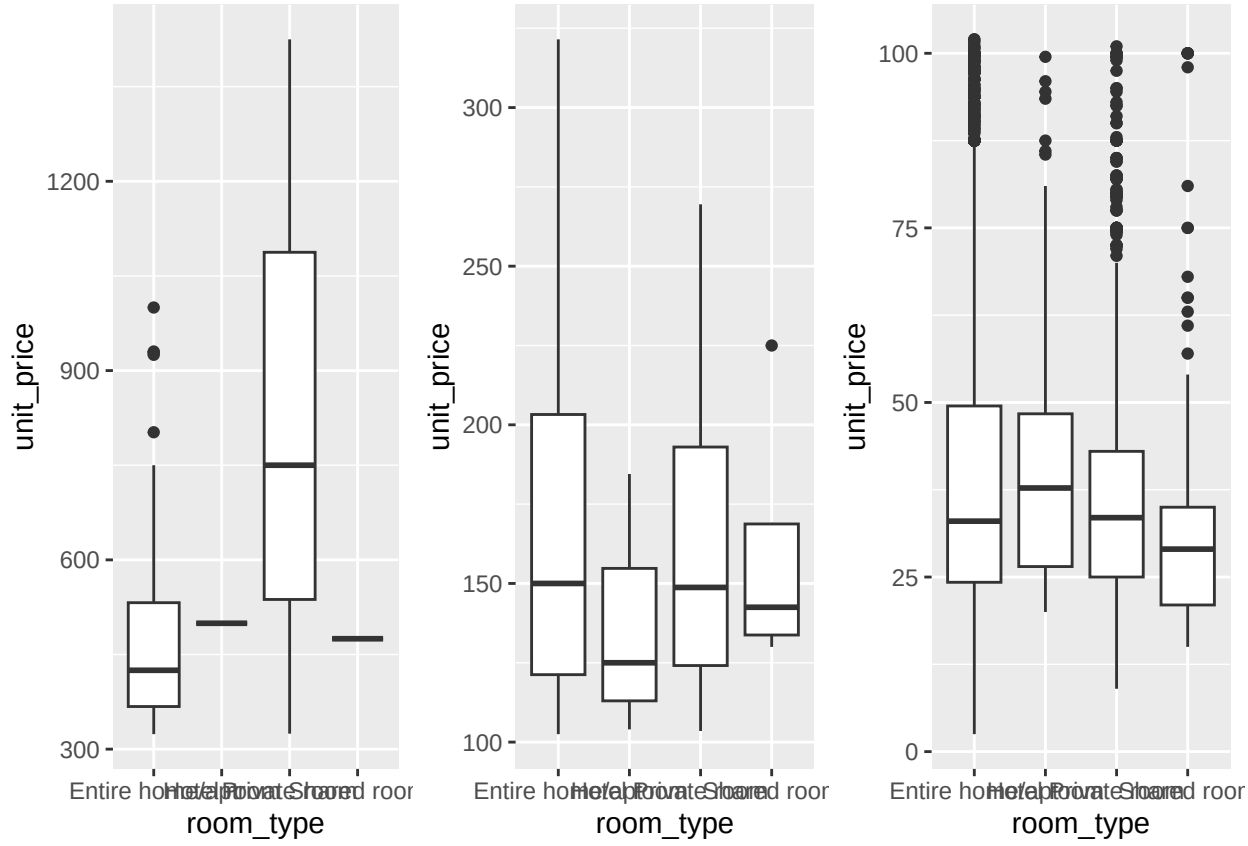

```
##### kmeans #####
res = kmeans(scale(train_data), 3, nstart=25)
final_data_price_cluster = final_data %>% mutate(Cluster = res$cluster)
ggplot(final_data_price_cluster) + geom_bar(aes(x=Cluster))
```




```
ggplot(final_data_price_cluster) + geom_count(aes(x=longitude, y=latitude, color=factor(Cluster))) + yl
```




```
########
# cluster1
cluster1_data = final_data_price_cluster %>%
  filter(Cluster==1)
c1 = ggplot(cluster1_data) + geom_boxplot(aes(x=room_type, y=unit_price))
# cluster2
cluster2_data = final_data_price_cluster %>%
  filter(Cluster==2)
c2 = ggplot(cluster2_data) + geom_boxplot(aes(x=room_type, y=unit_price))
# cluster3
cluster3_data = final_data_price_cluster %>%
  filter(Cluster==3)
c3 = ggplot(cluster3_data) + geom_boxplot(aes(x=room_type, y=unit_price))
ggarrange(c1, c2, c3, nrow=1)
```

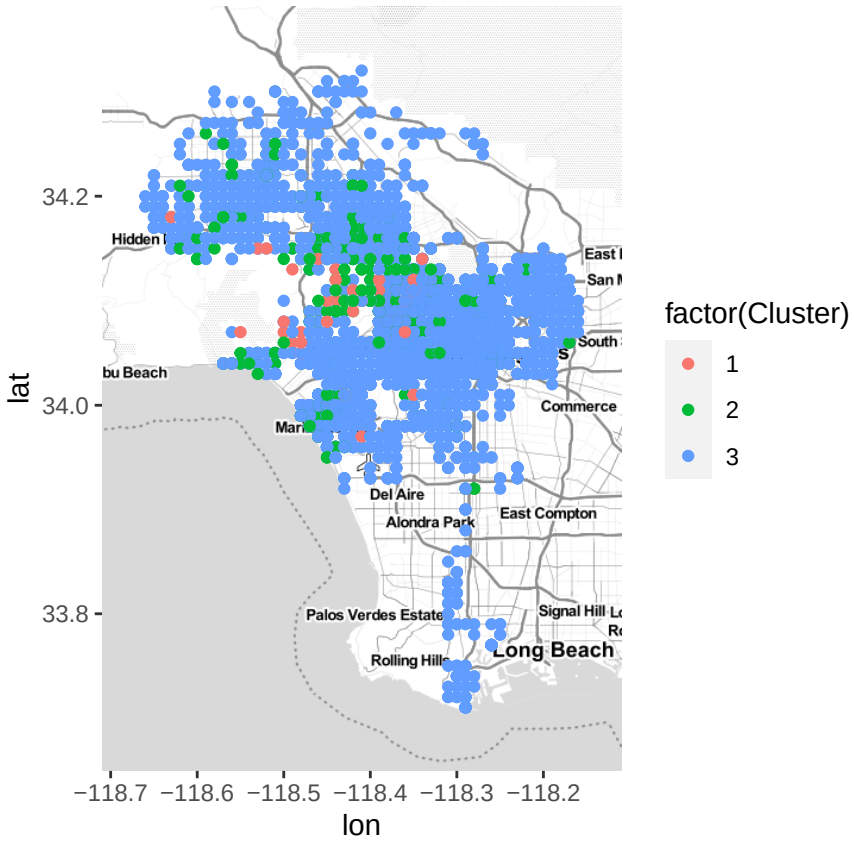



```
#############
library(ggmap)
height <- max(final_data_price_cluster$latitude) - min(final_data_price_cluster$latitude)
width <- max(final_data_price_cluster$longitude) - min(final_data_price_cluster$longitude)
LA_borders <- c(bottom  = min(final_data_price_cluster$latitude)  - 0.1 * height,
                top     = max(final_data_price_cluster$latitude)  + 0.1 * height,
                left    = min(final_data_price_cluster$longitude) - 0.1 * width,
                right   = max(final_data_price_cluster$longitude) + 0.1 * width)

map <- get_stamenmap(LA_borders, zoom = 10, maptype = "toner-lite")
```

```
## i Map tiles by Stamen Design, under CC BY 3.0. Data by OpenStreetMap, under ODbL.
```

```
ggmap(map) +
    geom_point(data = final_data_price_cluster, mapping = aes(x = longitude, y = latitude, color=factor
```

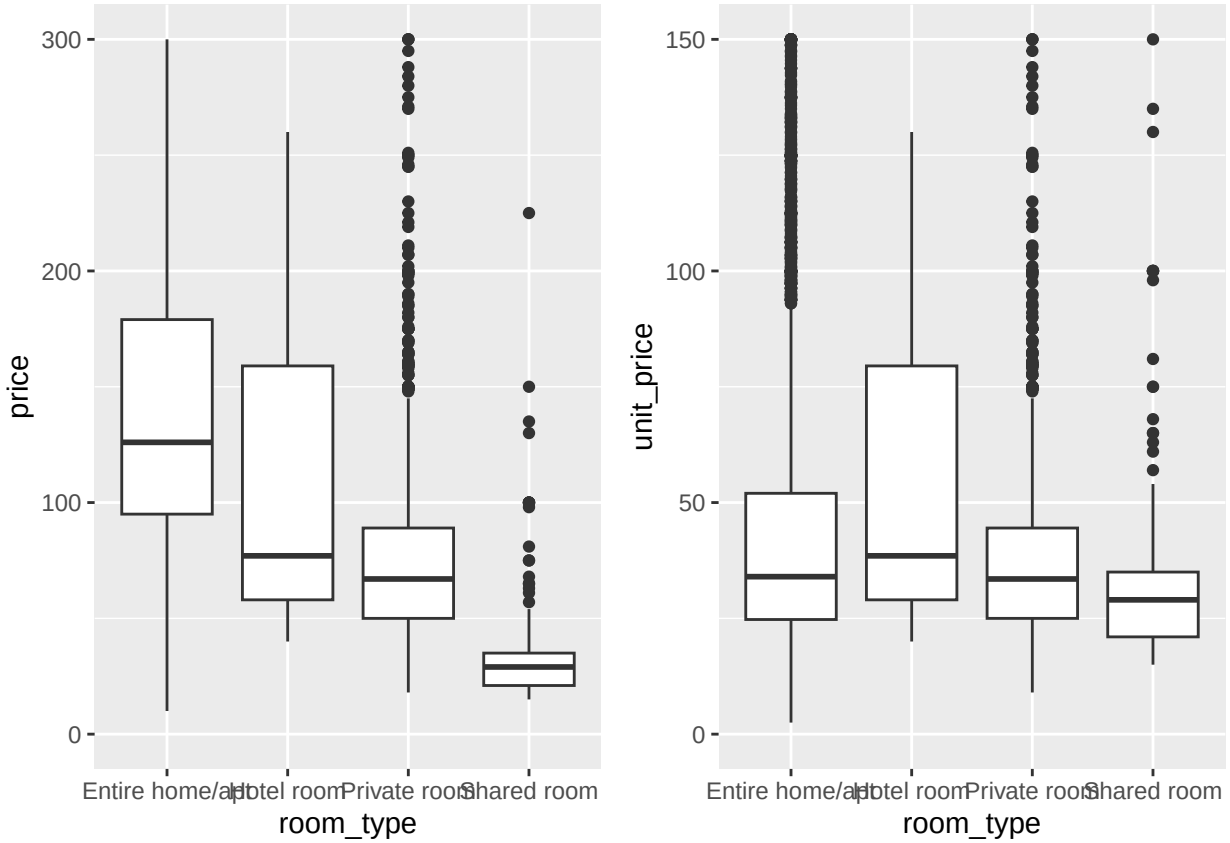


price vs unit_price

```
rp1 = ggplot(final_data) + geom_boxplot(aes(x=room_type, y=price)) + ylim(0, 300)
rup1 = ggplot(final_data) + geom_boxplot(aes(x=room_type, y=unit_price)) + ylim(0, 150)
ggarrange(rp1, rup1, nrow=1)
```

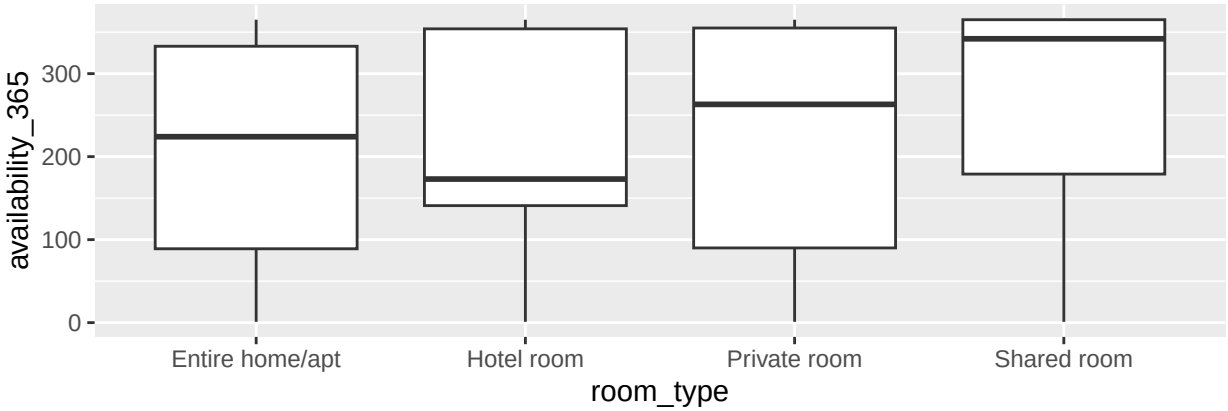
```
## Warning: Removed 1068 rows containing non-finite values (‘stat_boxplot()‘).
```

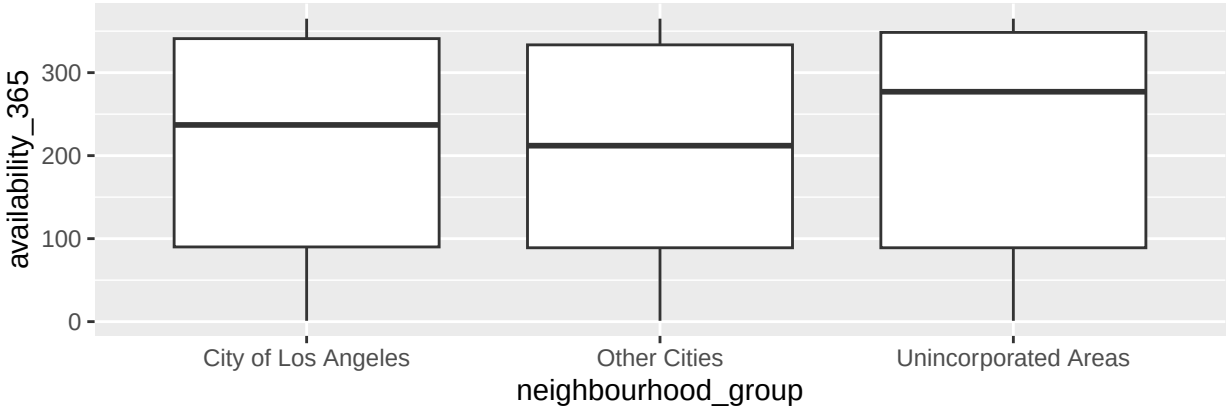
```
## Warning: Removed 389 rows containing non-finite values (‘stat_boxplot()‘).
```



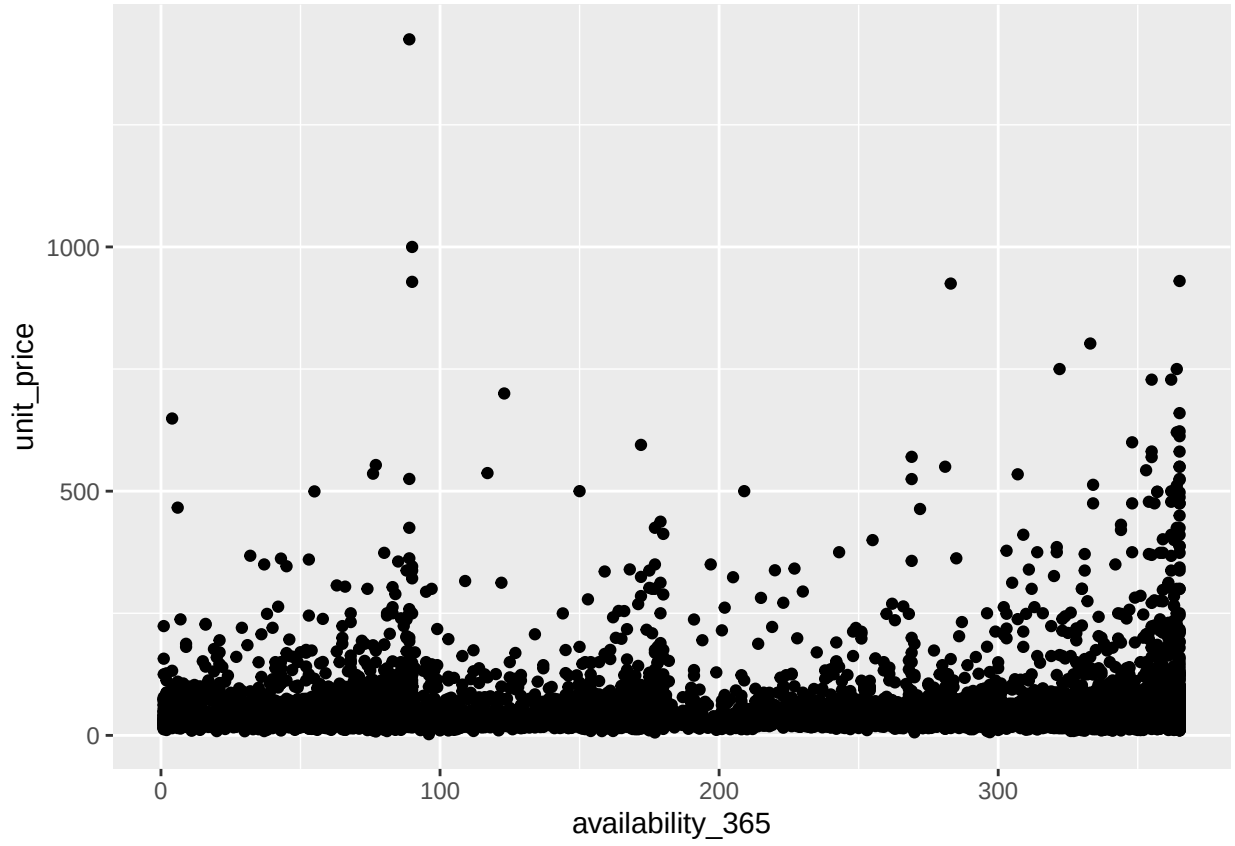
EDA - availability_365

```
nga1 = ggplot(final_data) + geom_boxplot(aes(x=neighbourhood_group, y=availability_365))
ra1 = ggplot(final_data) + geom_boxplot(aes(x=room_type, y=availability_365))
na1 = ggplot(final_data) + geom_boxplot(aes(x=neighbourhood, y=availability_365))
ggarrange(ra1, nga1, nrow=2)
```

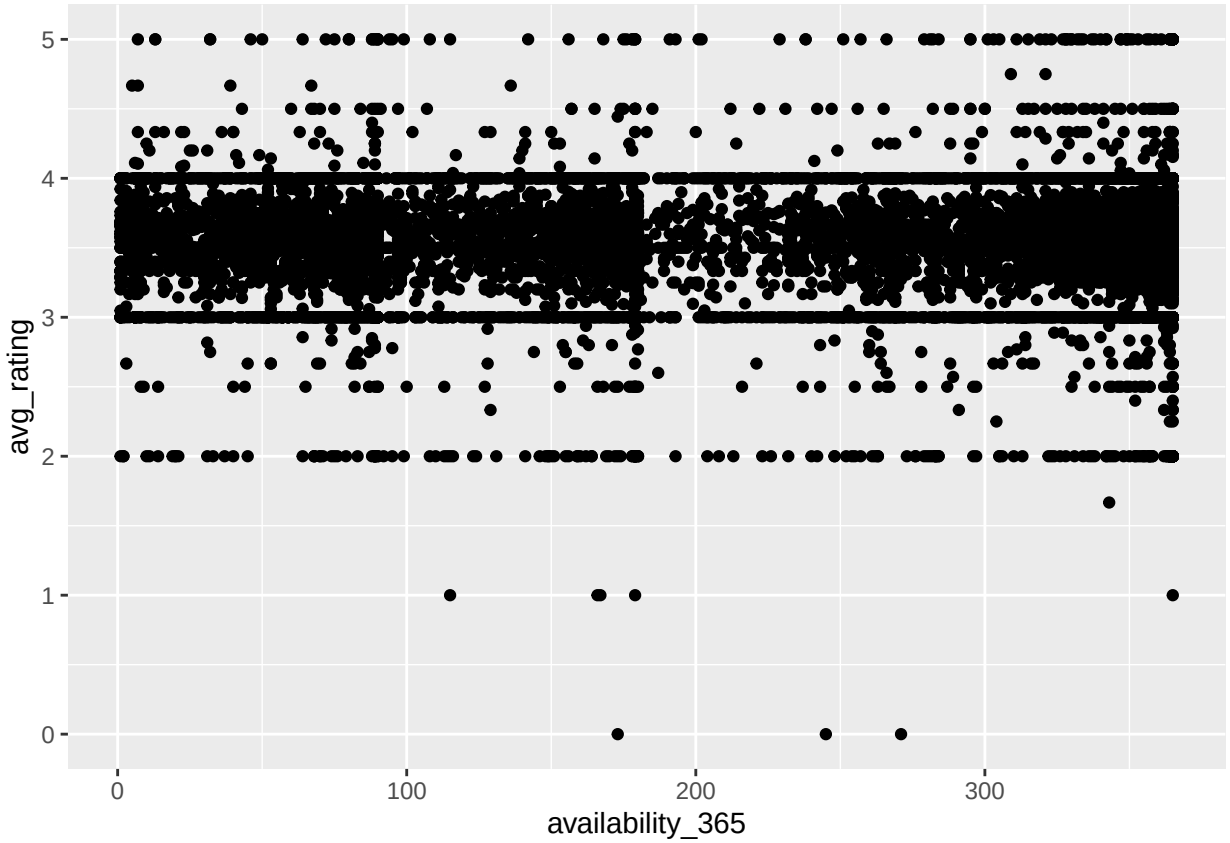




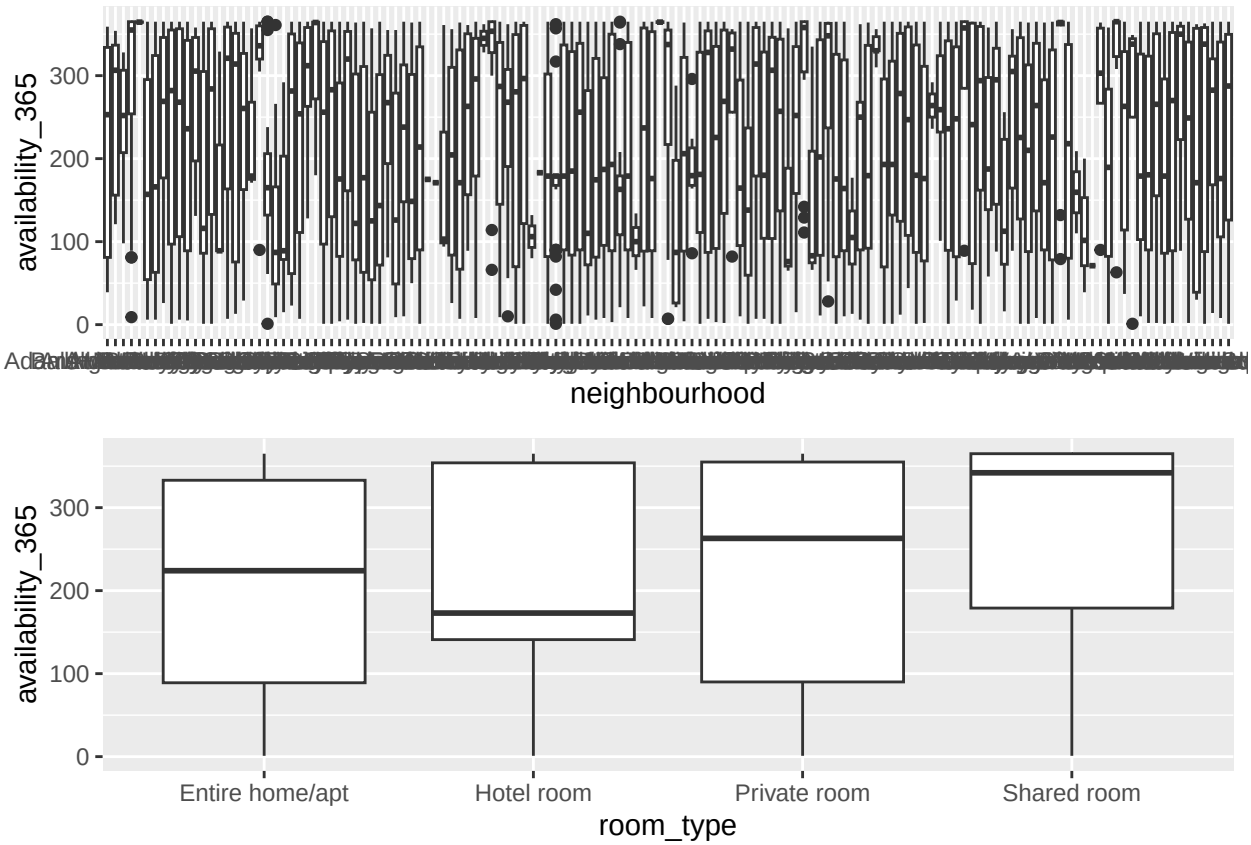

```
ggplot(final_data) + geom_point(aes(x=availability_365, y=unit_price))
```




```
ggplot(final_data) + geom_point(aes(x=availability_365, y=avg_rating))
```



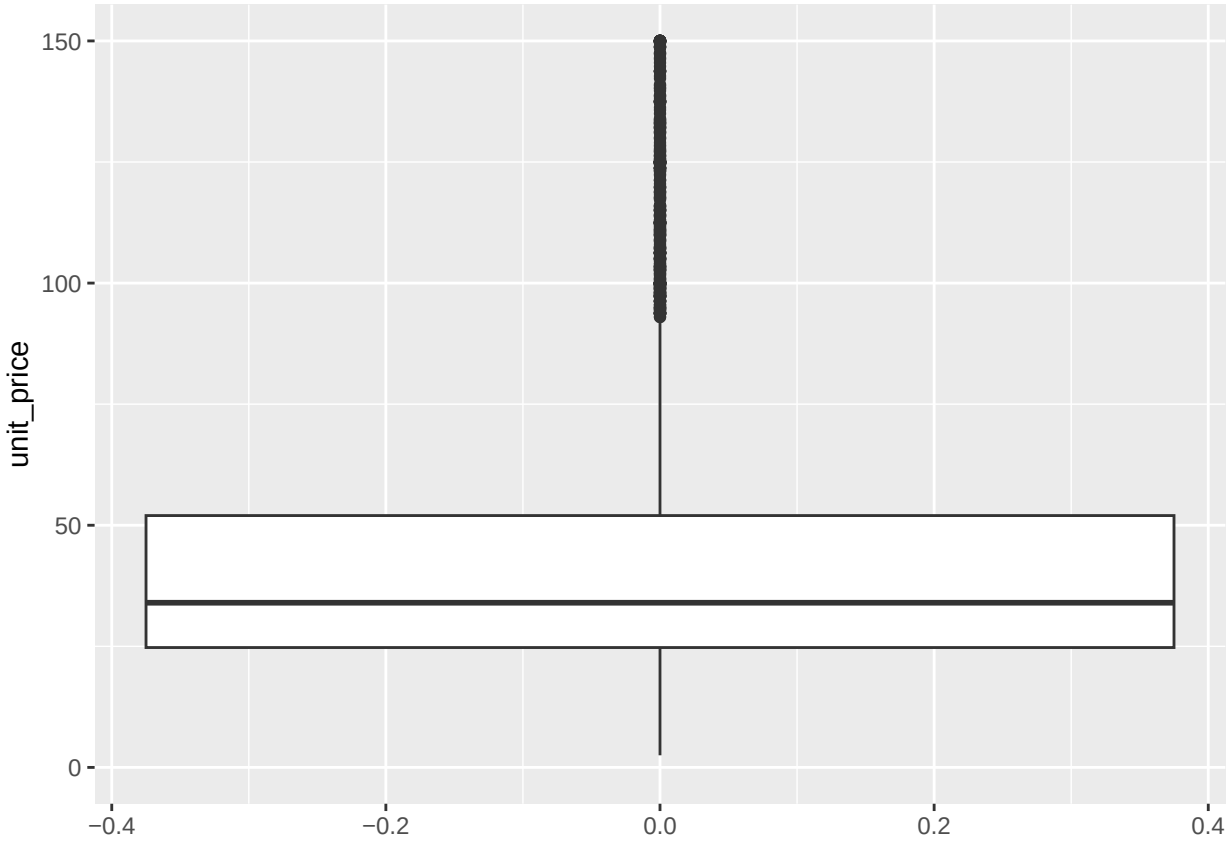

```
ggarrange(na1, ra1, nrow=2)
```



EDA - Bivariate Analysis (unit_price)

```
#####
Entire_data = final_data %>%
  filter(room_type=='Entire home/apt')
ggplot(Entire_data) + geom_boxplot(aes(y=unit_price)) + ylim(0, 150)
```

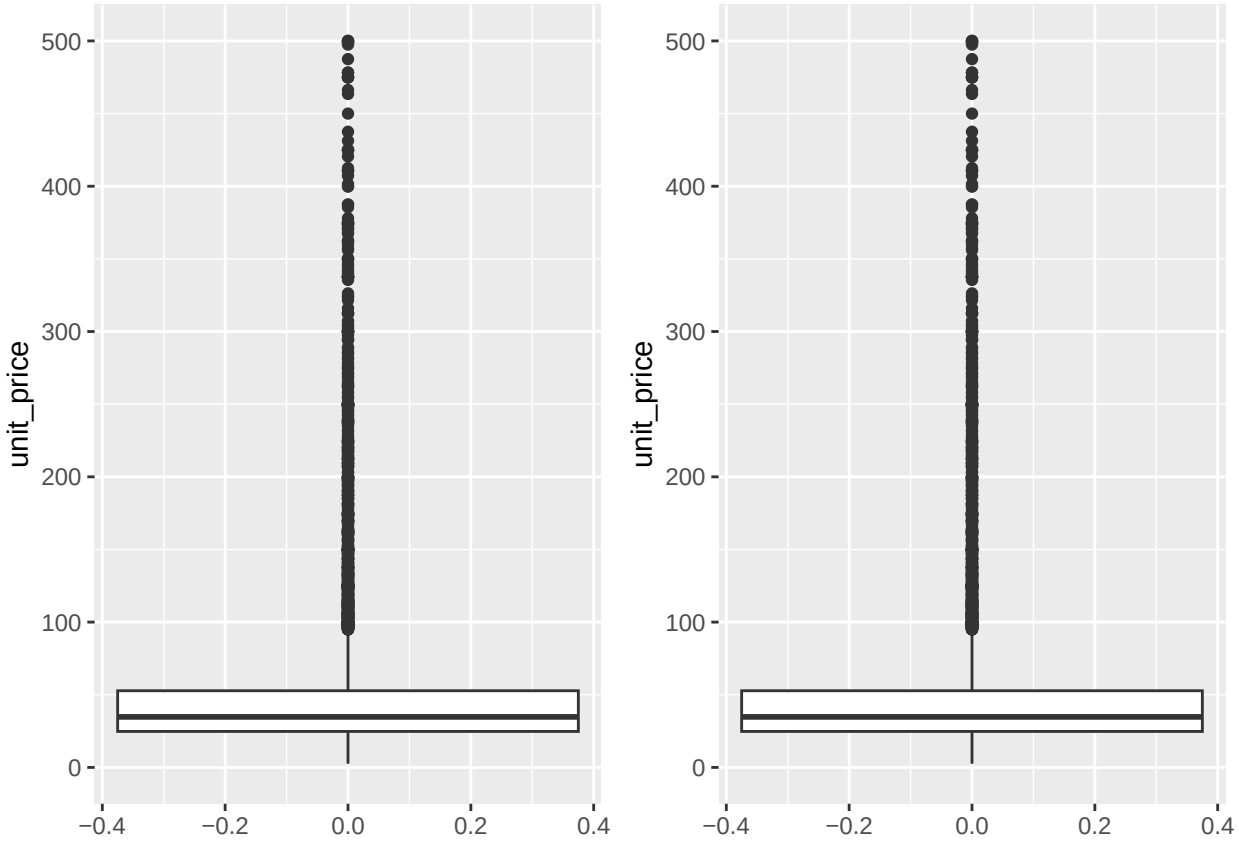
```
## Warning: Removed 362 rows containing non-finite values (‘stat_boxplot()‘).
```




```
####
b1 = ggplot(final_data) + geom_boxplot(aes(y=unit_price)) + ylim(0, 500)
ggarrange(b1, b1, nrow=1)
```

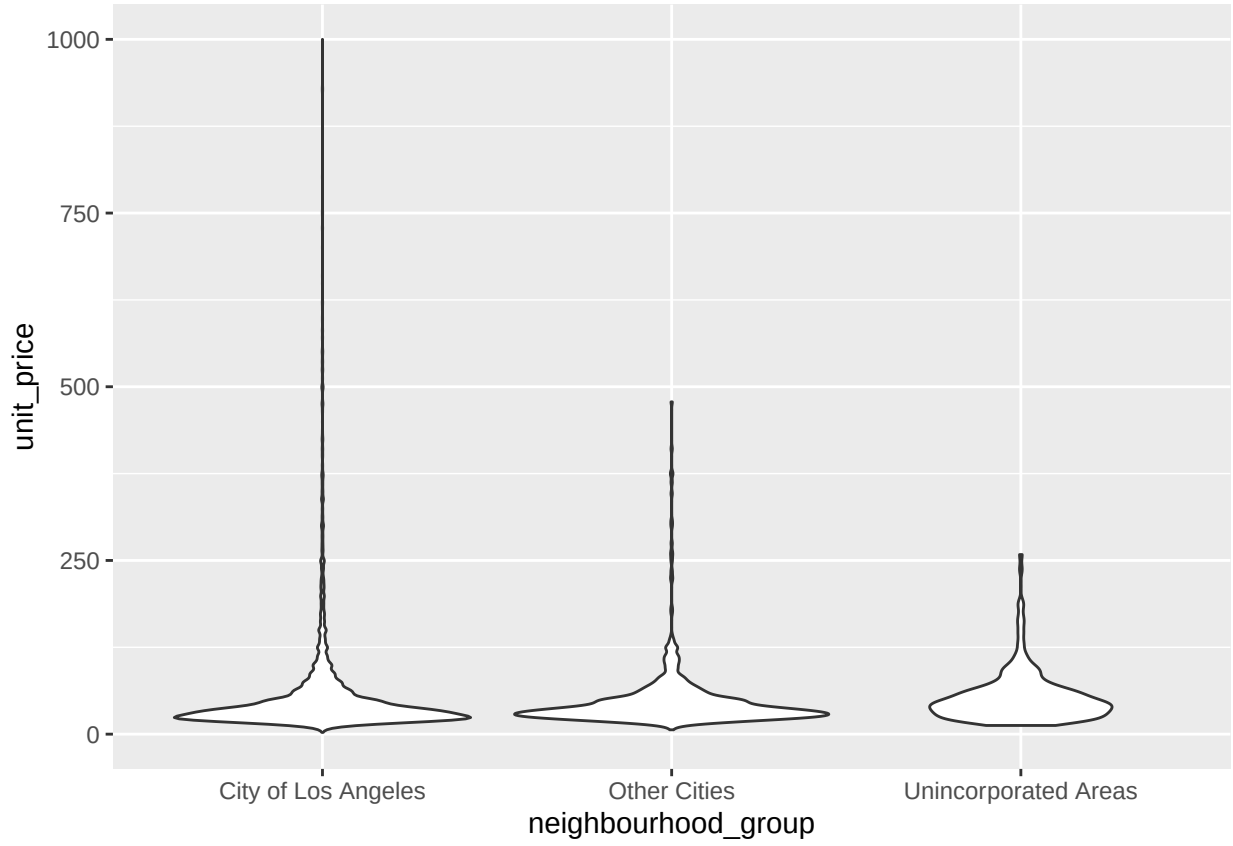
```
## Warning: Removed 36 rows containing non-finite values (‘stat_boxplot()‘).

## Warning: Removed 36 rows containing non-finite values (‘stat_boxplot()‘).
```




```
# neighbourhood_group vs. unit_price
np1 = ggplot(final_data) + geom_boxplot(aes(x=neighbourhood_group, y=unit_price))
np2 = ggplot(final_data) + geom_boxplot(aes(x=neighbourhood_group, y=unit_price)) + ylim(0, 150)
ggplot(final_data) + geom_violin(aes(x=neighbourhood_group, y=unit_price)) + ylim(0, 1000)
```

```
## Warning: Removed 1 rows containing non-finite values (‘stat_ydensity()‘).
```



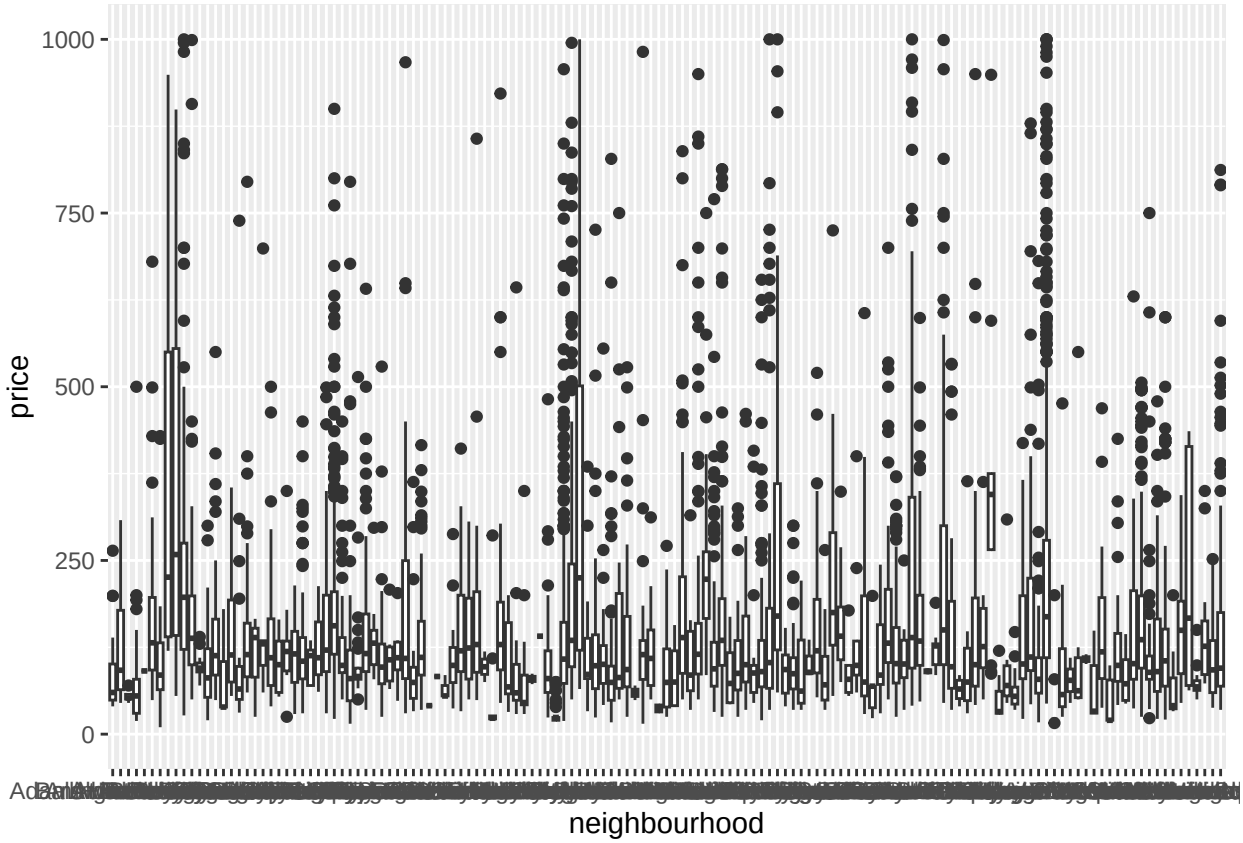

```
#plot(final_data$neighbourhood_group, join_tb$price, ylim=c(1, 300))

# neighbourhood vs. price
distinct(join_tb, neighbourhood)
```

```
## # A tibble: 141 x 1
##    neighbourhood
##    <chr>
##  1 Hollywood
##  2 Del Rey
##  3 Atwater Village
##  4 Venice
##  5 Gardena
##  6 Mid-City
##  7 Valley Village
##  8 Silver Lake
##  9 Hollywood Hills
## 10 Highland Park
## # i 131 more rows
```

```
ggplot(join_tb) + geom_boxplot(aes(x=neighbourhood, y=price)) + ylim(0, 1000)
```

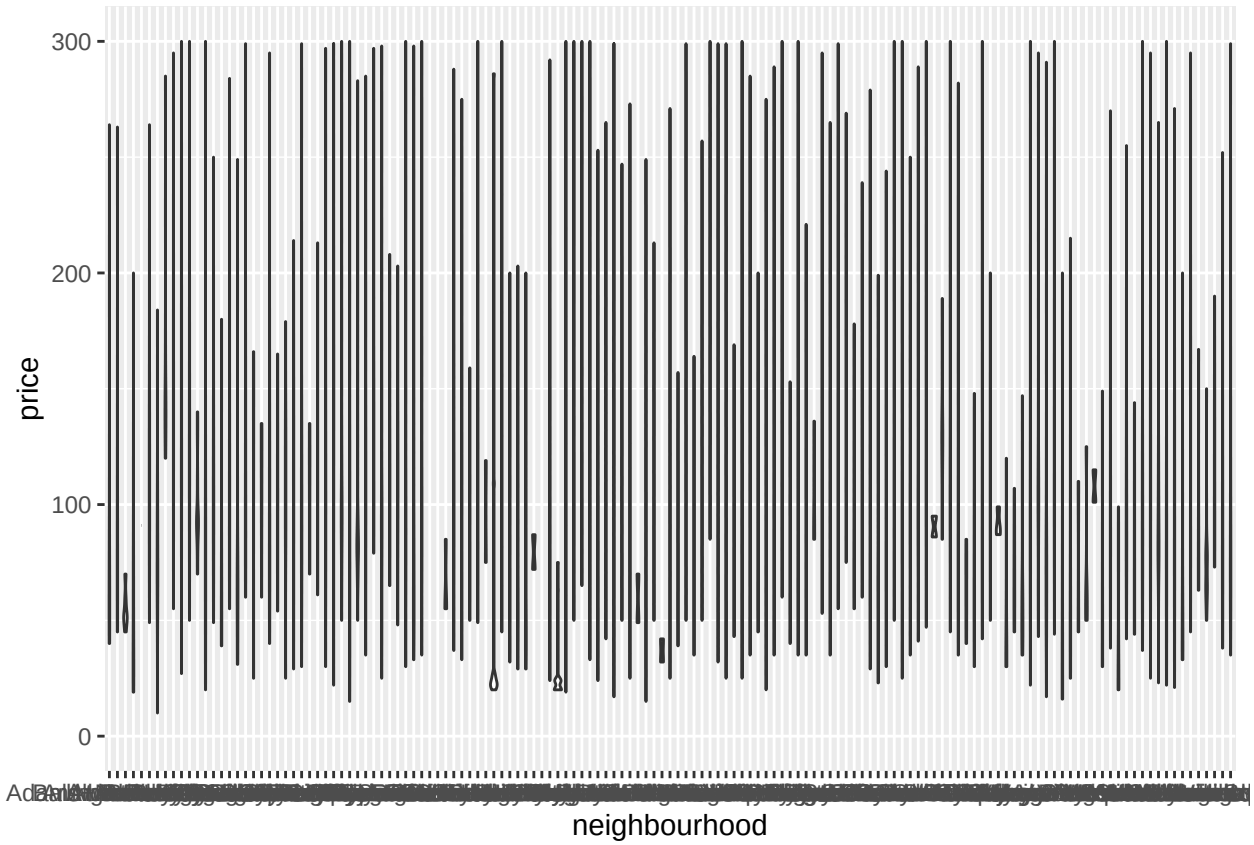
```
## Warning: Removed 164 rows containing non-finite values (`stat_boxplot()`).
```




```
ggplot(join_tb) + geom_violin(aes(x=neighbourhood, y=price)) + ylim(0, 300)
```

```
## Warning: Removed 1068 rows containing non-finite values (‘stat_ydensity()‘).
```

```
## Warning: Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
```



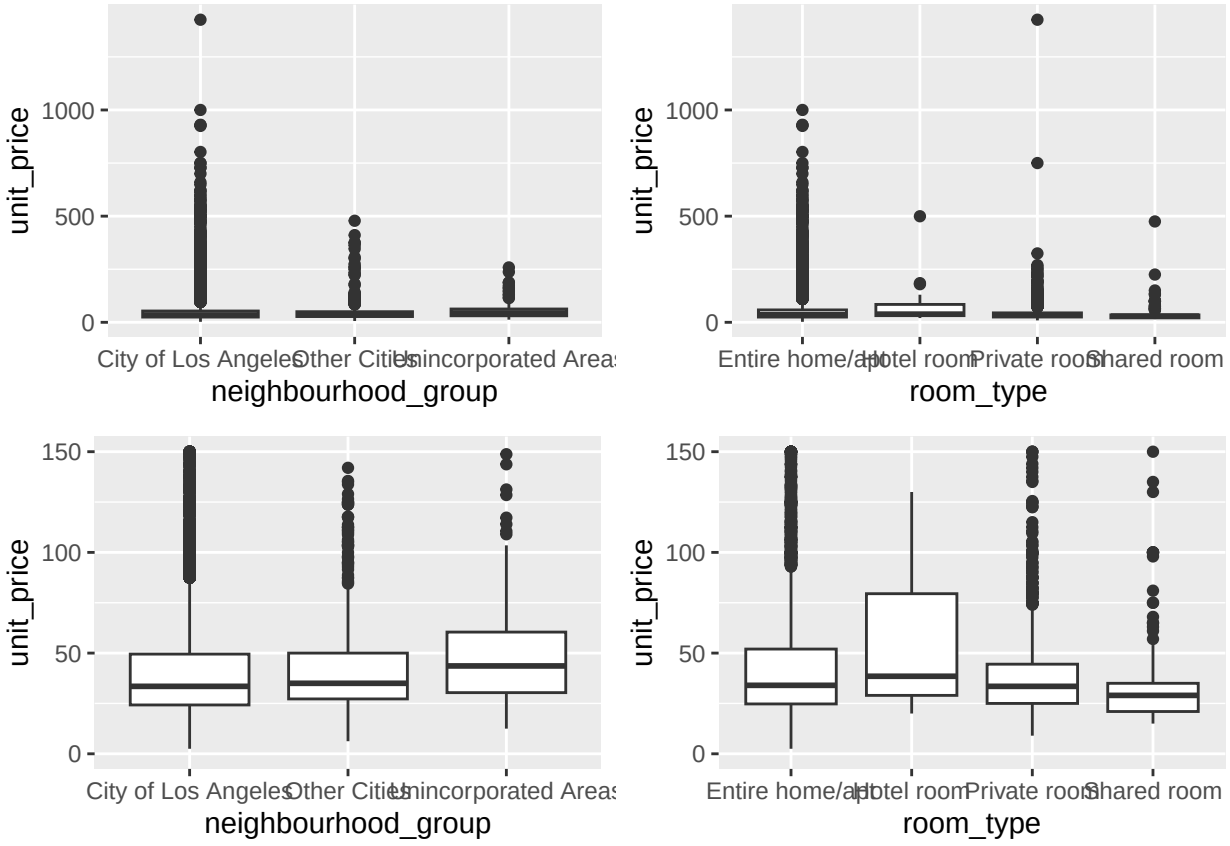

```
#plot(join_tb$neighbourhood, join_tb$price, ylim=c(1, 300))

#ggplot(join_tb) + geom_boxplot(aes(y=crime_level)) #+ ylim(0, 200)

# room_type vs. unit_price
rp1 = ggplot(final_data) + geom_boxplot(aes(x=room_type, y=unit_price))
rp2 = ggplot(final_data) + geom_boxplot(aes(x=room_type, y=unit_price)) + ylim(0, 150)
ggarrange(np1, rp1, np2, rp2, nrow=2, ncol=2)
```

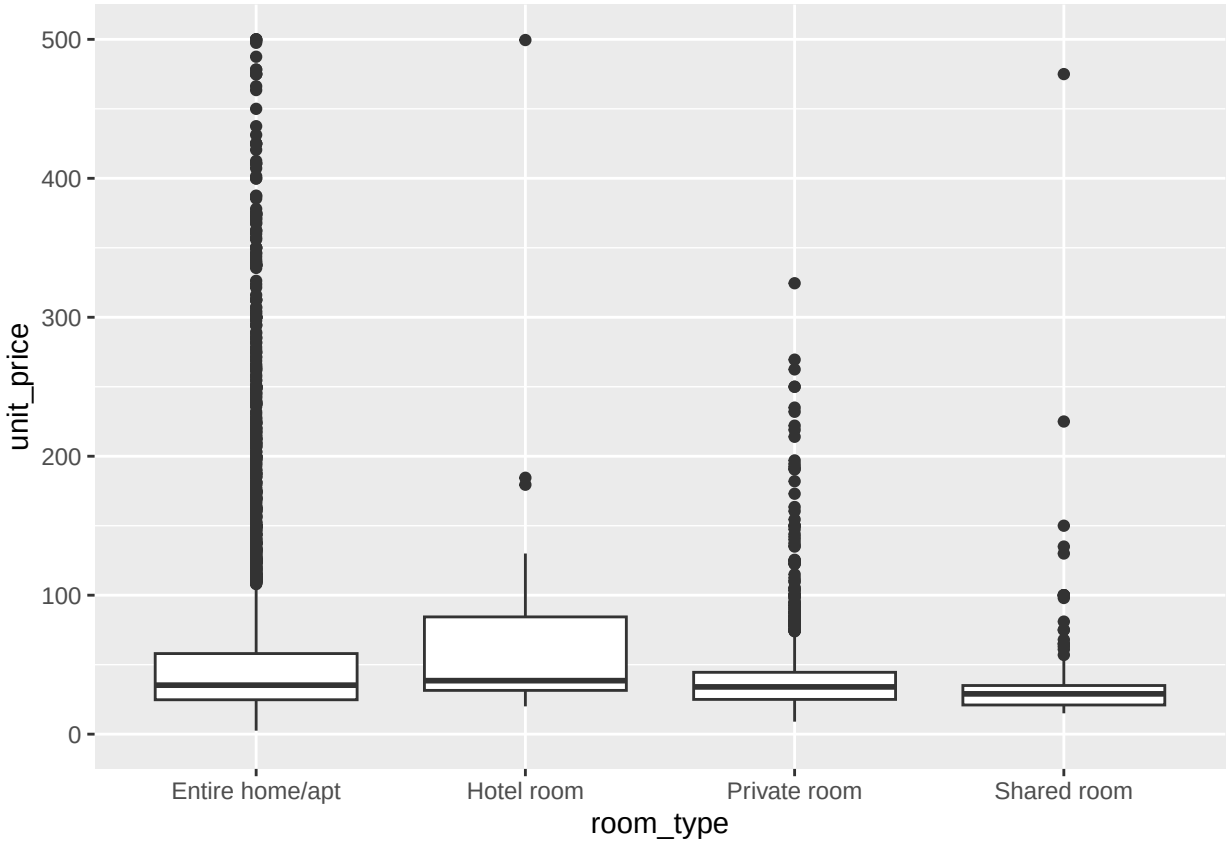
```
## Warning: Removed 389 rows containing non-finite values (‘stat_boxplot()‘).
```

```
## Warning: Removed 389 rows containing non-finite values (‘stat_boxplot()‘).
```



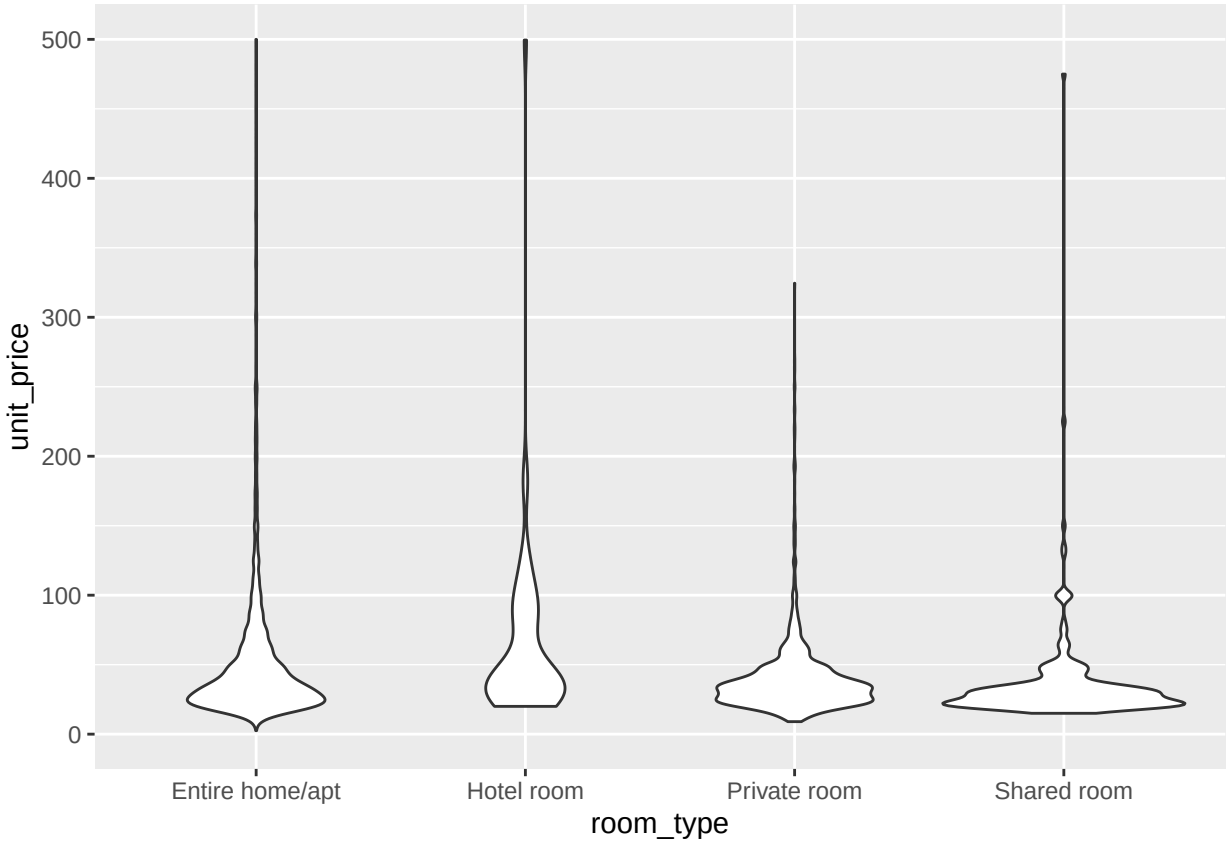

```
ggplot(final_data) + geom_boxplot(aes(x=room_type, y=unit_price)) + ylim(0, 500)
```

```
## Warning: Removed 36 rows containing non-finite values (‘stat_boxplot()‘).
```




```
ggplot(final_data) + geom_violin(aes(x=room_type, y=unit_price)) + ylim(0, 500)
```

```
## Warning: Removed 36 rows containing non-finite values (‘stat_ydensity()‘).
```

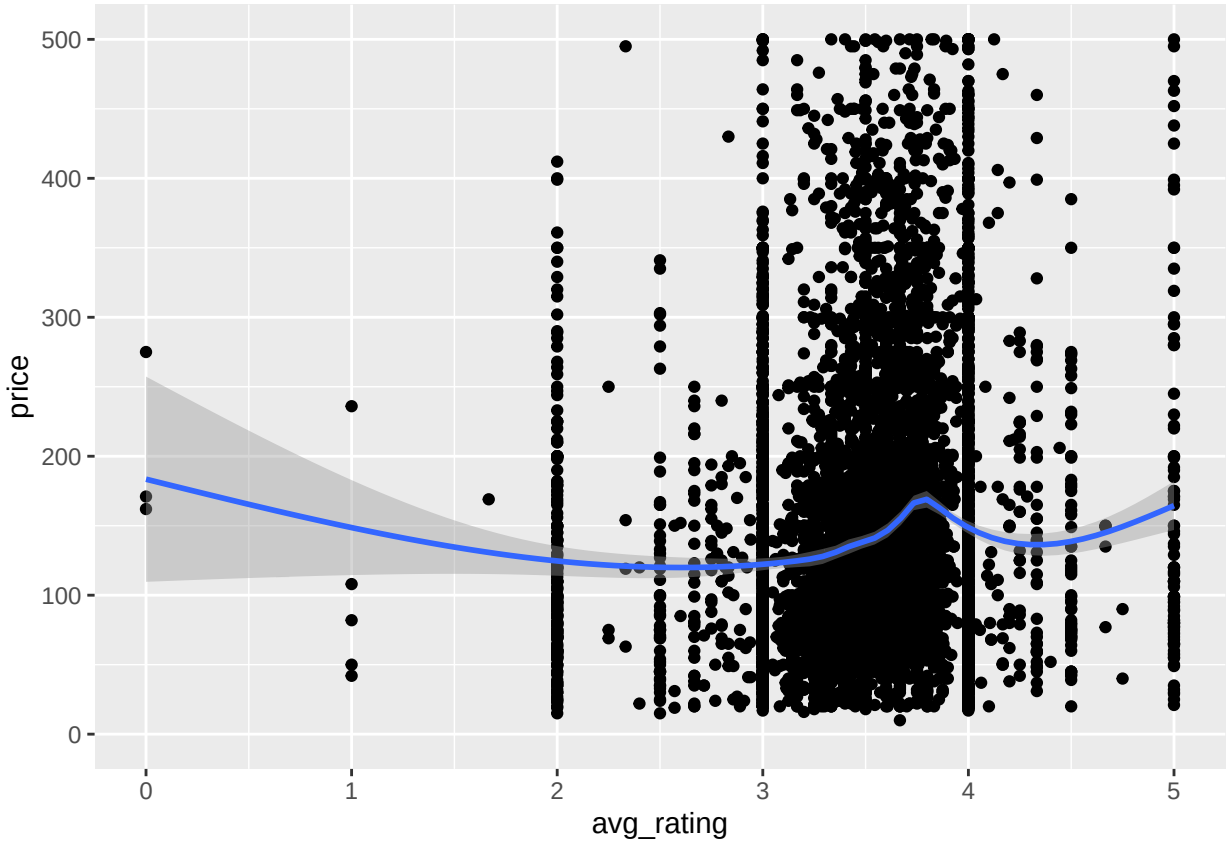



```
# avg_score vs. price
ggplot(final_data) + geom_point(aes(x=avg_rating, y=price)) + geom_smooth(aes(x=avg_rating, y=price)) +
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```

```
## Warning: Removed 474 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 474 rows containing missing values (`geom_point()`).
```



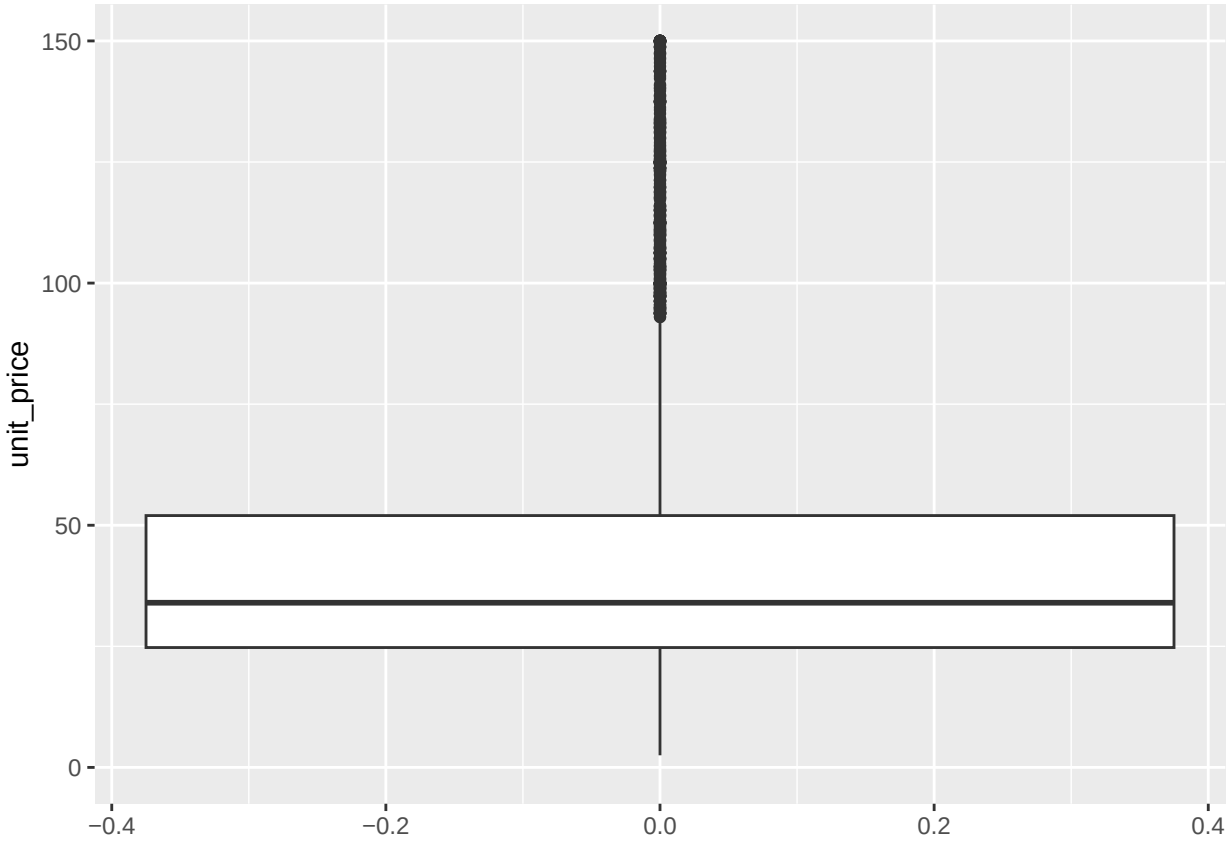

```
# avg_score vs. price vs. neighbourhood_group
#ggplot(join_tb) + geom_point(aes(x=avg_rating, y=price, color=neighbourhood_group)) + ylim(0, 500)

# avg_score vs. price vs. room_type
#ggplot(join_tb) + geom_point(aes(x=avg_score, y=price, color=room_type)) + ylim(0, 500)
```

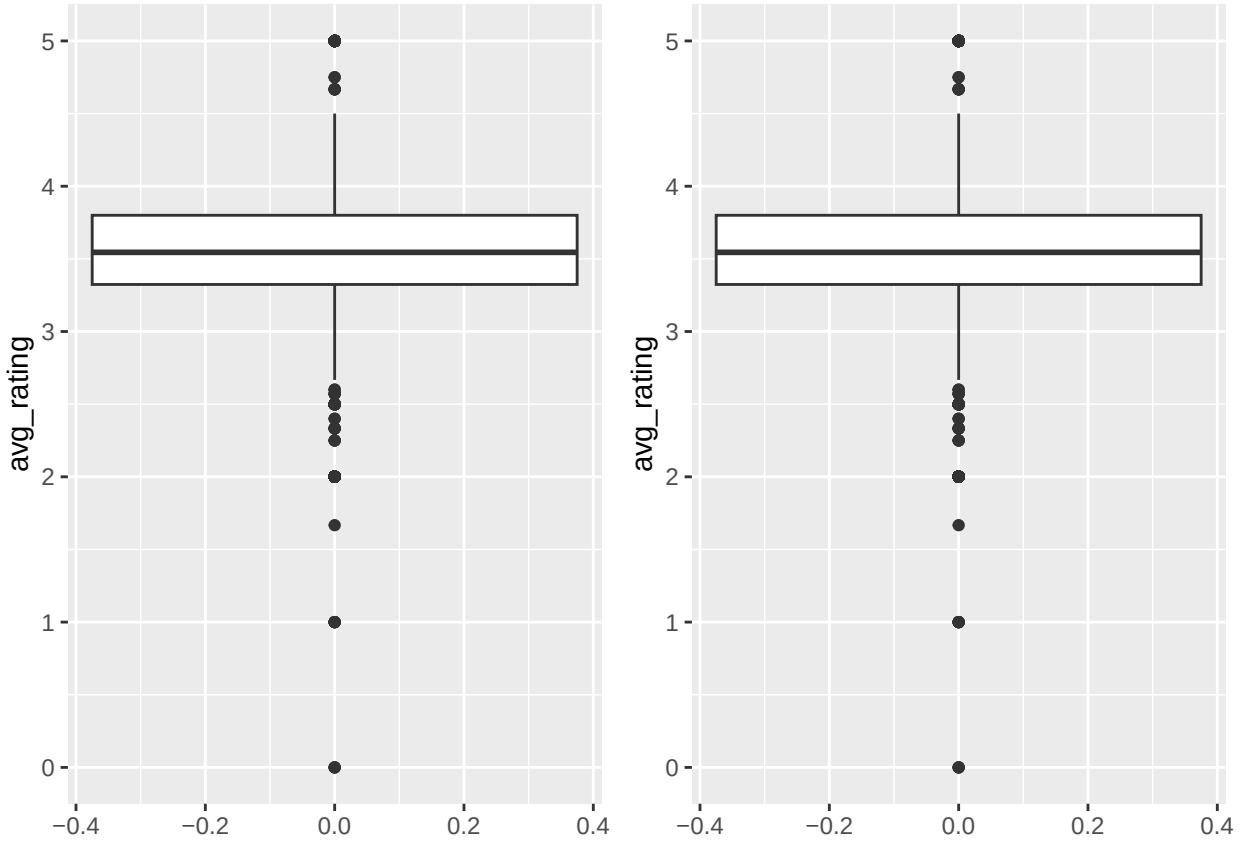
EDA - Bivariate Analysis (rating)

```
#####
Entire_data = final_data %>%
  filter(room_type=='Entire home/apt')
ggplot(Entire_data) + geom_boxplot(aes(y=unit_price)) + ylim(0, 150)
```

```
## Warning: Removed 362 rows containing non-finite values (`stat_boxplot()`).
```

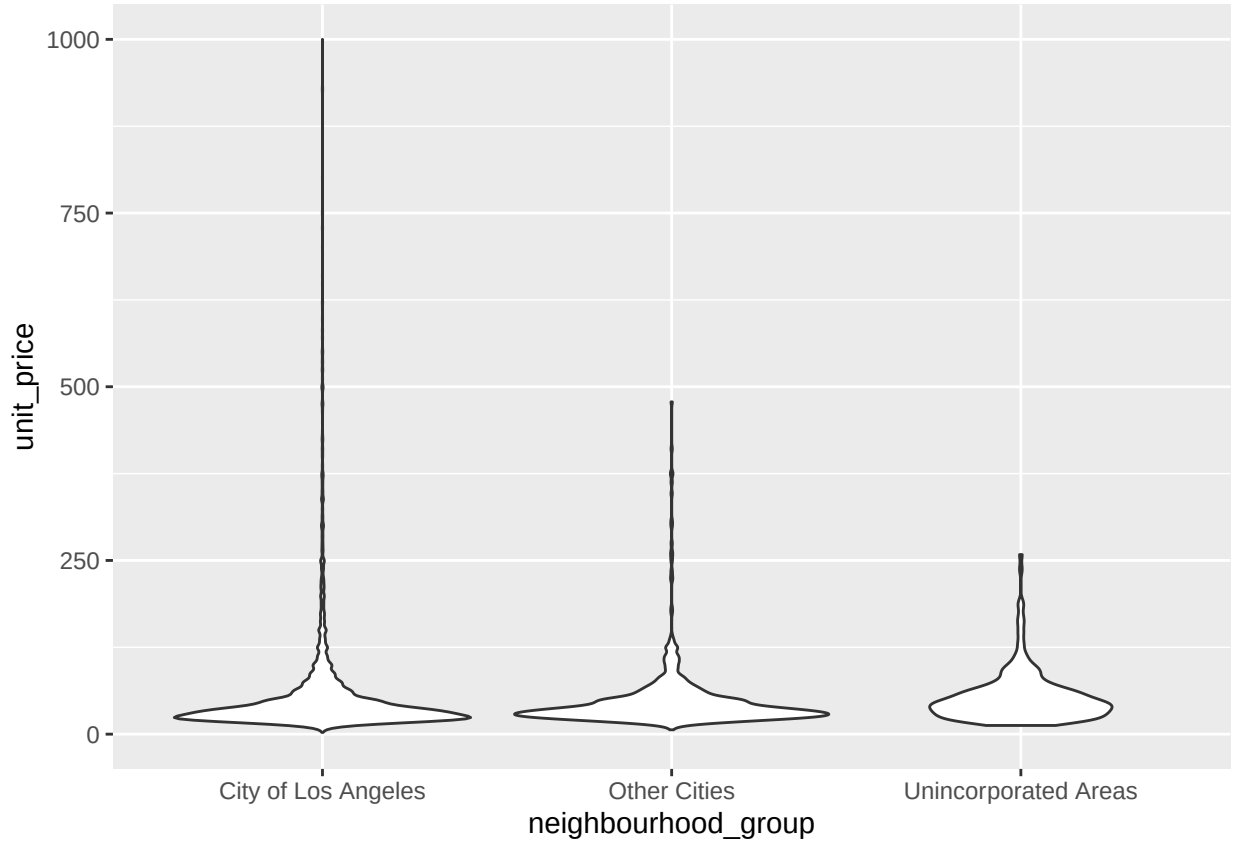



```
####
b1 = ggplot(final_data) + geom_boxplot(aes(y=avg_rating))
ggarrange(b1, b1, nrow=1)
```




```
# neighbourhood_group vs. unit_price
np1 = ggplot(final_data) + geom_boxplot(aes(x=neighbourhood_group, y=avg_rating))
np2 = ggplot(final_data) + geom_boxplot(aes(x=neighbourhood_group, y=unit_price)) + ylim(0, 150)
ggplot(final_data) + geom_violin(aes(x=neighbourhood_group, y=unit_price)) + ylim(0, 1000)
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_ydensity()`).
```



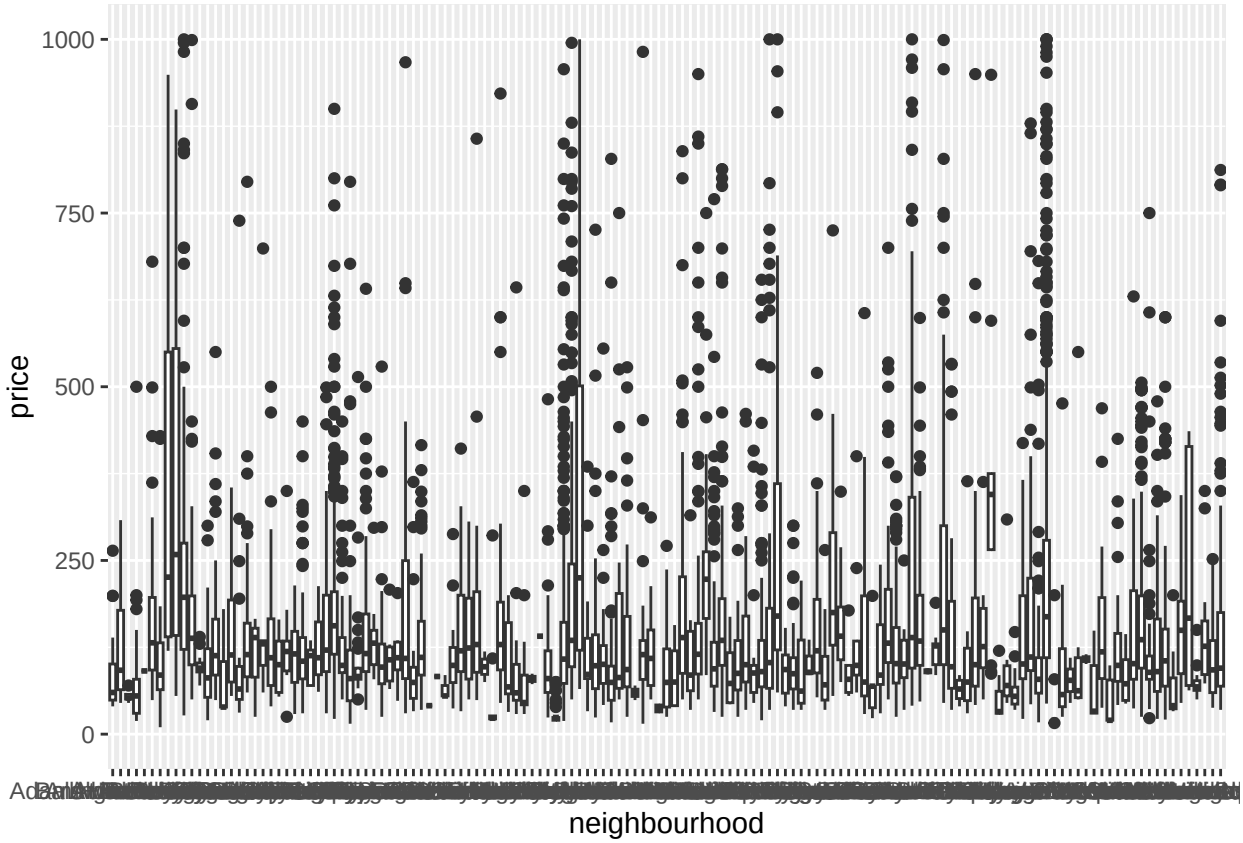

```
#plot(final_data$neighbourhood_group, join_tb$price, ylim=c(1, 300))

# neighbourhood vs. rating
distinct(join_tb, neighbourhood)
```

```
## # A tibble: 141 x 1
##    neighbourhood
##    <chr>
##  1 Hollywood
##  2 Del Rey
##  3 Atwater Village
##  4 Venice
##  5 Gardena
##  6 Mid-City
##  7 Valley Village
##  8 Silver Lake
##  9 Hollywood Hills
## 10 Highland Park
## # i 131 more rows
```

```
ggplot(join_tb) + geom_boxplot(aes(x=neighbourhood, y=price)) + ylim(0, 1000)
```

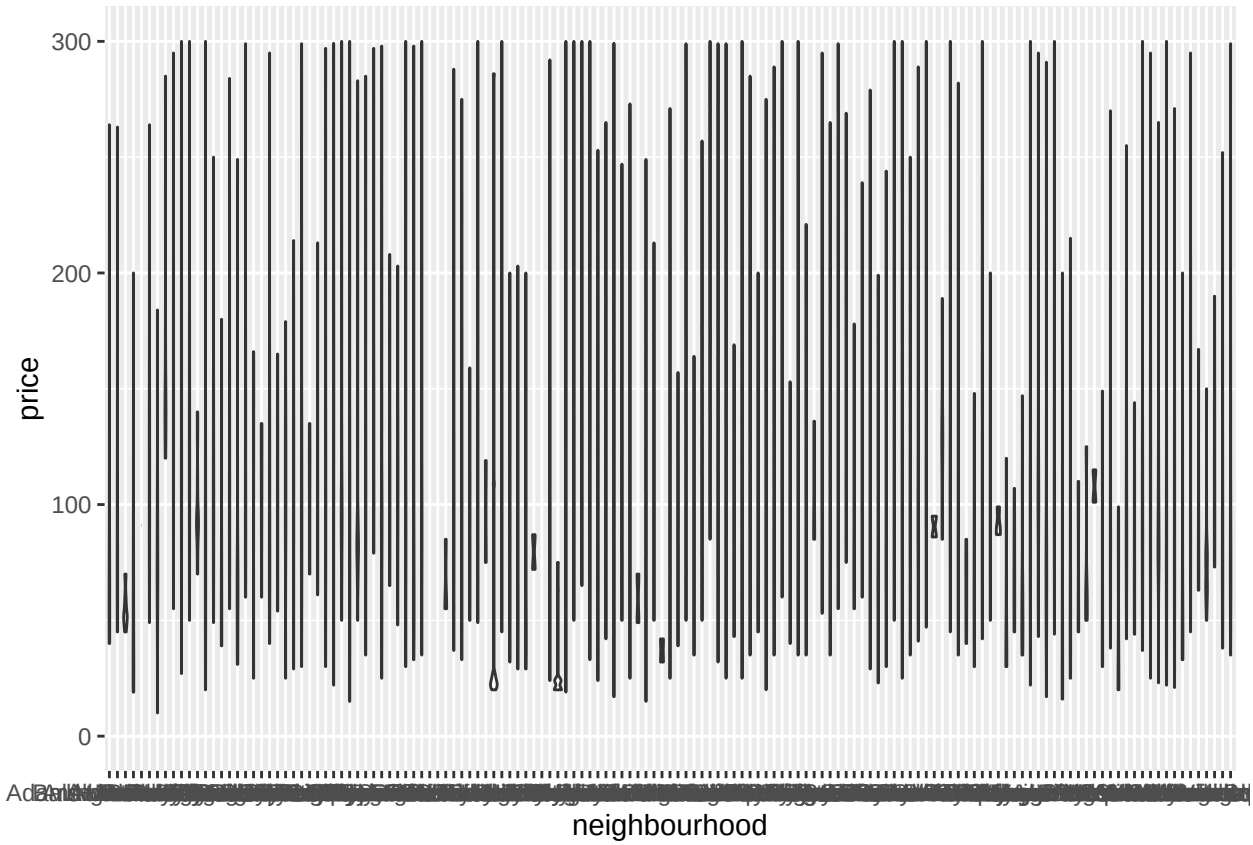
```
## Warning: Removed 164 rows containing non-finite values (`stat_boxplot()`).
```




```
ggplot(join_tb) + geom_violin(aes(x=neighbourhood, y=price)) + ylim(0, 300)
```

```
## Warning: Removed 1068 rows containing non-finite values (‘stat_ydensity()‘).
```

```
## Warning: Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
```



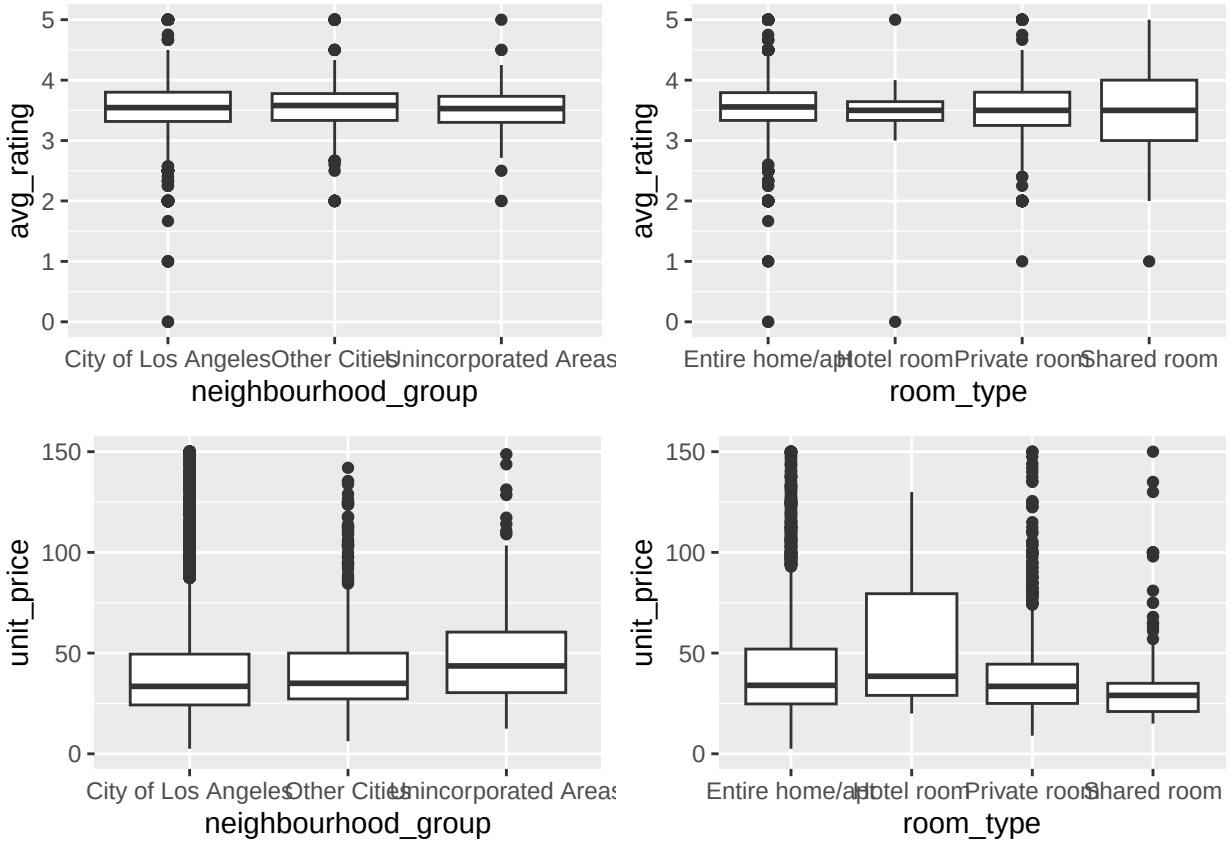

```
#plot(join_tb$neighbourhood, join_tb$price, ylim=c(1, 300))

#ggplot(join_tb) + geom_boxplot(aes(y=crime_level)) #+ ylim(0, 200)

# room_type vs. unit_price
rp1 = ggplot(final_data) + geom_boxplot(aes(x=room_type, y=avg_rating))
rp2 = ggplot(final_data) + geom_boxplot(aes(x=room_type, y=unit_price)) + ylim(0, 150)
ggarrange(np1, rp1, np2, rp2, nrow=2, ncol=2)
```

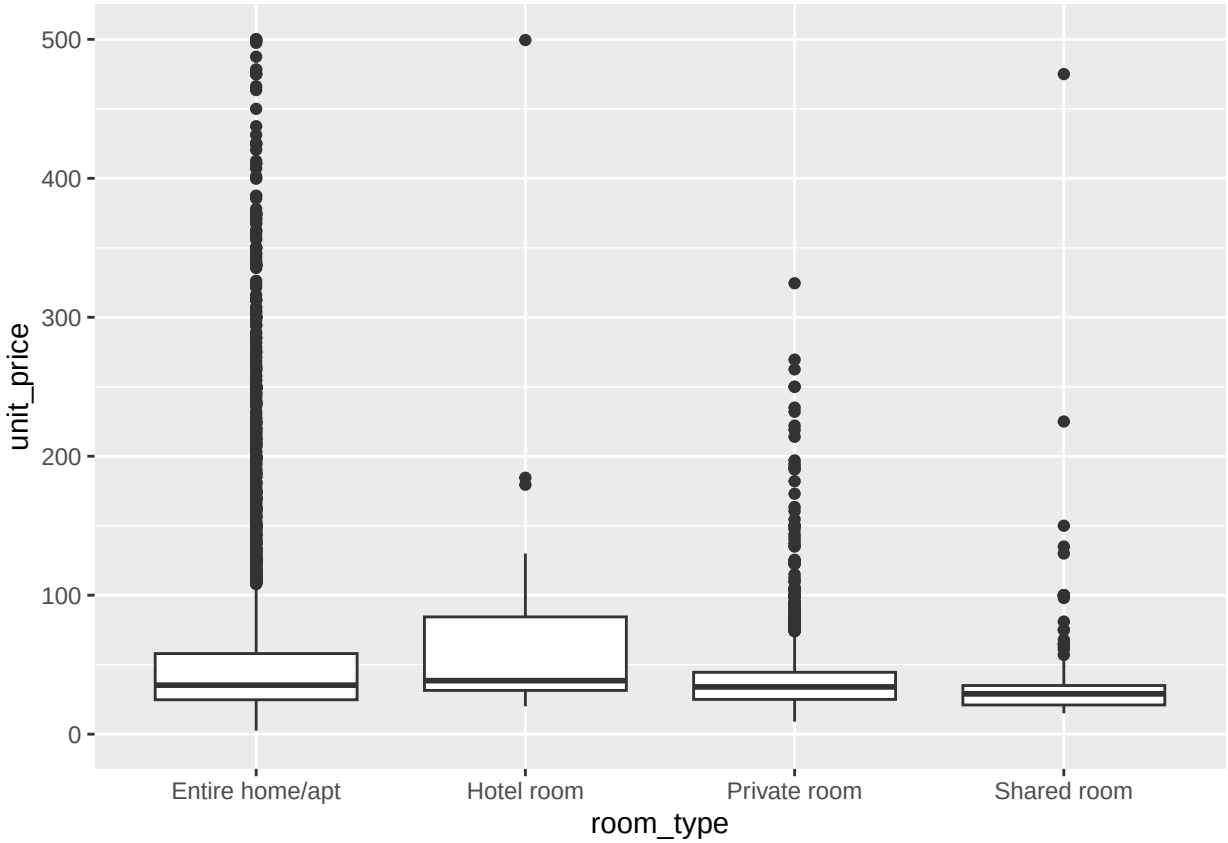
```
## Warning: Removed 389 rows containing non-finite values (‘stat_boxplot()‘).
```

```
## Warning: Removed 389 rows containing non-finite values (‘stat_boxplot()‘).
```



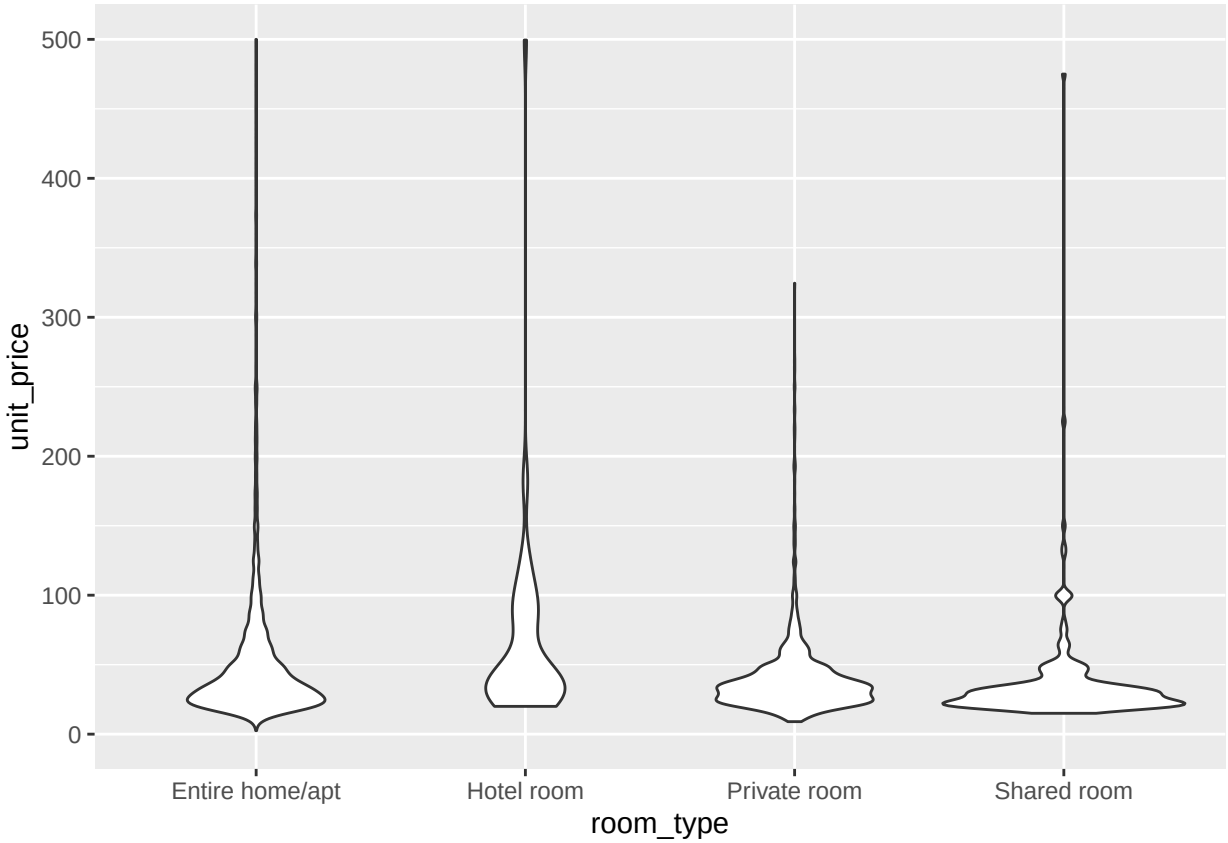

```
ggplot(final_data) + geom_boxplot(aes(x=room_type, y=unit_price)) + ylim(0, 500)
```

```
## Warning: Removed 36 rows containing non-finite values (‘stat_boxplot()‘).
```




```
ggplot(final_data) + geom_violin(aes(x=room_type, y=unit_price)) + ylim(0, 500)
```

```
## Warning: Removed 36 rows containing non-finite values (‘stat_ydensity()‘).
```




```
# avg_score vs. price
#ggplot(final_data) + geom_point(aes(x=avg_score, y=price)) + geom_smooth(aes(x=avg_score, y=price)) +

# avg_score vs. price vs. neighbourhood_group
#ggplot(join_tb) + geom_point(aes(x=avg_rating, y=price, color=neighbourhood_group)) + ylim(0, 500)

# avg_score vs. price vs. room_type
#ggplot(join_tb) + geom_point(aes(x=avg_score, y=price, color=room_type)) + ylim(0, 500)
```

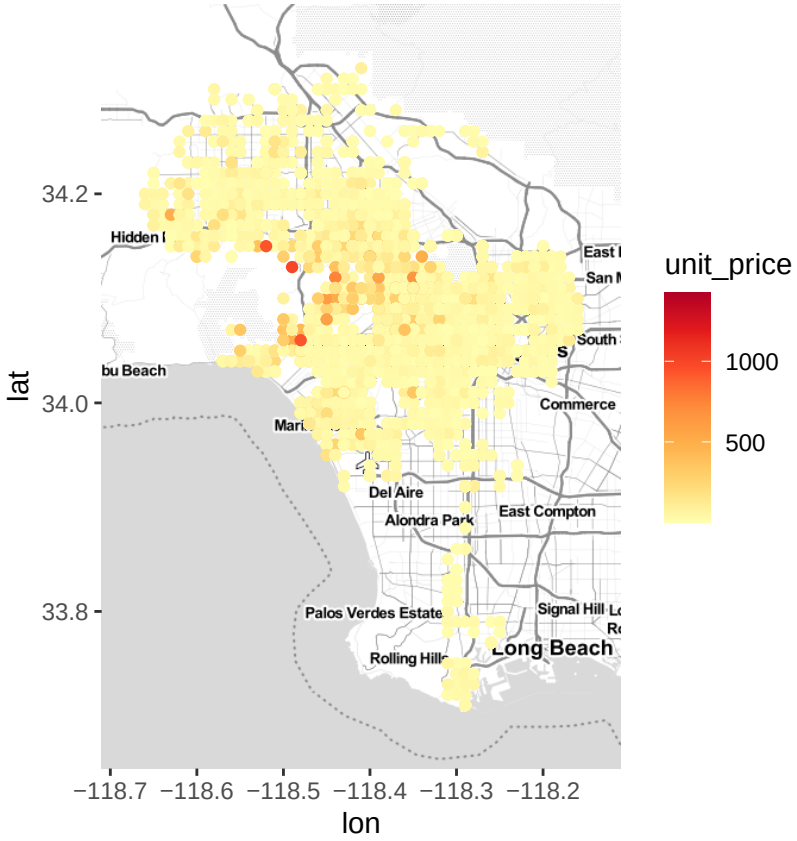
Plot all housings unit_price on LA map

```
final_data_roundprice = cbind(final_data)
final_data_roundprice$unit_price =
  round(final_data_roundprice$unit_price, 0)
height <- max(final_data_roundprice$latitude) - min(final_data_roundprice$latitude)
width <- max(final_data_roundprice$longitude) - min(final_data_roundprice$longitude)
LA_borders <- c(bottom  = min(final_data_roundprice$latitude)  - 0.1 * height,
                top     = max(final_data_roundprice$latitude)  + 0.1 * height,
                left    = min(final_data_roundprice$longitude) - 0.1 * width,
                right   = max(final_data_roundprice$longitude) + 0.1 * width)

map <- get_stamenmap(LA_borders, zoom = 10, maptype = "toner-lite")
```

```
## i Map tiles by Stamen Design, under CC BY 3.0. Data by OpenStreetMap, under ODbL.
```

```
ggmap(map) +
    geom_point(data = final_data_roundprice, mapping = aes(x = longitude, y = latitude, color=unit_price
```



Cross table (neighbourhood_group vs room_type)

```
source("http://pcwww.liv.ac.uk/~william/R/crosstab.r") #crosstab
crosstab(final_data, row.vars = "room_type", col.vars = "neighbourhood_group", type = "f")
```

```
##                 neighbourhood_group City of Los Angeles Other Cities Unincorporated Areas  Sum
## room_type
## Entire home/apt                                5704          606                 172 6482
## Hotel room                                       57            1                   0   58
## Private room                                   1544          210                  46 1800
## Shared room                                     221            6                   9  236
## Sum                                            7526          823                 227 8576
```

```
crosstab(final_data, row.vars = "neighbourhood_group", col.vars = "room_type", type = "f")
```

```
##                      room_type Entire home/apt Hotel room Private room Shared room  Sum
## neighbourhood_group
## City of Los Angeles                       5704         57         1544         221 7526
## Other Cities                               606          1          210           6  823
## Unincorporated Areas                       172          0           46           9  227
## Sum                                       6482         58         1800         236 8576
```

```
library(knitr)
final_data %>%
  group_by(room_type, neighbourhood_group) %>%
  summarise(count=n()) %>%
  spread(neighbourhood_group, count) #%>%
```

```
## `summarise()` has grouped output by 'room_type'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 4 x 4
## # Groups:   room_type [4]
##   room_type       `City of Los Angeles` `Other Cities` `Unincorporated Areas`
##   <chr>                           <int>          <int>                  <int>
## 1 Entire home/apt                  5704            606                    172
## 2 Hotel room                         57              1                     NA
## 3 Private room                     1544            210                     46
## 4 Shared room                       221              6                      9
```

```
#  kable()
```

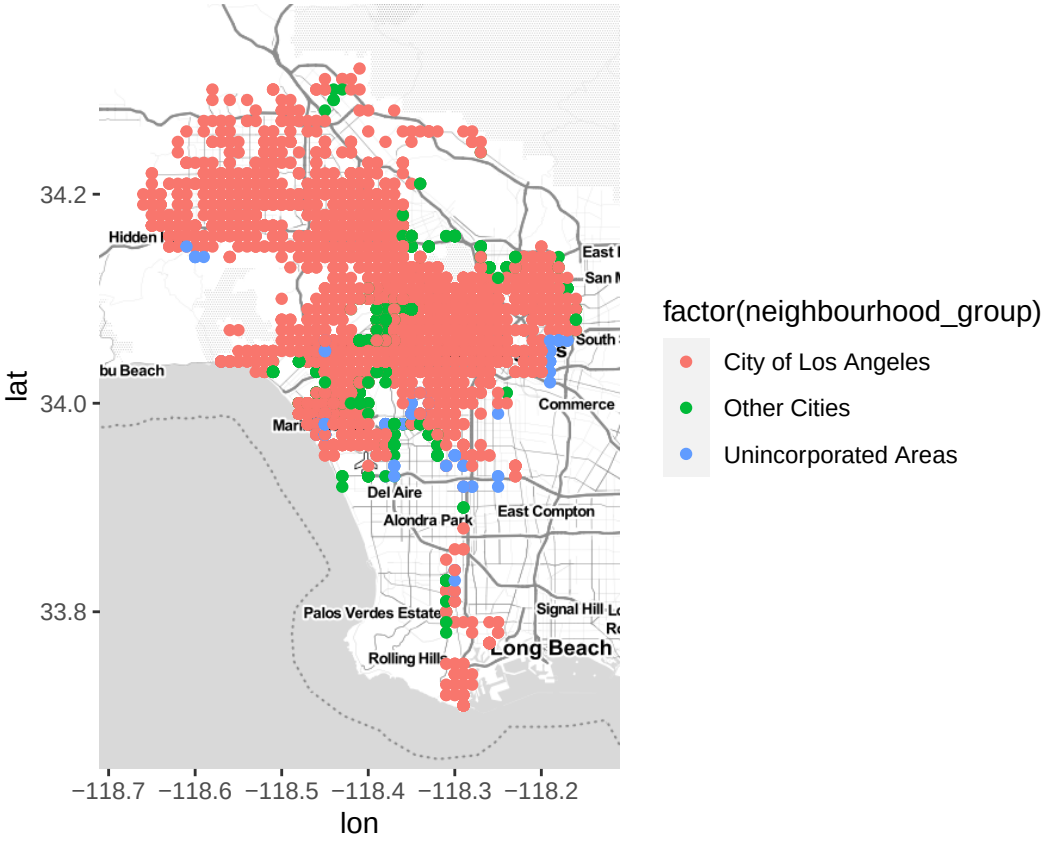
Plot all neighbourhood_group on LA map

```
height <- max(final_data$latitude) - min(final_data$latitude)
width <- max(final_data$longitude) - min(final_data$longitude)
LA_borders <- c(bottom  = min(final_data$latitude)  - 0.1 * height,
                top     = max(final_data$latitude)  + 0.1 * height,
                left    = min(final_data$longitude) - 0.1 * width,
                right   = max(final_data$longitude) + 0.1 * width)

map <- get_stamenmap(LA_borders, zoom = 10, maptype = "toner-lite")
```

```
## i Map tiles by Stamen Design, under CC BY 3.0. Data by OpenStreetMap, under ODbL.
```

```
ggmap(map) +
    geom_point(data = final_data, mapping = aes(x = longitude, y = latitude, color=factor(neighbourhood
```




```
# + scale_color_distiller(palette = "YlOrRd", direction = 1)
```

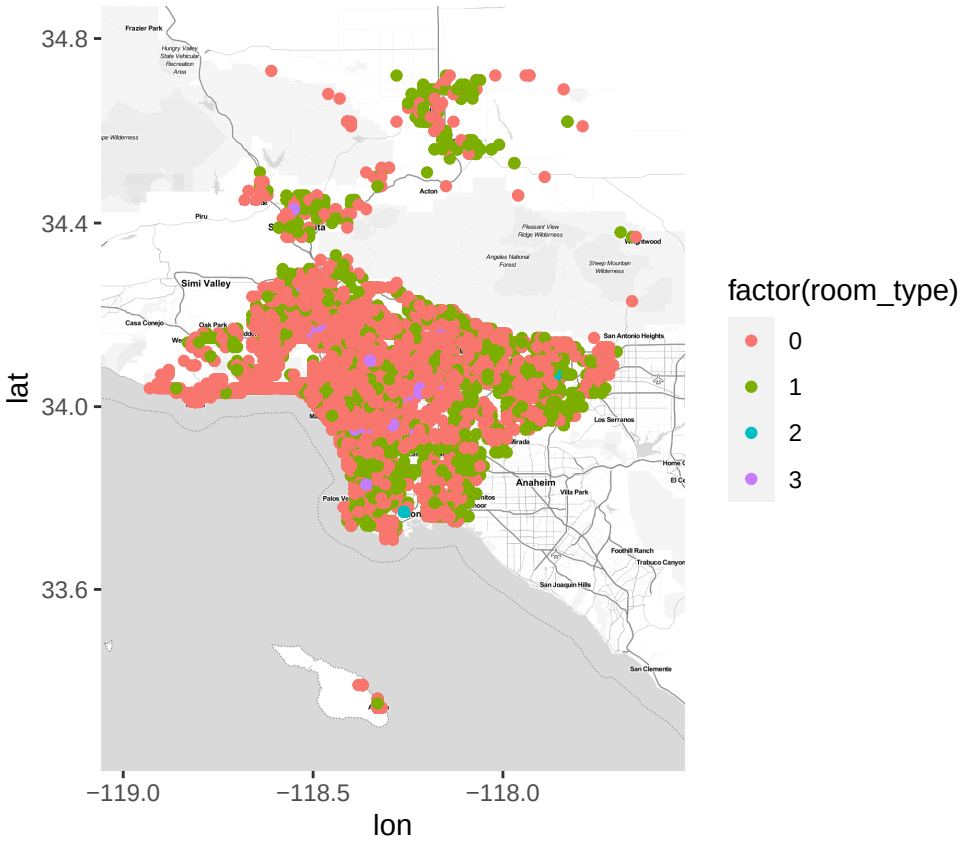
Plot all room_type on LA map

```
listings_tb = as_tibble(read.csv("./input/listings.csv"))
height <- max(listings_tb$latitude) - min(listings_tb$latitude)
width <- max(listings_tb$longitude) - min(listings_tb$longitude)
LA_borders <- c(bottom  = min(listings_tb$latitude)  - 0.1 * height,
                top     = max(listings_tb$latitude)  + 0.1 * height,
                left    = min(listings_tb$longitude) - 0.1 * width,
                right   = max(listings_tb$longitude) + 0.1 * width)

map <- get_stamenmap(LA_borders, zoom = 10, maptype = "toner-lite")
```

```
## i Map tiles by Stamen Design, under CC BY 3.0. Data by OpenStreetMap, under ODbL.
```

```
ggmap(map) +
    geom_point(data = listings_tb, mapping = aes(x = longitude, y = latitude, color=factor(room_type)))
```




```
# values count
listings_tb %>%
  group_by(room_type) %>%
  summarise(no_rows = length(room_type))
```

```
## # A tibble: 4 x 2
##   room_type no_rows
##       <int>   <int>
## 1         0   12450
## 2         1    5551
## 3         2     125
## 4         3     498
```

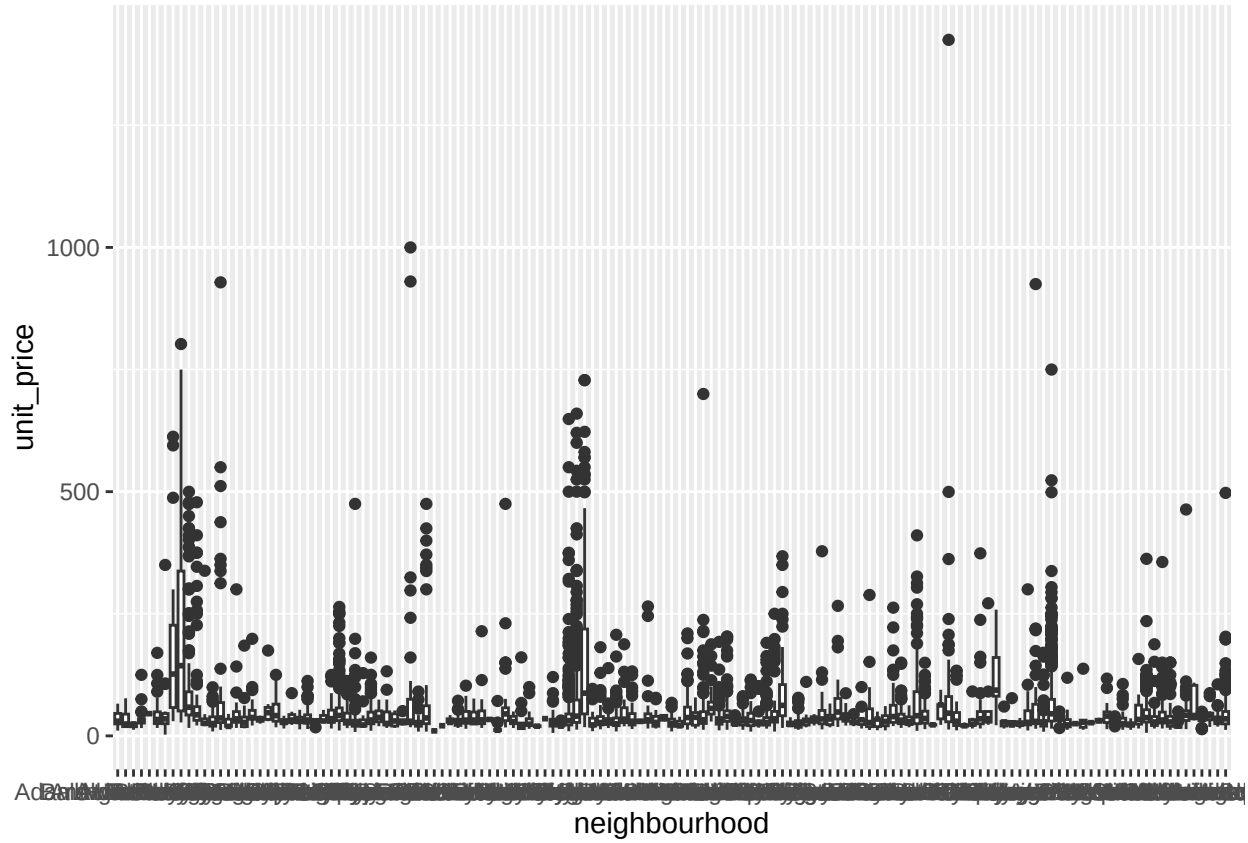
EDA - Bivariate Analysis (Categories vs Measures)

```
distinct(final_data, neighbourhood)
```

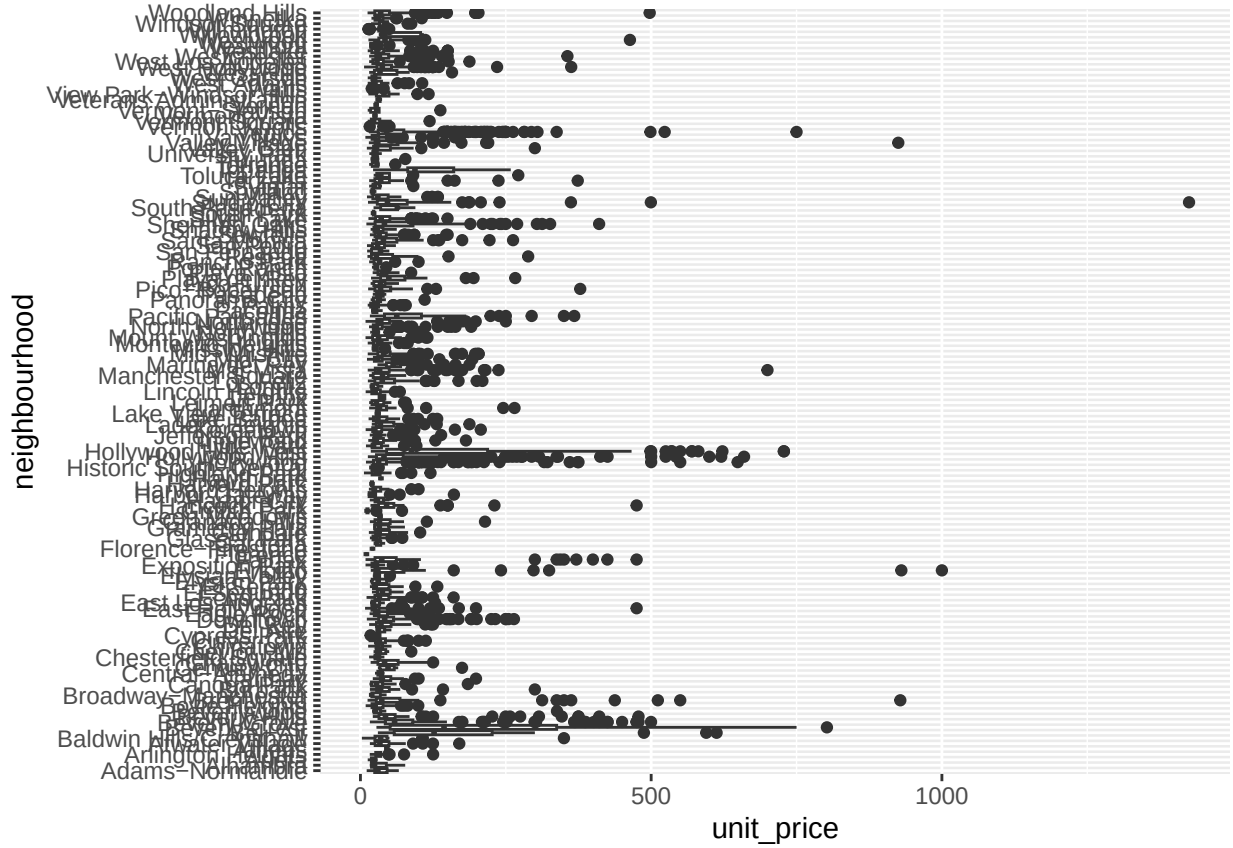
```
## # A tibble: 141 x 1
##    neighbourhood
##    <chr>
##  1 Hollywood
##  2 Hollywood Hills
##  3 Hollywood Hills West
##  4 Del Rey
```

```
##  5 Culver City
##  6 Atwater Village
##  7 Glendale
##  8 Venice
##  9 Marina del Rey
## 10 Santa Monica
## # i 131 more rows
```

```
ggplot(final_data) + geom_boxplot(aes(x=neighbourhood, y=unit_price))
```




```
ggplot(final_data) + geom_boxplot(aes(x=unit_price, y=neighbourhood))
```



take off the outlier

```
final_data = final_data %>%
  filter(unit_price<1100)
```

o

o

o

o

o

o

o

o

o

o

o

o