Analysis

Contents

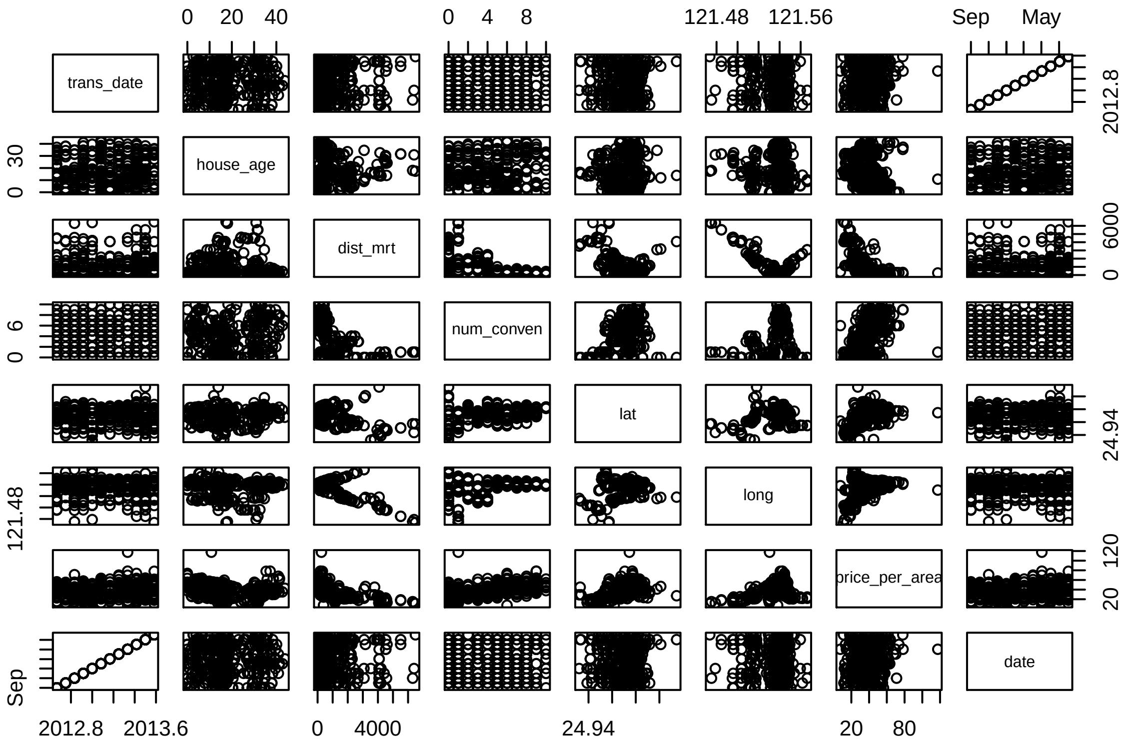
Sub-problem 1: load and summarize the data (20 points)

```r
real_estate = read_excel("Real estate valuation data set.xlsx")

colnames(real_estate) = c(
  "trans_num",
  "trans_date",
  "house_age",
  "dist_mrt",
  "num_conven",
  "lat",
  "long",
  "price_per_area"
)

real_estate = real_estate %>%
  mutate(date = date_decimal(trans_date)) %>%
  select(-trans_num) %>%
  arrange(price_per_area)

summary(real_estate)
```

```
##    trans_date     house_age        dist_mrt         num_conven
##  Min.   :2013   Min.   : 0.000   Min.   :  23.38   Min.   : 0.000
##  1st Qu.:2013   1st Qu.: 9.025   1st Qu.: 289.32   1st Qu.: 1.000
##  Median :2013   Median :16.100   Median : 492.23   Median : 4.000
##  Mean   :2013   Mean   :17.713   Mean   :1083.89   Mean   : 4.094
##  3rd Qu.:2013   3rd Qu.:28.150   3rd Qu.:1454.28   3rd Qu.: 6.000
##  Max.   :2014   Max.   :43.800   Max.   :6488.02   Max.   :10.000
##       lat             long        price_per_area        date
```

```
##  Min.   :24.93   Min.   :121.5   Min.   :  7.60   Min.   :2012-09-01 00:00:01
##  1st Qu.:24.96   1st Qu.:121.5   1st Qu.: 27.70   1st Qu.:2012-12-01 12:00:01
##  Median :24.97   Median :121.5   Median : 38.45   Median :2013-03-02 20:00:01
##  Mean   :24.97   Mean   :121.5   Mean   : 37.98   Mean   :2013-02-24 07:22:36
##  3rd Qu.:24.98   3rd Qu.:121.5   3rd Qu.: 46.60   3rd Qu.:2013-06-02 02:00:01
##  Max.   :25.01   Max.   :121.6   Max.   :117.50   Max.   :2013-08-01 21:59:58
```

```r
pairs(real_estate)
```



We first explore the general relationships between variables with a pair plot. Our primary interest at this time is in examining how the outcome variable, price per area, interacts with the variables. We see that price per area seems to be fairly evenly distributed across dates, somewhat positively increasing with number of convenience stores, somewhat decreasing with distance from an MRT station, and there seems to be a very weak negative correlation with house age. Price per area's correlation with latitude and longitute is difficult to assess from the graphs.

Latitude and Longitude do not behave like typical continuous variables since from domain knowledge we know that these indicate geographic locations when taken together and so we hypothesize at this time that to get useful information from these two variables we will need to consider them together. We do some work now to explore a combination of latitude and longitude.

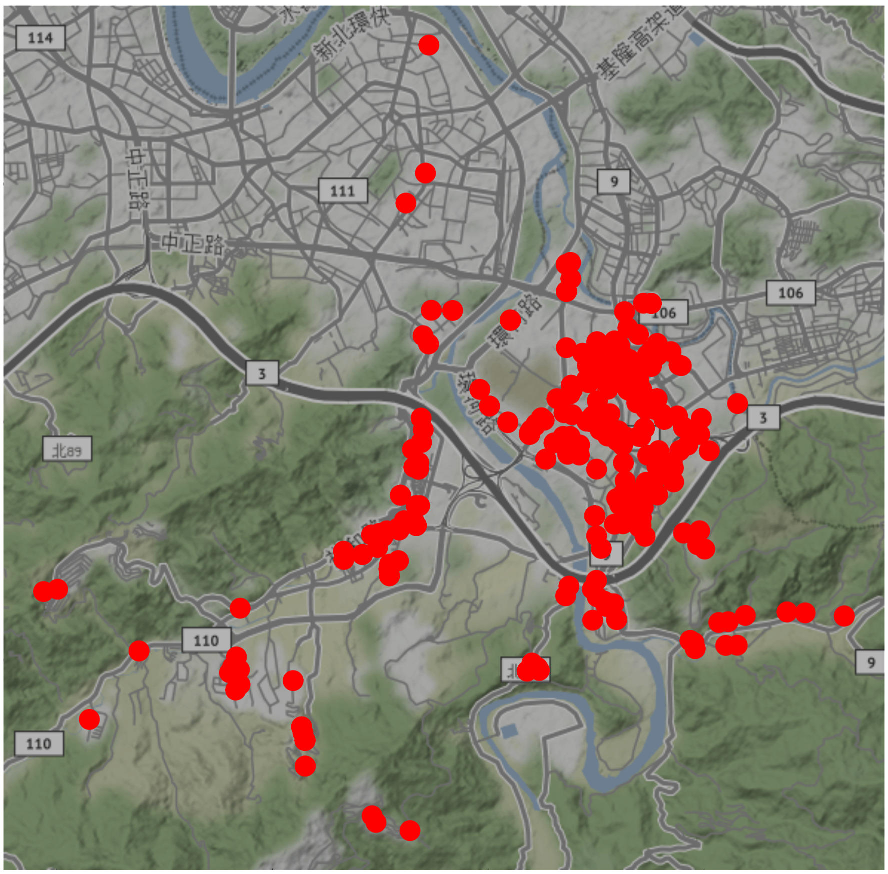
We find the location of the observations on a map and then we try to identify observations which are close together. We do this due to prior domain knowledge about real estate price clustering: in brief, houses in a neighborhood tend to have price fluctuations together as neighborhoods become more or less desirable to live in due to some changes in attributes. This is useful because we can likely capture lots of implicit information in our model if we identify neighborhoods well.

```r
real_estate %>%
  qmplot(long, lat, data = ., color = I("red"), size = I(3), darken = 0.3)
```

```
## Using zoom = 13...
```

```
## Source : http://tile.stamen.com/terrain/13/6860/3507.png
```

```
## Source : http://tile.stamen.com/terrain/13/6861/3507.png
## Source : http://tile.stamen.com/terrain/13/6862/3507.png
## Source : http://tile.stamen.com/terrain/13/6860/3508.png
## Source : http://tile.stamen.com/terrain/13/6861/3508.png
## Source : http://tile.stamen.com/terrain/13/6862/3508.png
## Source : http://tile.stamen.com/terrain/13/6860/3509.png
## Source : http://tile.stamen.com/terrain/13/6861/3509.png
## Source : http://tile.stamen.com/terrain/13/6862/3509.png
```

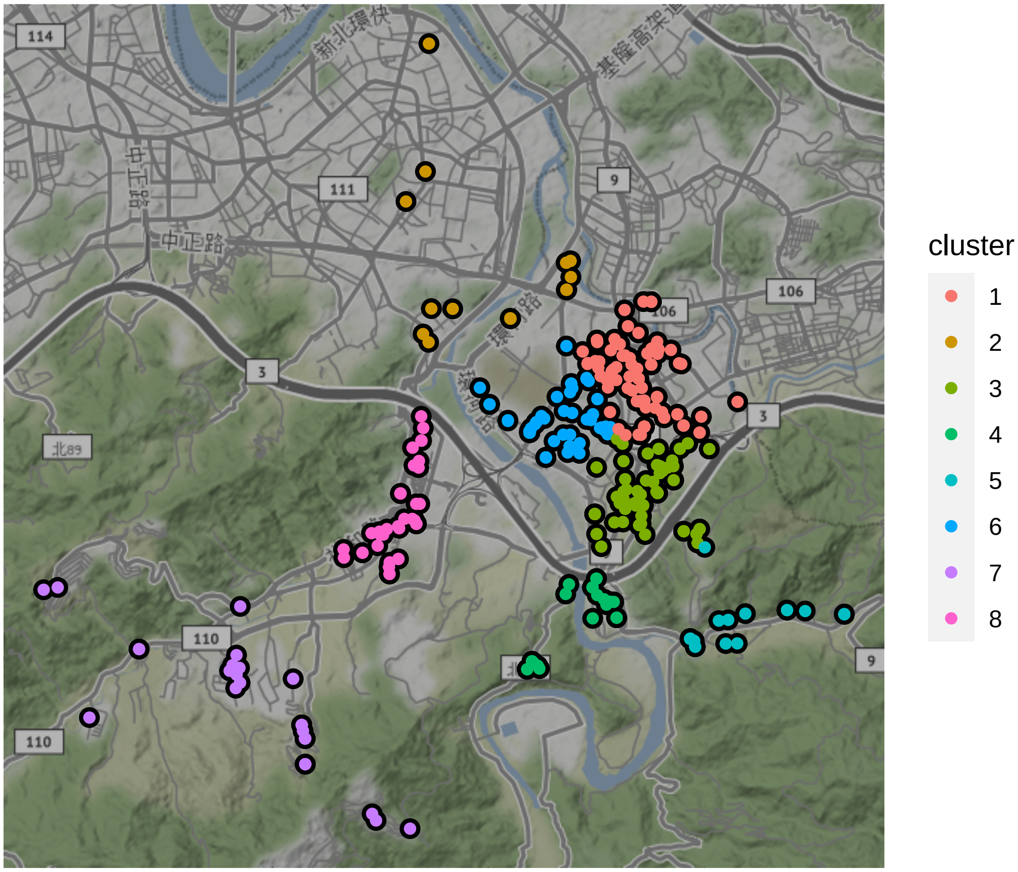


We see that there is a mix of rural and what seems to be more urban houses in our dataset. It is likely that these urban vs rural houses will have different implicit market characteristics and so our crudest level of separation may be at this level rural vs urban, but we may explore additional separation that could be important if there are additional distinct characteristics within the rural and urban categories. We use a k-means algorithm with 8 centers to segment the data and we will aggregate up later for analysis and test appropriateness using cross-validation.

```r
set.seed(4)
real_estate = real_estate %>%
  mutate(cluster = kmeans(real_estate[,c("lat","long")], centers = 8, nstart = 25) %$% factor(cluster))

taiwan_plot = real_estate %>%
  qmplot(long, lat, data = ., size = I(3), darken = 0.3) +
  geom_point(aes(x = long, y = lat, color = cluster))
```

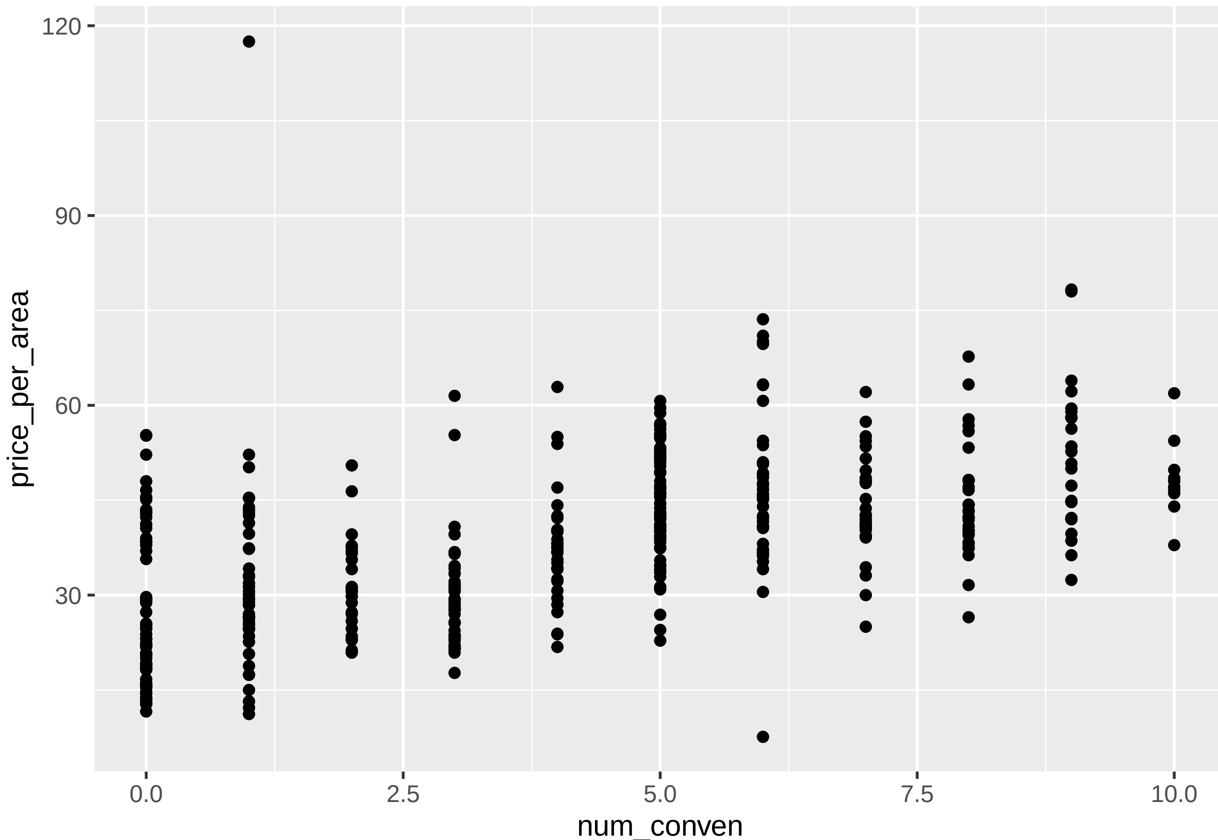
```
## Using zoom = 13...
```

```r
taiwan_plot
```

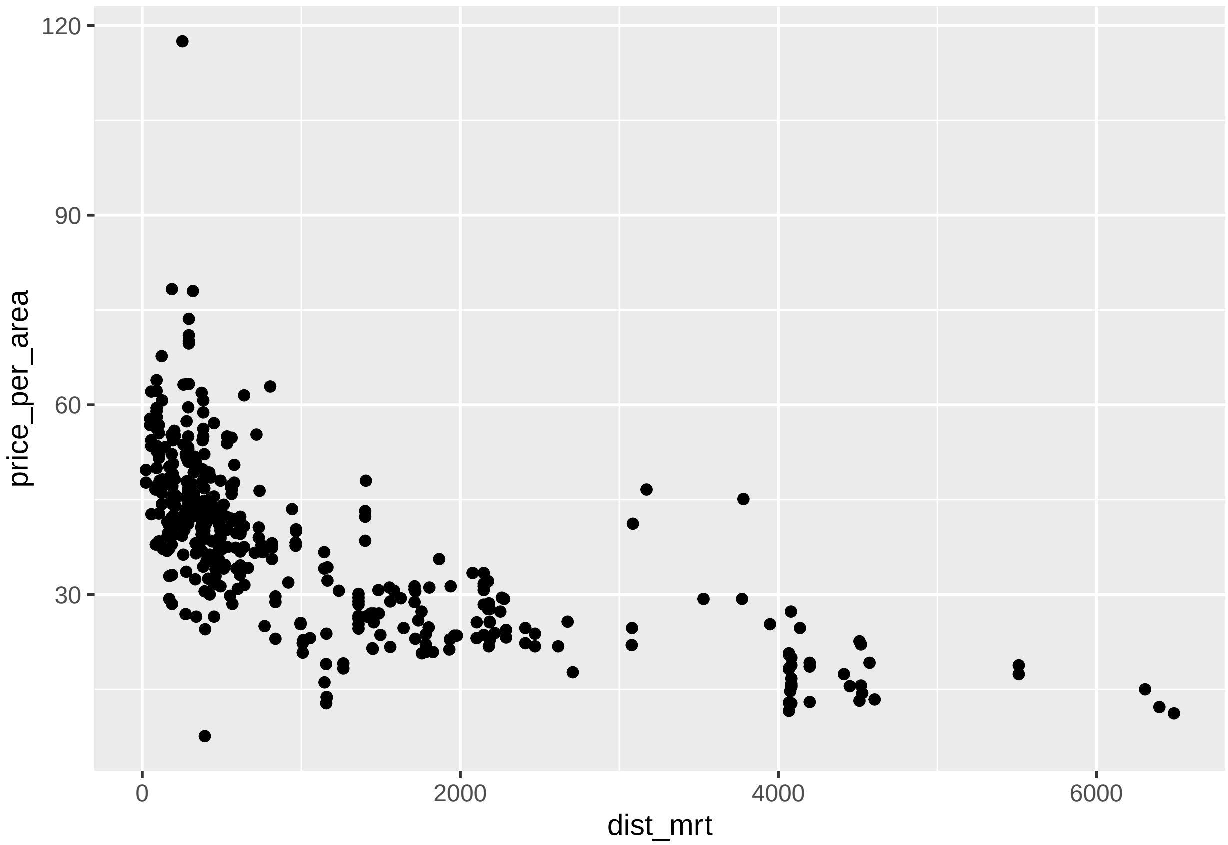


We focus now on the other three variables: number of convenience stores nearby, the distance from the nearest MRT station, and the house age.

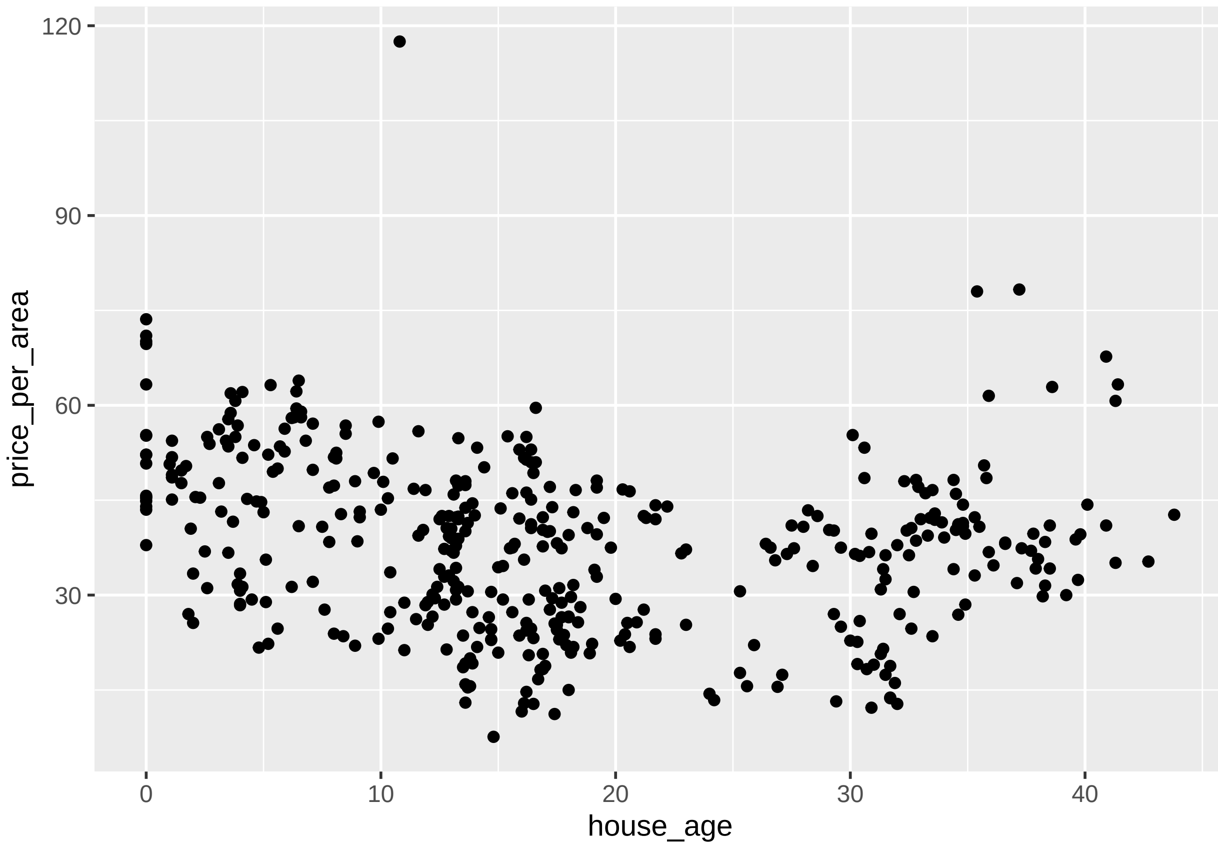
```r
conv_plot = real_estate %>%
  ggplot(aes(x = num_conven, y = price_per_area)) +
  geom_point()

mrt_plot = real_estate %>%
  ggplot(aes(x = dist_mrt, y = price_per_area)) +
  geom_point()

houseage_plot = real_estate %>%
  ggplot(aes(x = house_age, y = price_per_area)) +
  geom_point()

conv_plot
```




```
mrt_plot
```

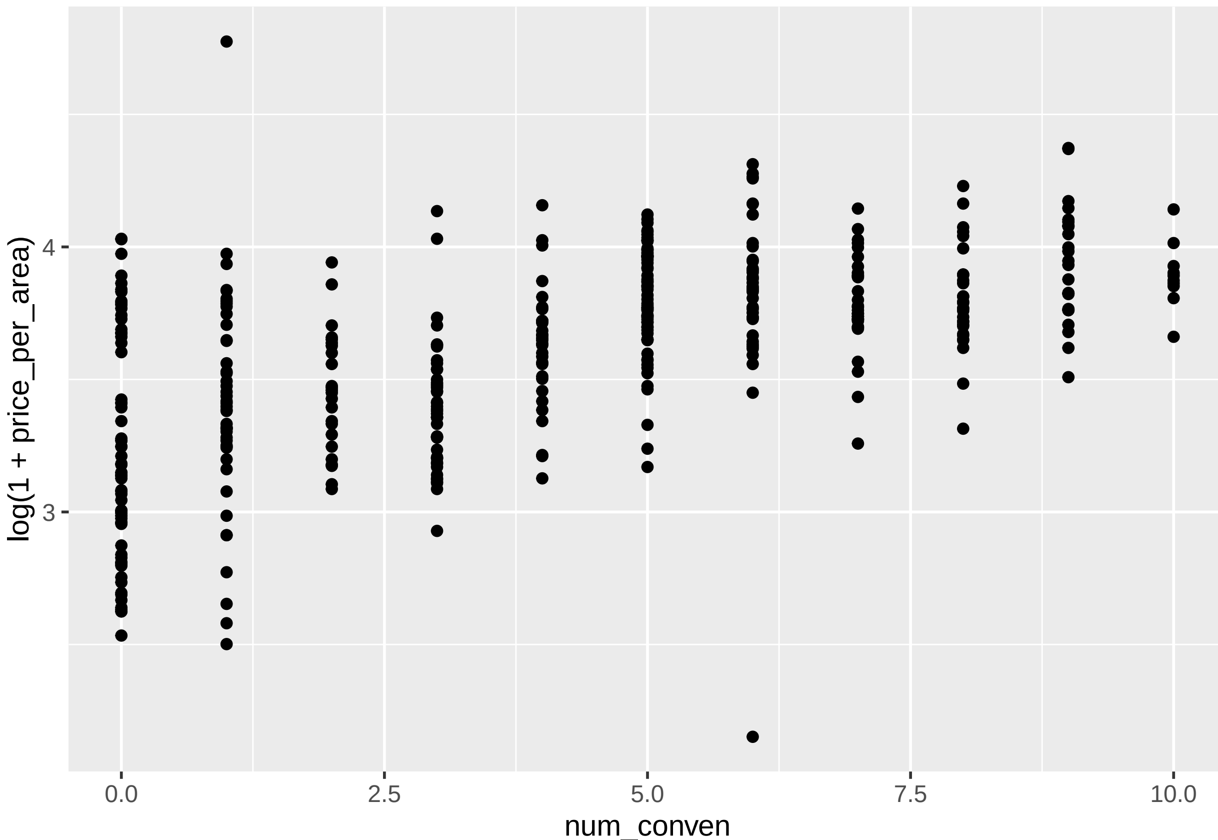



```
houseage_plot
```

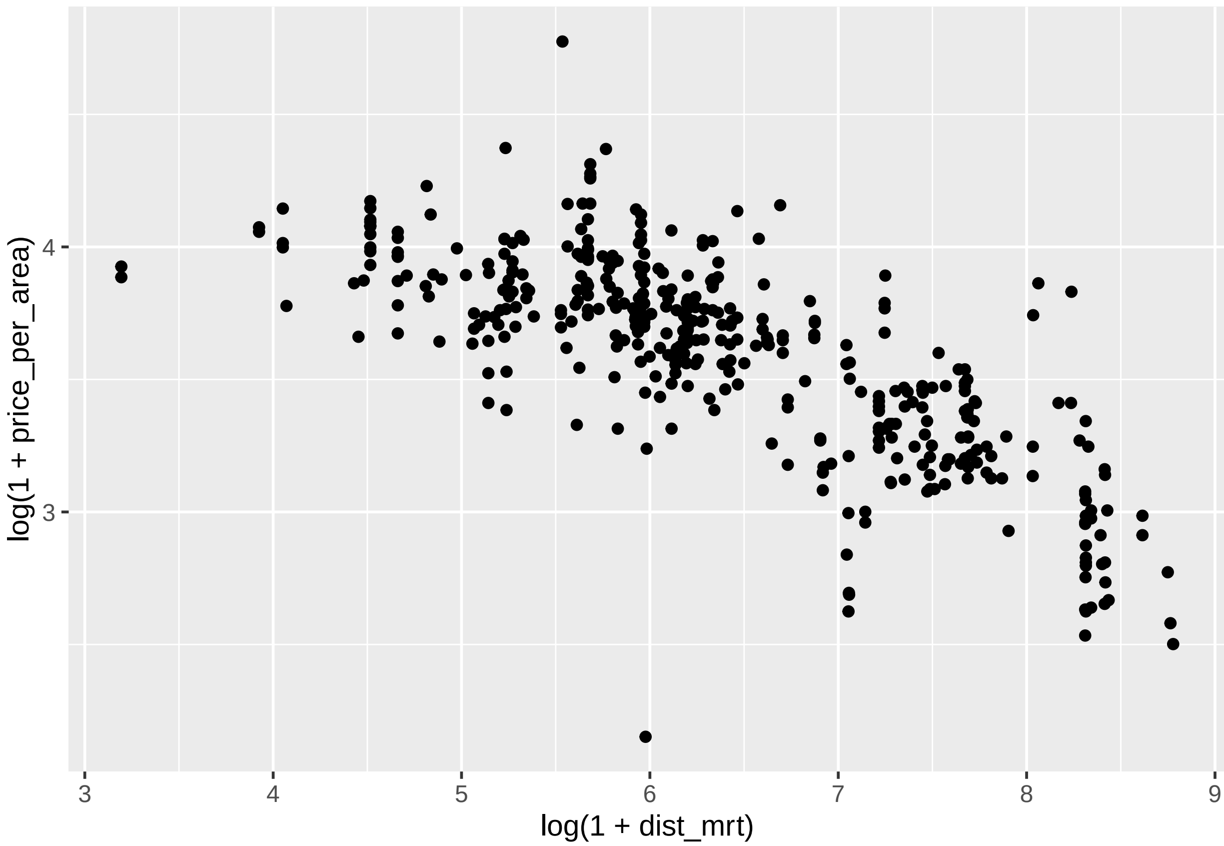


We see from the graphs confirmation of our previous conclusions from the pair plots. The data between MRT distance and price per area, and between house age and price per area seem clustered together rather than spread across the axes. We investigate if a log transformation will spread the data more effectively for regression analysis.

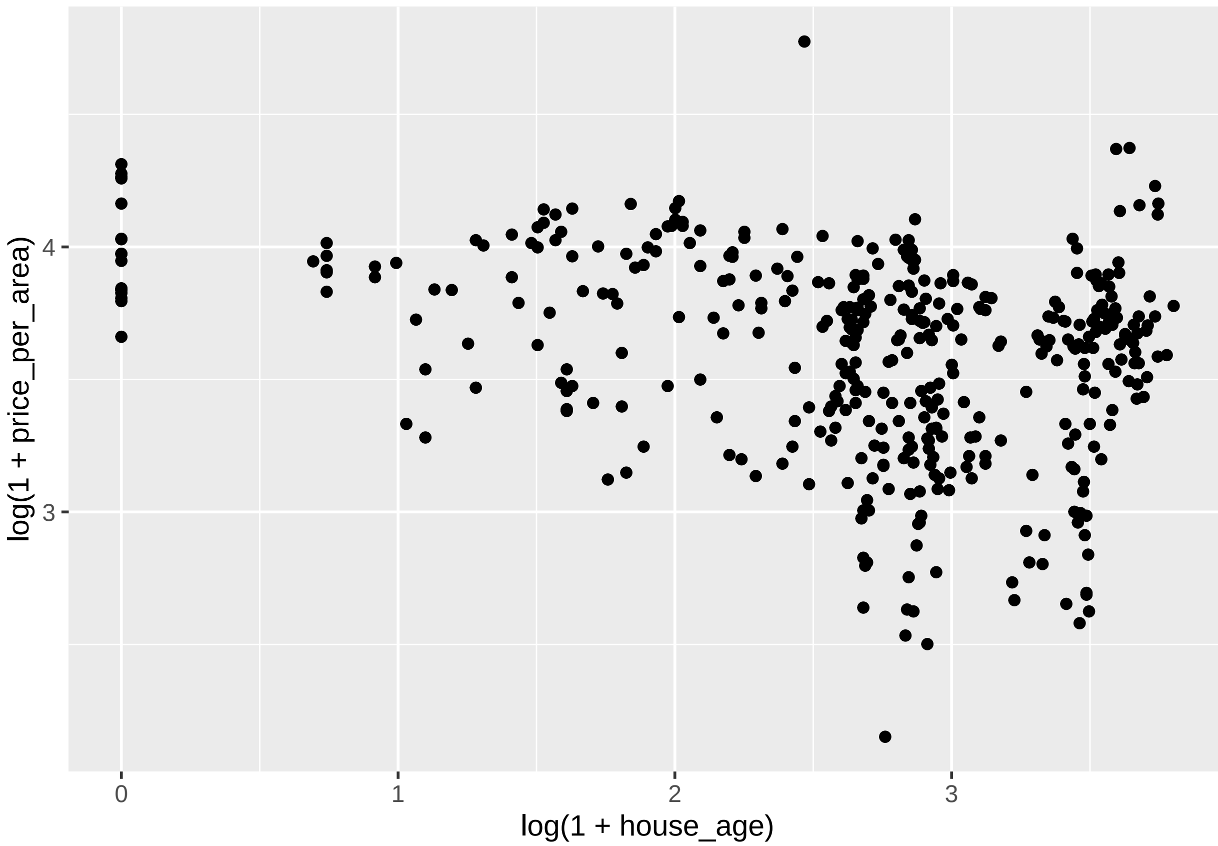
```
# Log Transforms

conv_plot = real_estate %>%
  ggplot(aes(x = num_conven, y = log(1+price_per_area))) +
  geom_point()

mrt_plot = real_estate %>%
  ggplot(aes(x = log(1+dist_mrt), y = log(1+price_per_area))) +
  geom_point()

houseage_plot = real_estate %>%
  ggplot(aes(x = log(1+house_age), y = log(1+price_per_area))) +
  geom_point()

conv_plot
```




```
mrt_plot
```

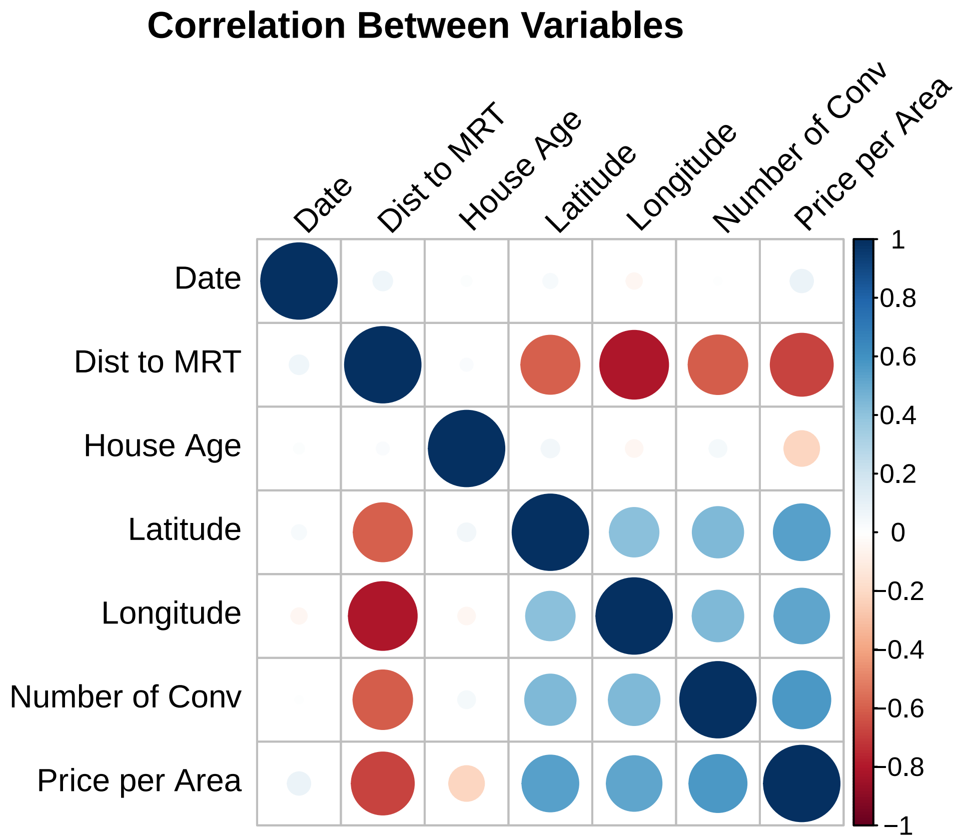



```
houseage_plot
```



The data seems be better distributed in the case of house age and price per area, as well as for MRT distance and price per area. The transformed number of convenience stores doesn't seem to be particularly changed relative to the untransformed version. We explore correlations between untransformed versions of all variables and between transformed versions of all variables.

```
corr = cor(real_estate %>%
             select(
               house_age,
               dist_mrt,
               num_conven,
               price_per_area,
               trans_date,
               lat,
               long
             ), method = "pearson")
colnames(corr) = c("House Age", "Dist to MRT", "Number of Conv", "Price per Area", "Date", "Latitude",
rownames(corr) = c("House Age", "Dist to MRT", "Number of Conv", "Price per Area", "Date", "Latitude",

corrplot(corr,
         type = "full",
         order = "alphabet",
         tl.col = "black",
         tl.srt = 45,
         title = "Correlation Between Variables",
         mar = c(0,0,2,0))
```

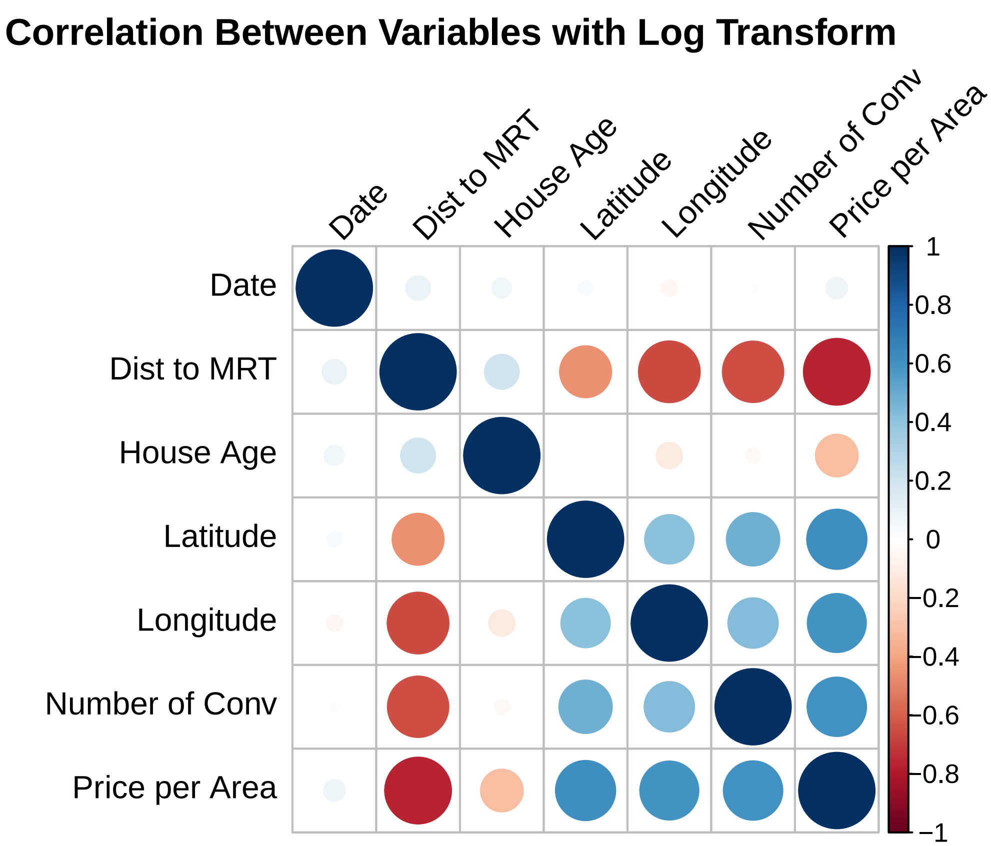



```
corr
```

```
##                   House Age Dist to MRT Number of Conv Price per Area
## House Age        1.00000000  0.02562205    0.049592513    -0.21056705
## Dist to MRT      0.02562205  1.00000000   -0.602519145    -0.67361286
## Number of Conv   0.04959251 -0.60251914    1.000000000     0.57100491
## Price per Area  -0.21056705 -0.67361286    0.571004911     1.00000000
## Date             0.01754234  0.06088009    0.009544199     0.08752927
## Latitude         0.05441990 -0.59106657    0.444143306     0.54630665
## Longitude       -0.04852005 -0.80631677    0.449099007     0.52328651
##                        Date    Latitude   Longitude
## House Age        0.017542341  0.05441990 -0.04852005
## Dist to MRT      0.060880095 -0.59106657 -0.80631677
## Number of Conv   0.009544199  0.44414331  0.44909901
## Price per Area   0.087529272  0.54630665  0.52328651
## Date             1.000000000  0.03501631 -0.04106508
## Latitude         0.035016305  1.00000000  0.41292394
## Longitude       -0.041065078  0.41292394  1.00000000
```

```
corr = cor(real_estate %>%
             select(
               house_age,
               dist_mrt,
               num_conven,
               price_per_area,
               trans_date,
               lat,
               long
```

```r
            ) %>%
            mutate(
              house_age = log(1 + house_age),
              dist_mrt = log(1 + dist_mrt),
              price_per_area = log(1 + price_per_area),
              num_conven = log(1 + num_conven),
              trans_date = log(1 + trans_date),
              lat = log(1 + lat),
              long = log(1 + long)
            ), method = "pearson")
colnames(corr) = c("House Age", "Dist to MRT", "Number of Conv", "Price per Area", "Date", "Latitude",
rownames(corr) = c("House Age", "Dist to MRT", "Number of Conv", "Price per Area", "Date", "Latitude",

corrplot(corr,
         type = "full",
         order = "alphabet",
         tl.col = "black",
         tl.srt = 45,
         title = "Correlation Between Variables with Log Transform",
         mar = c(0,0,2,0))
```




```r
corr
```

```
##                 House Age Dist to MRT Number of Conv Price per Area
## House Age       1.000000000  0.20310351    -0.03581629    -0.30733566
## Dist to MRT     0.203103512  1.00000000    -0.64310560    -0.76244894
## Number of Conv -0.035816287 -0.64310560     1.00000000     0.60105429
## Price per Area -0.307335662 -0.76244894     0.60105429     1.00000000
```

```
## Date            0.063509025  0.09752009     0.01236860     0.07580286
## Latitude       -0.003167461 -0.45757775     0.48348892     0.61704894
## Longitude      -0.113198529 -0.65027811     0.43046409     0.59334777
##                       Date     Latitude   Longitude
## House Age        0.06350903 -0.003167461 -0.11319853
## Dist to MRT      0.09752009 -0.457577750 -0.65027811
## Number of Conv   0.01236860  0.483488916  0.43046409
## Price per Area   0.07580286  0.617048943  0.59334777
## Date             1.00000000  0.034989221 -0.04106121
## Latitude         0.03498922  1.000000000  0.41304709
## Longitude       -0.04106121  0.413047095  1.00000000
```

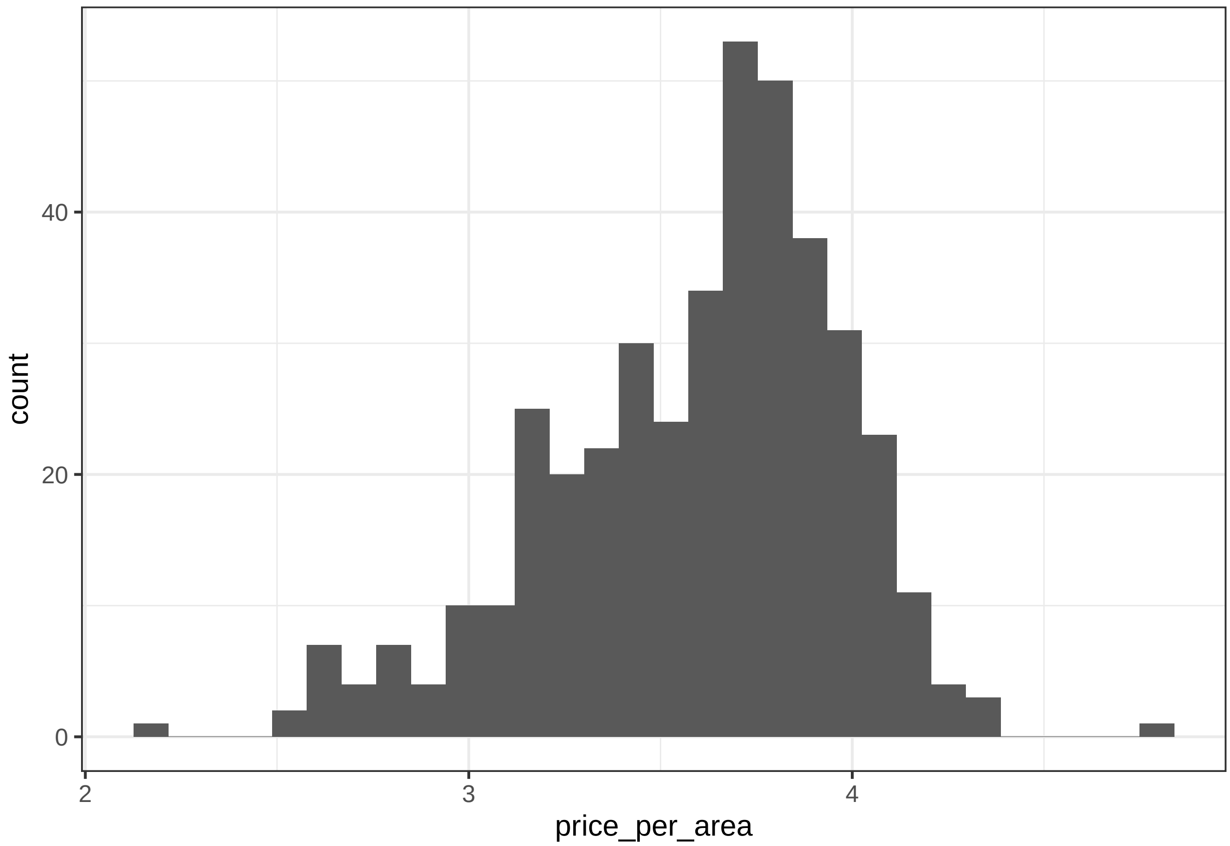
```r
real_estate = real_estate %>%
  mutate(
    house_age = log(1 + house_age),
    dist_mrt = log(1 + dist_mrt),
    price_per_area = log(1 + price_per_area),
    num_conven = log(1 + num_conven),
    lat = log(1 + lat),
    long = log(1 + long)
  )
```

We see that all of the correlations with the outcome variable grow stronger in magnitude. We also notice a slight increase in correlation between house age and MRT distance, but we will assume this is neglible at this stage. We therefore conclude that we will use the log-transformed versions of these variables for the analysis.

We look at the distribution of the log transformed price per area to check for outliers.

```r
real_estate %>%
  ggplot(aes(x = price_per_area)) +
  geom_histogram() +
  theme_bw()
```

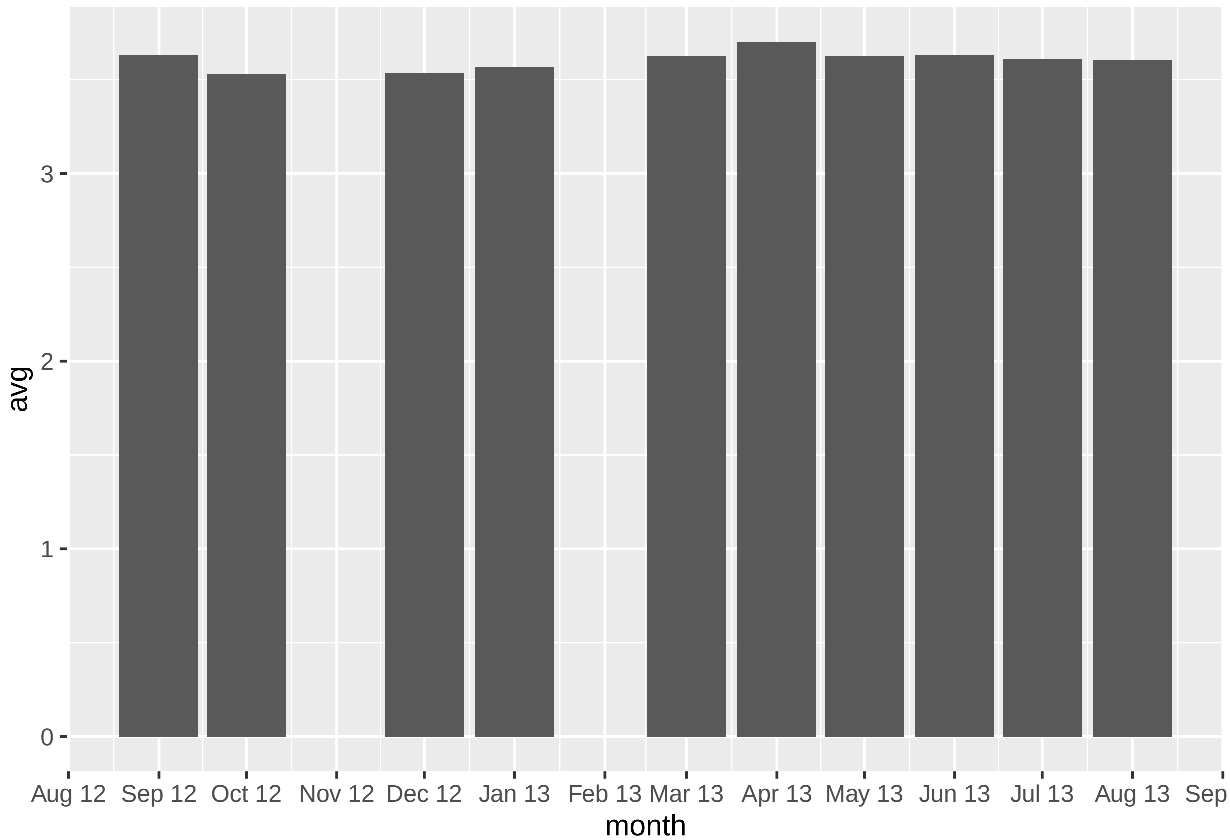
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We see what appears to be two significant outliers to the data. This may be cause for concern but we will revisit this later.

We next explore how data is distributed across time by looking at the average price by month-year in our dataset.

```
date_plot = real_estate %>%
  group_by(month = floor_date(date, unit = "month")) %>%
  summarise(avg = mean(price_per_area)) %>%
  ggplot(aes(x = month, y = avg)) +
  geom_bar(stat = "identity") +
  scale_x_datetime(date_labels = "%b %y", breaks = "month")

date_plot
```



We don't see any meaningful information here about prices across date and we notice two months have no data. From here, including the correlation results, we conclude that date is simply not a useful predictor for price.

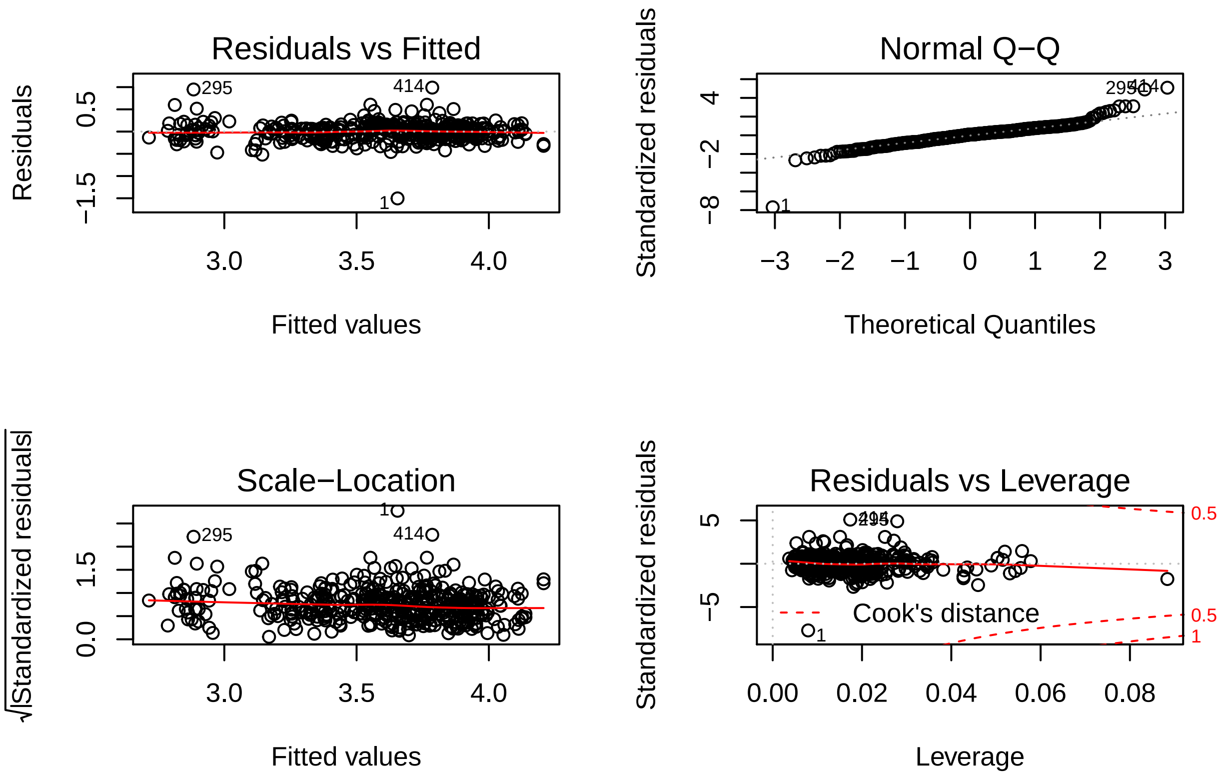
Sub-problem 2: multiple linear regression model (25 points)

```r
reg = lm(price_per_area ~ ., data = real_estate %>% select(-cluster, -date))
old.par <- par(mfrow=c(2,2))

summary(reg)
```

```
##
## Call:
## lm(formula = price_per_area ~ ., data = real_estate %>% select(-cluster,
##     -date))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.50309 -0.10727  0.01491  0.09771  0.98885
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.689e+03  4.912e+02  -5.474 7.70e-08 ***
## trans_date   1.688e-01  3.464e-02   4.873 1.58e-06 ***
## house_age   -8.856e-02  1.129e-02  -7.845 3.85e-14 ***
## dist_mrt    -1.545e-01  1.391e-02 -11.104  < 2e-16 ***
```

```
## num_conven   5.141e-02  1.725e-02   2.981  0.00305 **
## lat          2.455e+02  2.409e+01  10.192  < 2e-16 ***
## long         3.232e+02  1.030e+02   3.139  0.00182 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1961 on 407 degrees of freedom
## Multiple R-squared:  0.7365, Adjusted R-squared:  0.7326
## F-statistic: 189.6 on 6 and 407 DF,  p-value: < 2.2e-16
```

```
plot(reg)
```




```
par(old.par)
```

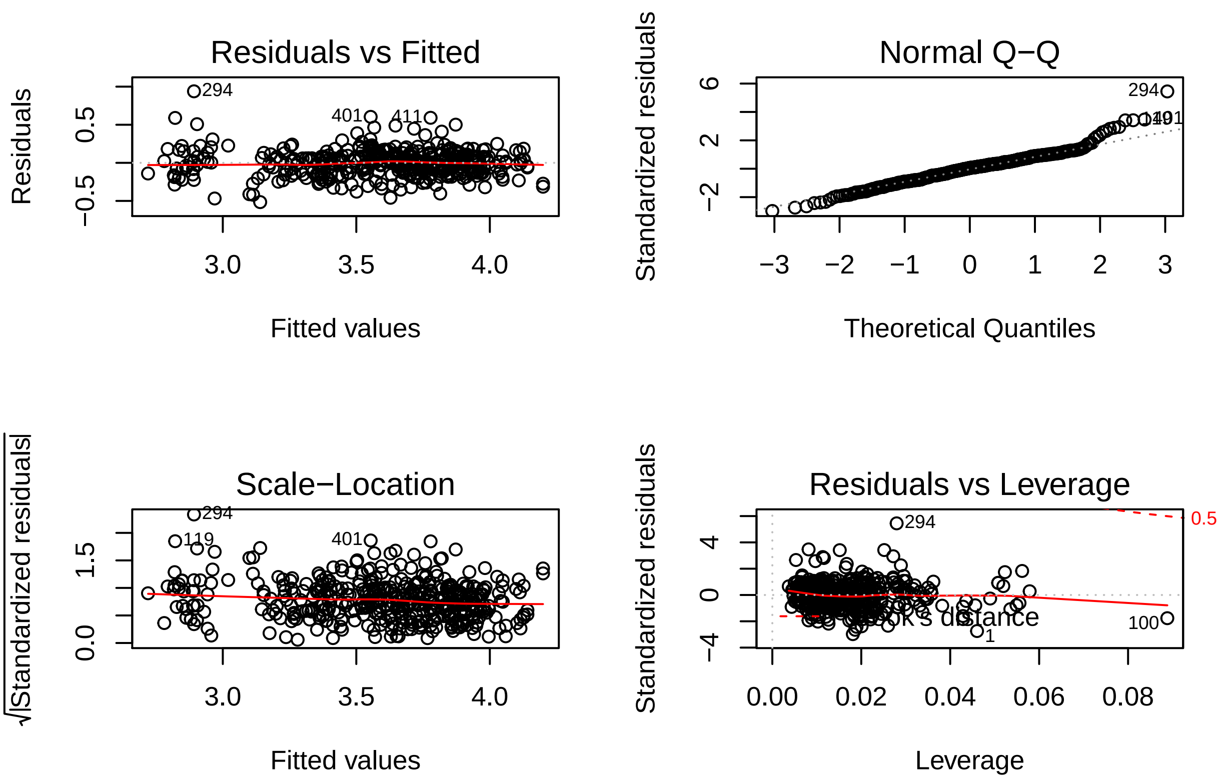
As suspected we see that the outliers identified earlier are causing issues with our regression model. The Normal Q-Q plot shows strong non-normality from the outliers, and the scale-location and residuals vs fitted both show large outlier effects. Outliers, generally, bias OLS regression models. We opt to omit these from the sample.

Model 1

```
real_estate = real_estate %>%
  slice(-1, -414)

reg = lm(price_per_area ~ ., data = real_estate %>% select(-cluster, -date))

old.par <- par(mfrow=c(2,2))

summary(reg)
```

```
##
## Call:
## lm(formula = price_per_area ~ ., data = real_estate %>% select(-cluster,
##     -date))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.51537 -0.11009  0.01016  0.09547  0.93873
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.816e+03  4.382e+02  -6.426 3.68e-10 ***
## trans_date   1.702e-01  3.093e-02   5.503 6.64e-08 ***
## house_age   -8.814e-02  1.006e-02  -8.758  < 2e-16 ***
## dist_mrt    -1.482e-01  1.246e-02 -11.887  < 2e-16 ***
## num_conven   6.573e-02  1.547e-02   4.250 2.66e-05 ***
## lat          2.305e+02  2.153e+01  10.708  < 2e-16 ***
## long         3.592e+02  9.191e+01   3.908 0.000109 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1748 on 405 degrees of freedom
## Multiple R-squared:  0.7787, Adjusted R-squared:  0.7754
## F-statistic: 237.5 on 6 and 405 DF,  p-value: < 2.2e-16
```

```r
plot(reg)
```




```r
par(old.par)
```

We see that removing these outliers improves the diagnostic plots, improves our adjusted r-squared, and improves the significance on two of our variables.

We next check for collinearity among predictors, get confidence intervals for predictor coefficients, and we find the prediction and 90% confidence interval for a "new" observation with explanatory variables set to the average of all the observations in our dataset.

```
vif(reg)
```

```
## trans_date  house_age   dist_mrt num_conven        lat       long 
##   1.026055   1.066330   2.612984   1.895512   1.428767   1.791137
```

```
confint(reg, level = 0.99)
```

```
##                     0.5 %        99.5 %
## (Intercept) -3.949949e+03 -1.681897e+03
## trans_date   9.014622e-02  2.502260e-01
## house_age   -1.141859e-01 -6.209249e-02
## dist_mrt    -1.804011e-01 -1.158945e-01
## num_conven   2.570227e-02  1.057667e-01
## lat          1.747789e+02  2.861935e+02
## long         1.213640e+02  5.971016e+02
```

```
means = real_estate %>%
  select(-cluster, -date, -price_per_area) %>%
  summarise_all(tibble::lst(mean))

colnames(means) = colnames(real_estate %>% select(-cluster, -date, -price_per_area))

avg_obs = predict(reg,newdata=means,interval='confidence',level = 0.9)
avg_obs
```

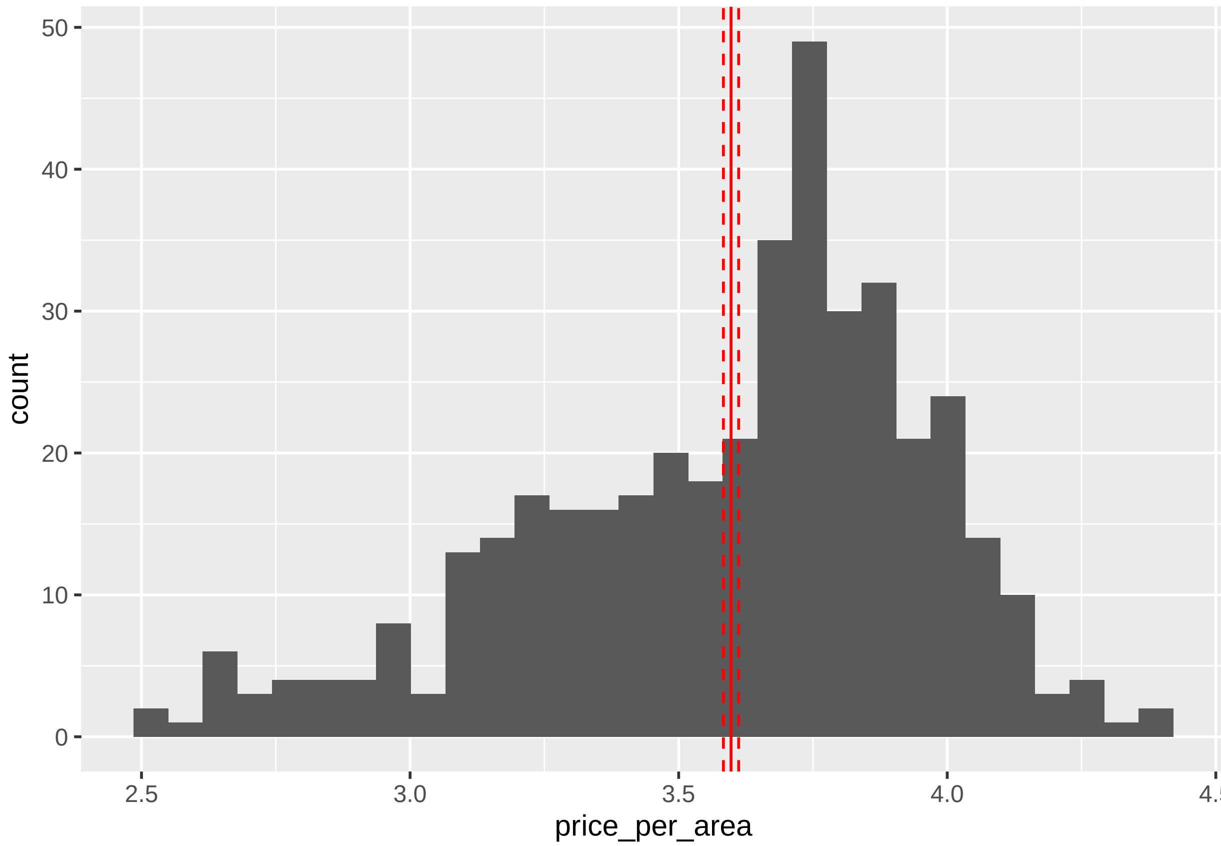
```
##        fit      lwr      upr
## 1 3.597518 3.583321 3.611715
```

Examining the results from the variance inflation factor calculation we see that the variables are all well below 5. The rule of thumb is that if the VIF exceeds 5 or 10 then this suggests a problem. As the calculated values do not cross this threshold we assume no problems in our model due to collinearity. When we look at the confidence intervals on the predictors we see that none of the confidence intervals include 0 which is to be expected in accordance with the significance levels identified previously.

We examine where the prediction for this average observation falls relative to the rest of our outcome data.

```
avg_plot = real_estate %>%
  ggplot(aes(x = price_per_area)) +
  geom_histogram() +
  geom_vline(xintercept = avg_obs[1], color = "red") +
  geom_vline(xintercept = avg_obs[2], color = "red", linetype = "dashed") +
  geom_vline(xintercept = avg_obs[3], color = "red", linetype = "dashed")

avg_plot
```

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



We see that the prediction for the average variable is around the average of the distribution of the outcome variable (`3.5975181`) which is what we would expect.

We also now explore replacing the latitude and longitude with our neighborhood clusters.

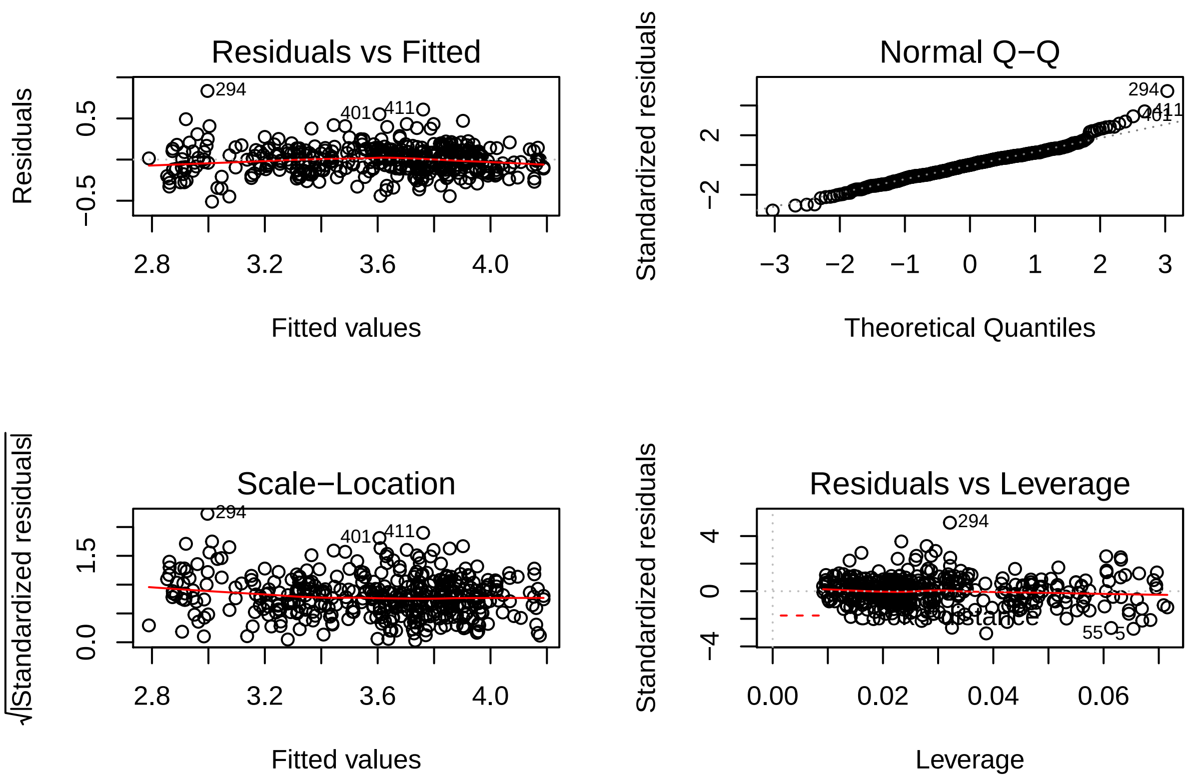
Model 2

```
reg = lm(price_per_area ~ ., data = real_estate %>% select(-lat, -long, -date))

old.par <- par(mfrow=c(2,2))

summary(reg)
```

```
##
## Call:
## lm(formula = price_per_area ~ ., data = real_estate %>% select(-lat,
##     -long, -date))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.51145 -0.10955  0.00116  0.09959  0.83409
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.398e+02  6.103e+01  -5.567 4.75e-08 ***
## trans_date   1.711e-01  3.032e-02   5.642 3.18e-08 ***
## house_age   -9.487e-02  1.017e-02  -9.325  < 2e-16 ***
## dist_mrt    -1.247e-01  1.640e-02  -7.607 2.03e-13 ***
```

```
## num_conven    1.076e-01  1.820e-02   5.909 7.38e-09 ***
## cluster2      2.469e-02  4.844e-02   0.510    0.611
## cluster3     -1.223e-01  2.487e-02  -4.915 1.30e-06 ***
## cluster4     -3.247e-01  3.973e-02  -8.174 3.99e-15 ***
## cluster5     -2.395e-01  4.934e-02  -4.854 1.74e-06 ***
## cluster6      6.519e-03  3.208e-02   0.203    0.839
## cluster7     -3.865e-01  5.346e-02  -7.230 2.47e-12 ***
## cluster8     -2.825e-01  4.037e-02  -6.997 1.10e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1706 on 400 degrees of freedom
## Multiple R-squared:  0.7917, Adjusted R-squared:  0.7859
## F-statistic: 138.2 on 11 and 400 DF,  p-value: < 2.2e-16
```

```r
plot(reg)
```




```r
par(old.par)
```

We find a slightly better adj r squared than with the latitude and longitude variables. We see that relative to cluster 1, clusters 2 and 6 seem to have insignificant effects, while clusters 4,7, and 8 have large negative effects. Cluster 3 has a weaker negative effect. Referring back to the graph, this follows a similar urban-rural divide with the exception that there seems to be two “types” of urban in clusters 1,2, and 6, and in cluster 3. We also explore briefly this separation.

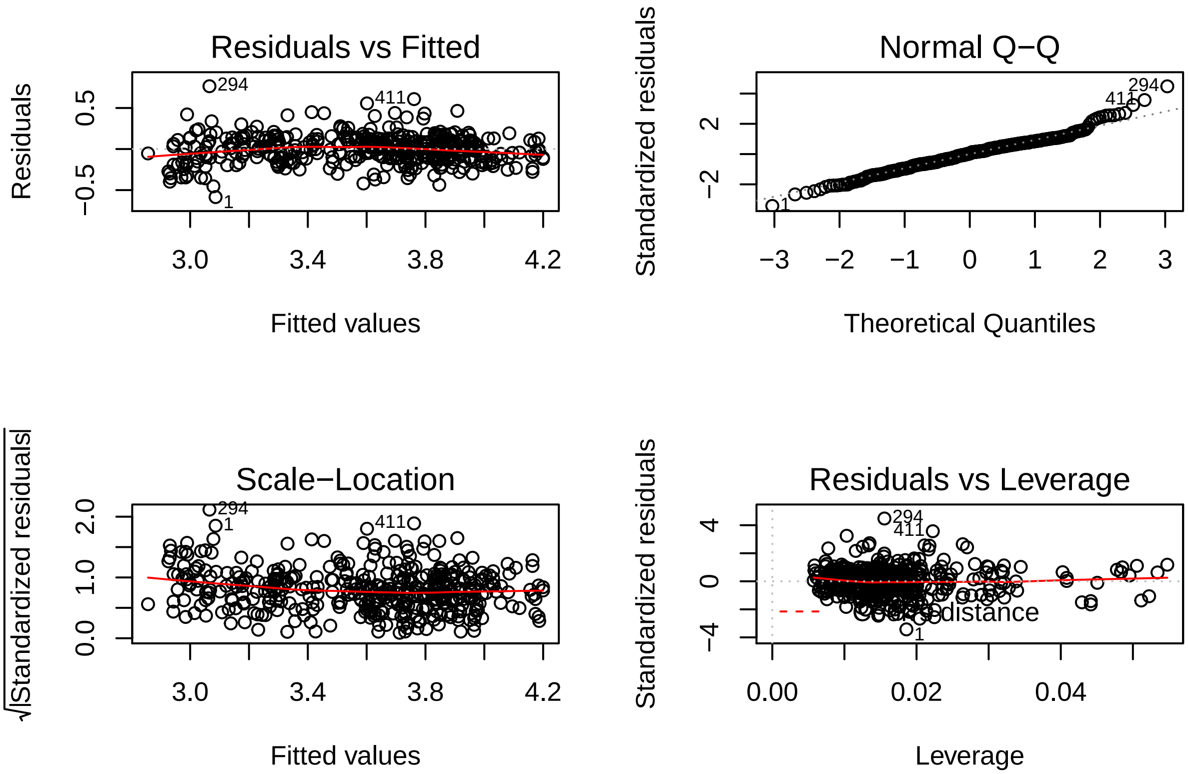
Model 3

```r
real_estate = real_estate %>%
  mutate(urban = recode(cluster, "c(1,2,6) = 1; c(4,5,7,8) = 0; c(3) = 2"))
```

```r
reg = lm(price_per_area ~ ., data = real_estate %>% select(-lat, -long, -date, -cluster))

old.par <- par(mfrow=c(2,2))

summary(reg)
```

```
##
## Call:
## lm(formula = price_per_area ~ ., data = real_estate %>% select(-lat,
##     -long, -date, -cluster))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5851 -0.1084  0.0083  0.1077  0.7652
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -338.75080   61.32621  -5.524 5.95e-08 ***
## trans_date     0.17041    0.03047   5.593 4.12e-08 ***
## house_age     -0.09853    0.01004  -9.814  < 2e-16 ***
## dist_mrt      -0.12459    0.01244 -10.016  < 2e-16 ***
## num_conven     0.11460    0.01472   7.783 5.95e-14 ***
## urban1         0.31046    0.02488  12.478  < 2e-16 ***
## urban2         0.18225    0.02999   6.076 2.84e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1722 on 405 degrees of freedom
## Multiple R-squared:  0.7851, Adjusted R-squared:  0.7819
## F-statistic: 246.6 on 6 and 405 DF,  p-value: < 2.2e-16
```

```r
plot(reg)
```




```
par(old.par)
```

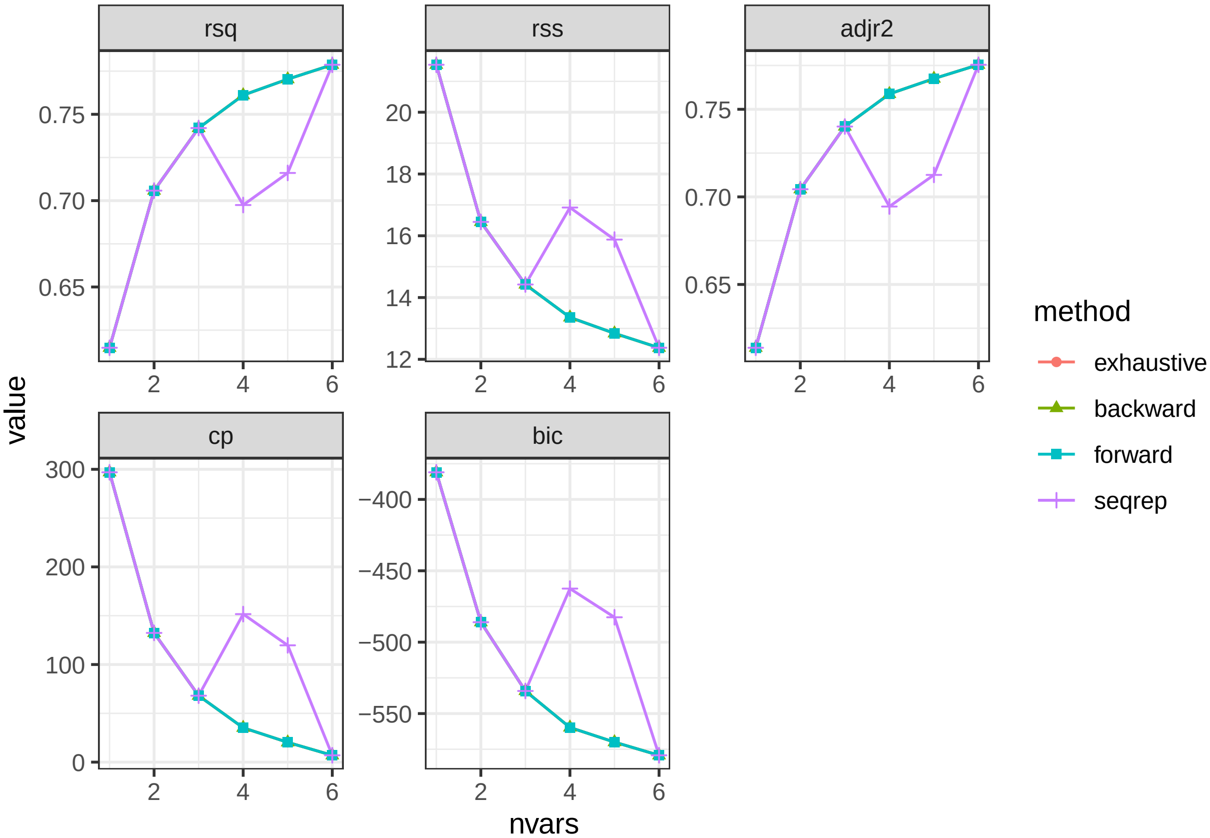
The adjusted r squared falls slightly but it is still higher than using the latitude and longitude variables.

Sub-problem 3: choose optimal models by exhaustive, forward and backward selection (20 points)

Model 1

```
target_data = real_estate %>%
  select(
    price_per_area,
    trans_date,
    house_age,
    dist_mrt,
    num_conven,
    long,
    lat
  )

summaryMetrics <- NULL
whichAll <- list()
for ( myMthd in c("exhaustive", "backward", "forward", "seqrep") ) {
  rsRes <- regsubsets(price_per_area~.,target_data,method=myMthd,nvmax=6)
  summRes <- summary(rsRes)
  whichAll[[myMthd]] <- summRes$which
  for ( metricName in c("rsq","rss","adjr2","cp","bic") ) {
    summaryMetrics <- rbind(summaryMetrics,
```

```
    data.frame(method=myMthd,metric=metricName,
               nvars=1:length(summRes[[metricName]]),
               value=summRes[[metricName]]))
  }
}
ggplot(summaryMetrics,aes(x=nvars,y=value,shape=method,colour=method)) + geom_path() + geom_point() + fa
```



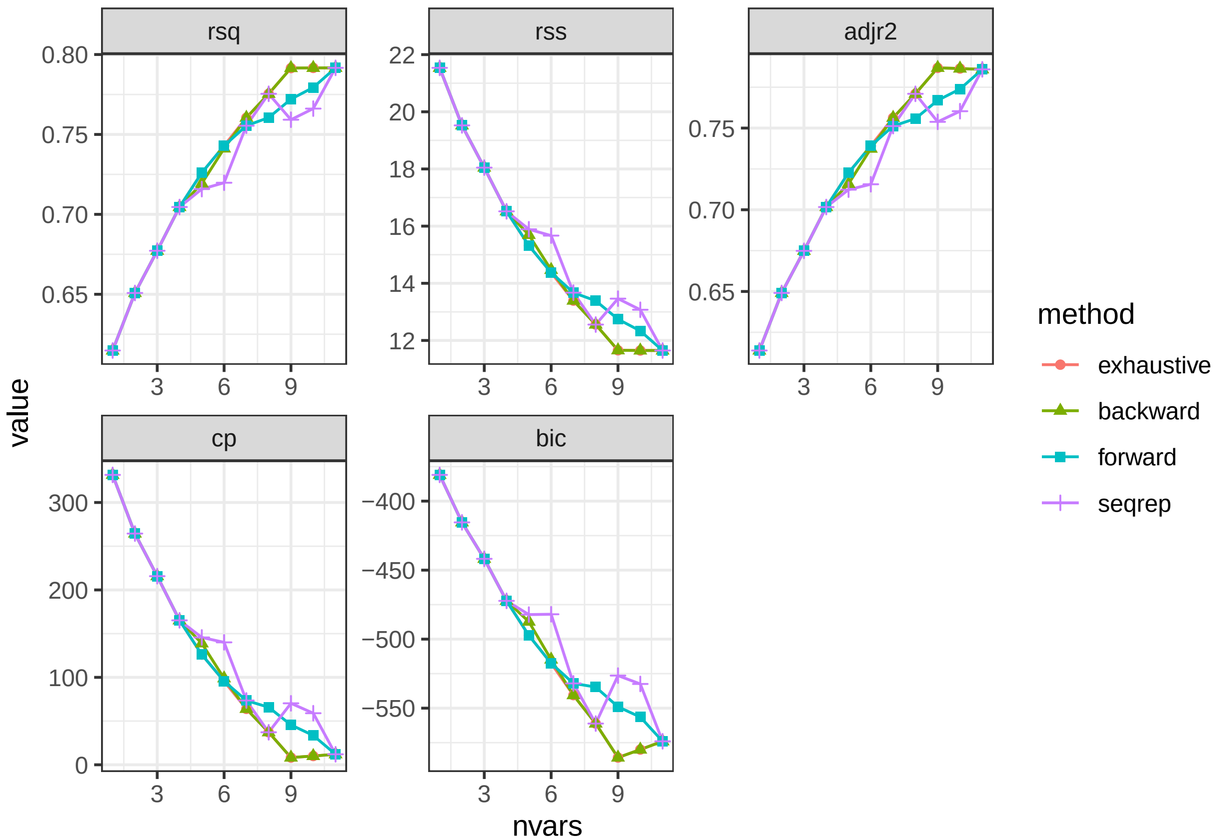
Apart from odd behavior on the seqrep method, we seen consistent model selection from the exhuastive, backward, and forward selection methods for the optimal model. We see that our primary metrics for model comparison (adjr2, cp, and bic) continue to improve as we add variables to the models suggesting that all variables are adding useful information.

We also do the same analysis for the cluster approach.

Model 2

```
target_data = real_estate %>%
  select(
    price_per_area,
    trans_date,
    house_age,
    dist_mrt,
    num_conven,
    cluster
  )

summaryMetrics <- NULL
```

```r
whichAll <- list()
for ( myMthd in c("exhaustive", "backward", "forward", "seqrep") ) {
  rsRes <- regsubsets(price_per_area~.,target_data,method=myMthd,nvmax=11)
  summRes <- summary(rsRes)
  whichAll[[myMthd]] <- summRes$which
  for ( metricName in c("rsq","rss","adjr2","cp","bic") ) {
    summaryMetrics <- rbind(summaryMetrics,
      data.frame(method=myMthd,metric=metricName,
                 nvars=1:length(summRes[[metricName]]),
                 value=summRes[[metricName]]))
  }
}
ggplot(summaryMetrics,aes(x=nvars,y=value,shape=method,colour=method)) + geom_path() + geom_point() + fa
```

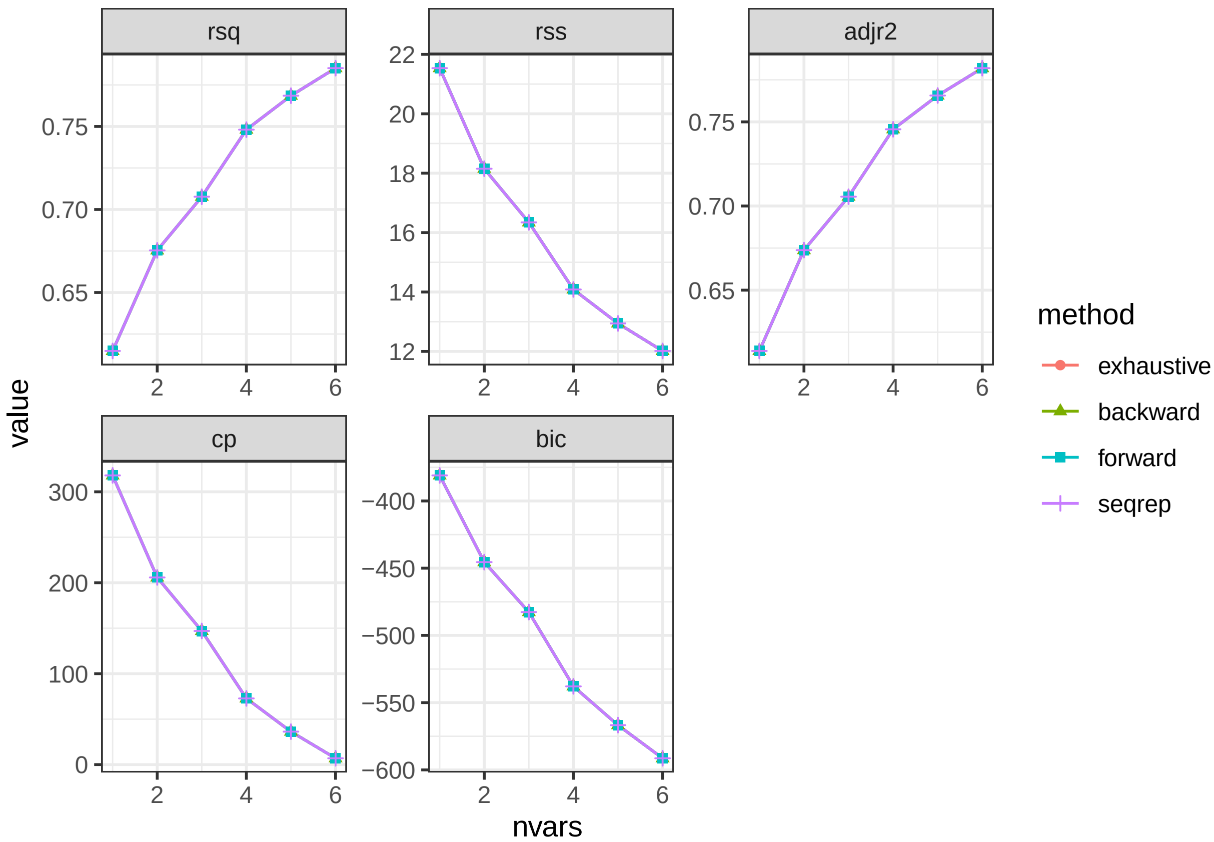


We see variations in the optimal model selection here between backward, forward, and seqrep here. The exhaustive selection method seems to follow the backward selection. We see that adjr2, cp, and bic all bottom out and suggest worse models from the addition of the 10th variable. This is likely corresponding to the 2 clusters without significance, the ones most similar to cluster 1. We re-do the analysis for the case where we use 2 urban clusters and 1 rural cluster.

Model 3

```r
target_data = real_estate %>%
  select(
    price_per_area,
    trans_date,
    house_age,
```

```
    dist_mrt,
    num_conven,
    urban
  )

summaryMetrics <- NULL
whichAll <- list()
for ( myMthd in c("exhaustive", "backward", "forward", "seqrep") ) {
  rsRes <- regsubsets(price_per_area~.,target_data,method=myMthd,nvmax=6)
  summRes <- summary(rsRes)
  whichAll[[myMthd]] <- summRes$which
  for ( metricName in c("rsq","rss","adjr2","cp","bic") ) {
    summaryMetrics <- rbind(summaryMetrics,
      data.frame(method=myMthd,metric=metricName,
                 nvars=1:length(summRes[[metricName]]),
                 value=summRes[[metricName]]))
  }
}
ggplot(summaryMetrics,aes(x=nvars,y=value,shape=method,colour=method)) + geom_path() + geom_point() + fa
```

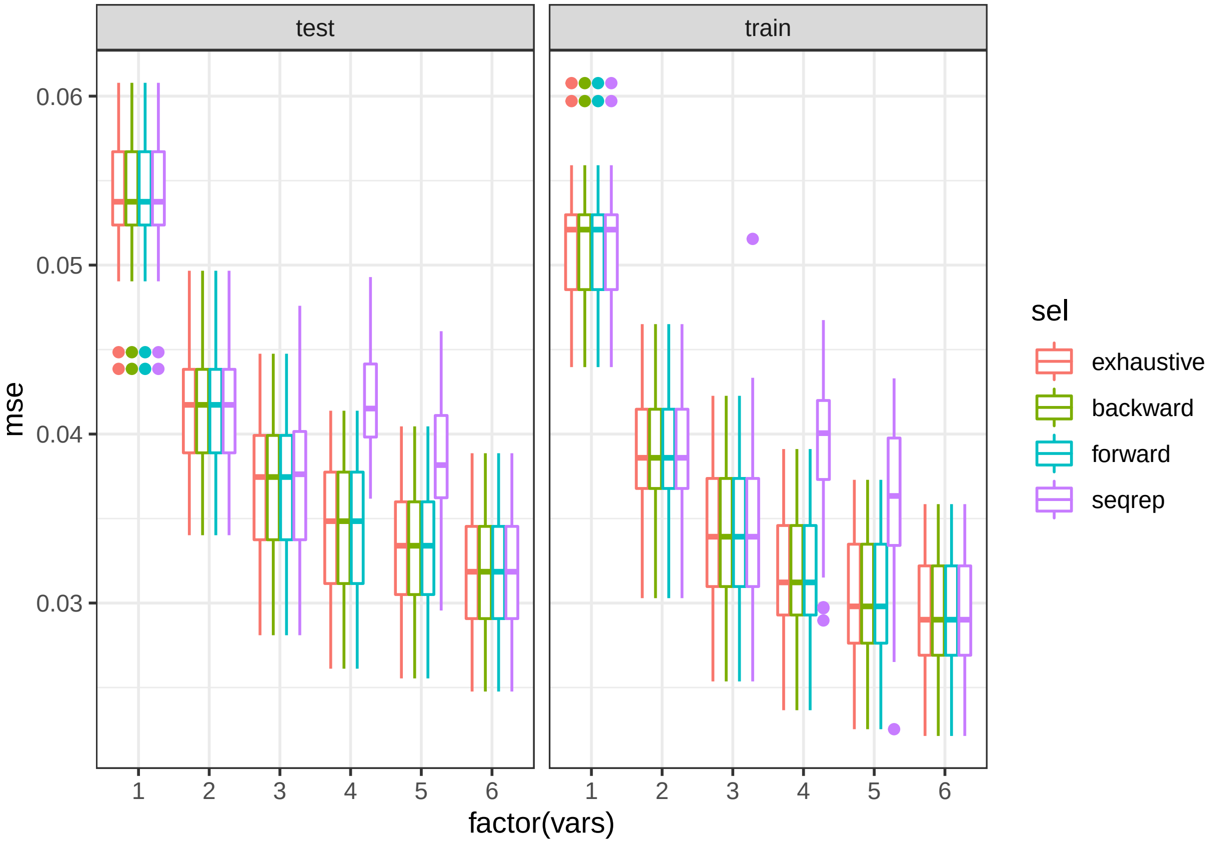


The model with 2 urban clusters and 1 rural cluster seems to do the best of all 3. All 4 selection methods agree, and our evaluation metrics are better than the other two variable subsets.

Sub-problem 4: optimal model by resampling (20 points)

Model 1

```
predict.regsubsets <- function (object, newdata, id, ...){
  form=as.formula(object$call [[2]])
  mat=model.matrix(form,newdata)
  coefi=coef(object,id=id)
  xvars=names (coefi)
  mat[,xvars] %*% coefi
}

target_data = real_estate %>%
  select(
    price_per_area,
    trans_date,
    house_age,
    dist_mrt,
    num_conven,
    long,
    lat
  )

dfTmp <- NULL
whichSum <- array(0,dim=c(6,7,4),
  dimnames=list(NULL,colnames(model.matrix(price_per_area~.,target_data)),
      c("exhaustive", "backward", "forward", "seqrep")))
# Split data into training and test 30 times:
nTries <- 30
for ( iTry in 1:nTries ) {
  bTrain <- sample(rep(c(TRUE,FALSE),length.out=nrow(target_data)))
  # Try each method available in regsubsets
  # to select the best model of each size:
  for ( jSelect in c("exhaustive", "backward", "forward", "seqrep") ) {
    rsTrain <- regsubsets(price_per_area~.,target_data[bTrain,],nvmax=6,method=jSelect)
    # Add up variable selections:
    whichSum[,,jSelect] <- whichSum[,,jSelect] + summary(rsTrain)$which
    # Calculate test error for each set of variables
    # using predict.regsubsets implemented above:
    for ( kVarSet in 1:6 ) {
      # make predictions:
      testPred <- predict(rsTrain,target_data[!bTrain,],id=kVarSet)
      # calculate MSE:
      mseTest <- mean((testPred-target_data[!bTrain,"price_per_area"])^2 %$% price_per_area)
      # add to data.frame for future plotting:
      dfTmp <- rbind(dfTmp,data.frame(sim=iTry,sel=jSelect,vars=kVarSet,
      mse=c(mseTest,summary(rsTrain)$rss[kVarSet]/sum(bTrain)),trainTest=c("test","train")))
    }
  }
}
# plot MSEs by training/test, number of
# variables and selection method:
ggplot(dfTmp,aes(x=factor(vars),y=mse,colour=sel)) + geom_boxplot()+facet_wrap(~trainTest)+theme_bw()
```

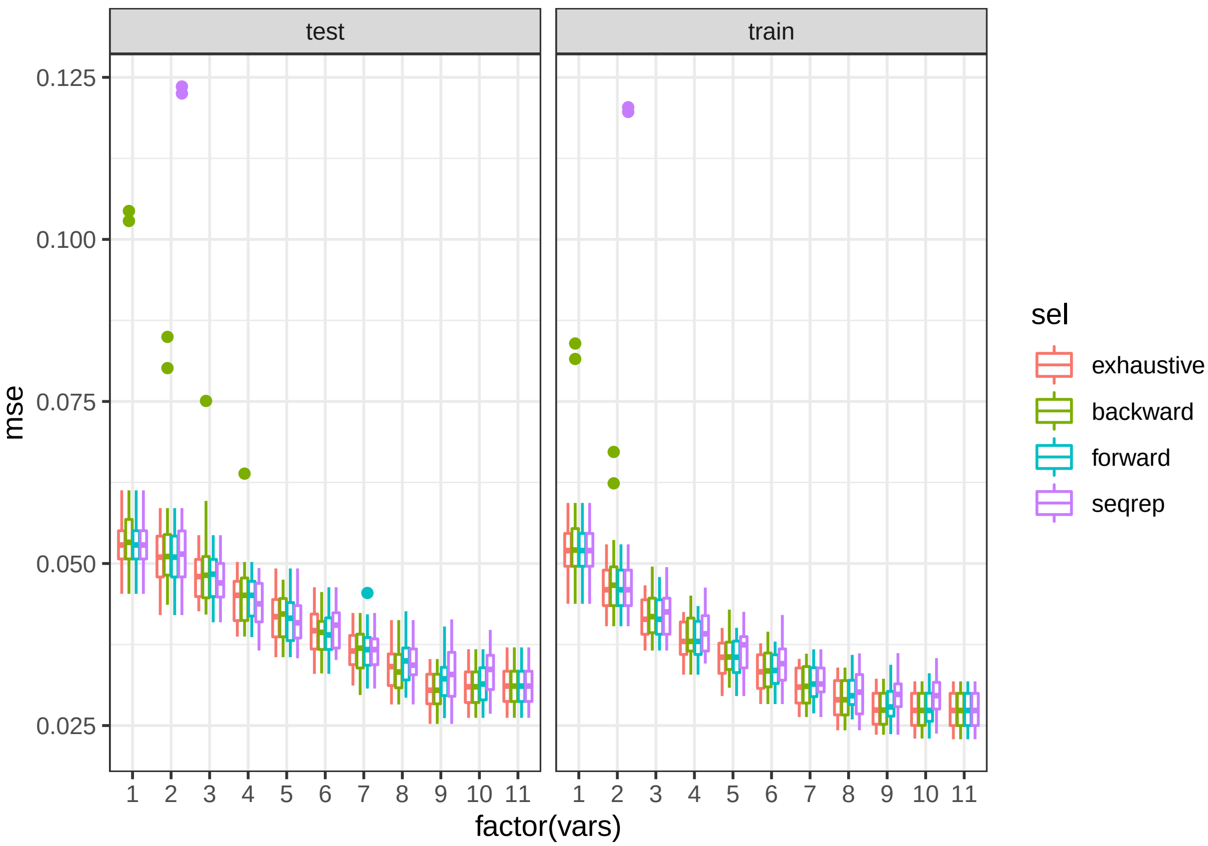


Model 2

In line with what we saw in the model selection process using regsubsets, the model with all variables does the best on the test and train data, so we do not have an overfitting problem. What's more, the difference in the errors in quite small so we are doing quite well in cross-validating the model. We now repeat with the two clustering cases.

```
target_data = real_estate %>%
  select(
    price_per_area,
    trans_date,
    house_age,
    dist_mrt,
    num_conven,
    cluster
  )

dfTmp <- NULL
whichSum <- array(0,dim=c(11,12,4),
  dimnames=list(NULL,colnames(model.matrix(price_per_area~.,target_data)),
      c("exhaustive", "backward", "forward", "seqrep")))
# Split data into training and test 30 times:
nTries <- 30
for ( iTry in 1:nTries ) {
  bTrain <- sample(rep(c(TRUE,FALSE),length.out=nrow(target_data)))
  # Try each method available in regsubsets
  # to select the best model of each size:
```

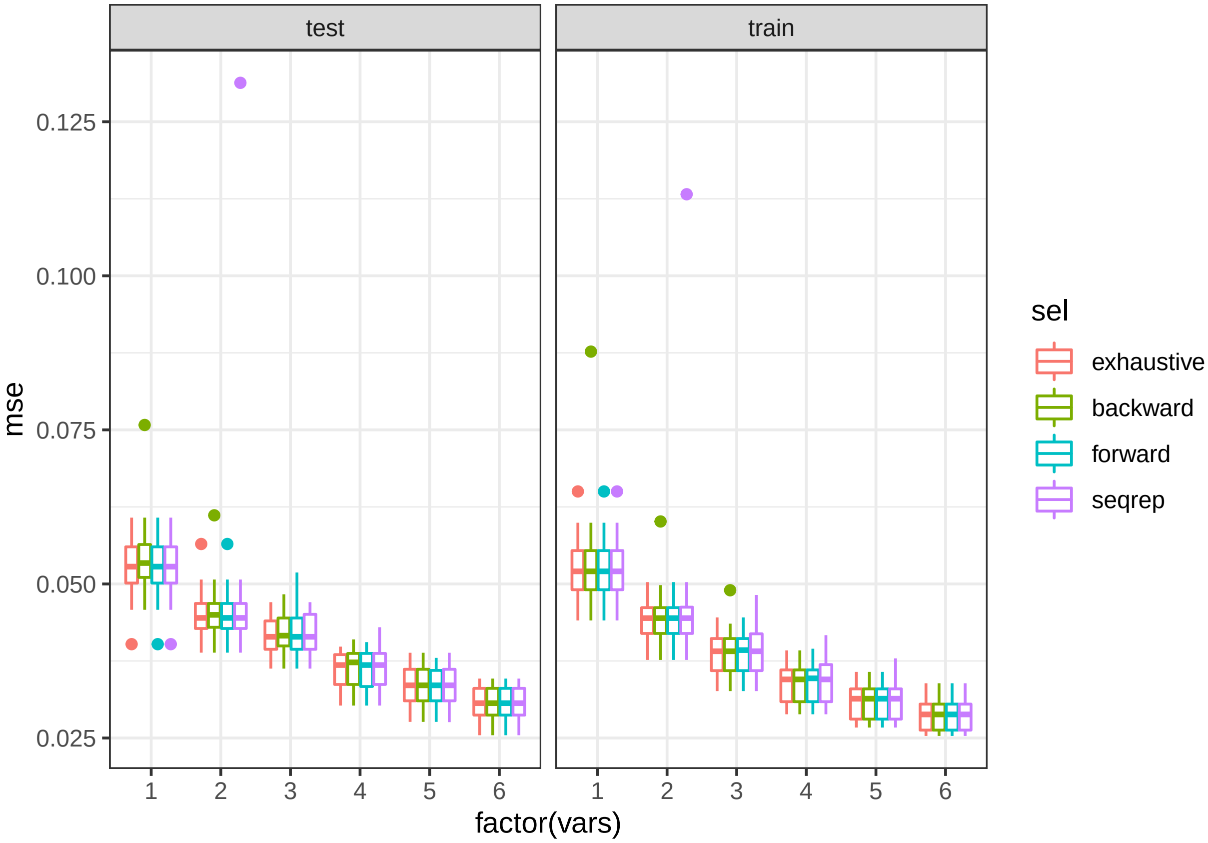
```r
  for ( jSelect in c("exhaustive", "backward", "forward", "seqrep") ) {
    rsTrain <- regsubsets(price_per_area~.,target_data[bTrain,],nvmax=11,method=jSelect)
    # Add up variable selections:
    whichSum[,,jSelect] <- whichSum[,,jSelect] + summary(rsTrain)$which
    # Calculate test error for each set of variables
    # using predict.regsubsets implemented above:
    for ( kVarSet in 1:11 ) {
      # make predictions:
      testPred <- predict(rsTrain,target_data[!bTrain,],id=kVarSet)
      # calculate MSE:
      mseTest <- mean((testPred-target_data[!bTrain,"price_per_area"])^2 %$% price_per_area)
      # add to data.frame for future plotting:
      dfTmp <- rbind(dfTmp,data.frame(sim=iTry,sel=jSelect,vars=kVarSet,
      mse=c(mseTest,summary(rsTrain)$rss[kVarSet]/sum(bTrain)),trainTest=c("test","train")))
    }
  }
}
# plot MSEs by training/test, number of
# variables and selection method:
ggplot(dfTmp,aes(x=factor(vars),y=mse,colour=sel)) + geom_boxplot()+facet_wrap(~trainTest)+theme_bw()
```



The cross-validation performance on this model is not so great. There are a lot of outliers and the minimum error on the test set seems to be around the 0.03 mark of the previous model's best error. There also seems to be a little bit of overfitting on the models with 10 and 11 variables as was suggested in the previous adjr2, cp, and bic analysis.

Model 3

```r
target_data = real_estate %>%
  select(
    price_per_area,
    trans_date,
    house_age,
    dist_mrt,
    num_conven,
    urban
  )

dfTmp <- NULL
whichSum <- array(0,dim=c(6,7,4),
  dimnames=list(NULL,colnames(model.matrix(price_per_area~.,target_data)),
      c("exhaustive", "backward", "forward", "seqrep")))
# Split data into training and test 30 times:
nTries <- 30
for ( iTry in 1:nTries ) {
  bTrain <- sample(rep(c(TRUE,FALSE),length.out=nrow(target_data)))
  # Try each method available in regsubsets
  # to select the best model of each size:
  for ( jSelect in c("exhaustive", "backward", "forward", "seqrep") ) {
    rsTrain <- regsubsets(price_per_area~.,target_data[bTrain,],nvmax=6,method=jSelect)
    # Add up variable selections:
    whichSum[,,jSelect] <- whichSum[,,jSelect] + summary(rsTrain)$which
    # Calculate test error for each set of variables
    # using predict.regsubsets implemented above:
    for ( kVarSet in 1:6 ) {
      # make predictions:
      testPred <- predict(rsTrain,target_data[!bTrain,],id=kVarSet)
      # calculate MSE:
      mseTest <- mean((testPred-target_data[!bTrain,"price_per_area"])^2 %$% price_per_area)
      # add to data.frame for future plotting:
      dfTmp <- rbind(dfTmp,data.frame(sim=iTry,sel=jSelect,vars=kVarSet,
      mse=c(mseTest,summary(rsTrain)$rss[kVarSet]/sum(bTrain)),trainTest=c("test","train")))
    }
  }
}
# plot MSEs by training/test, number of
# variables and selection method:
ggplot(dfTmp,aes(x=factor(vars),y=mse,colour=sel)) + geom_boxplot()+facet_wrap(~trainTest)+theme_bw()
```



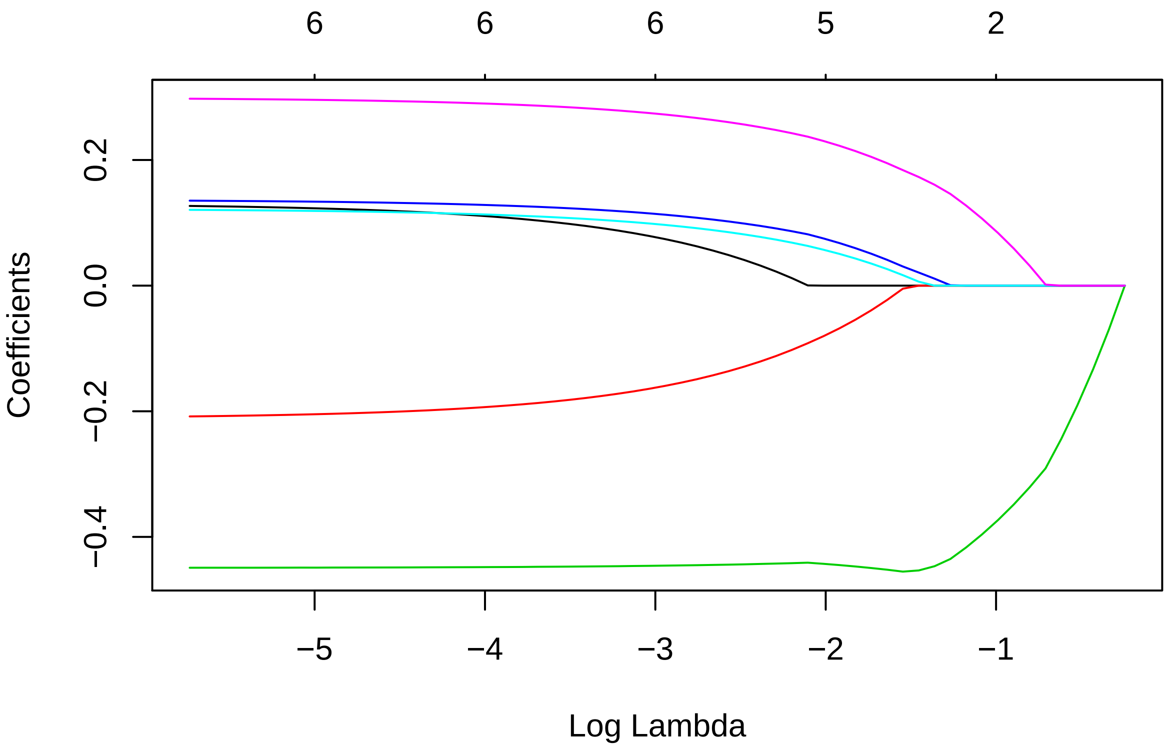
This variable subset does fairly well. It has a lower training error than the first model with latitude and longitude but a similar (almost identical) minimum test error. The results here agree with the previous adjr2, bic, cp, analysis but disagree somewhat on what is suggested by the test error. The previous analysis suggested this model might be better than the latitude and longitude model but this analysis suggests that the two models are about equivalent for this dataset.

Sub-problem 5: variable selection by lasso (15 points)

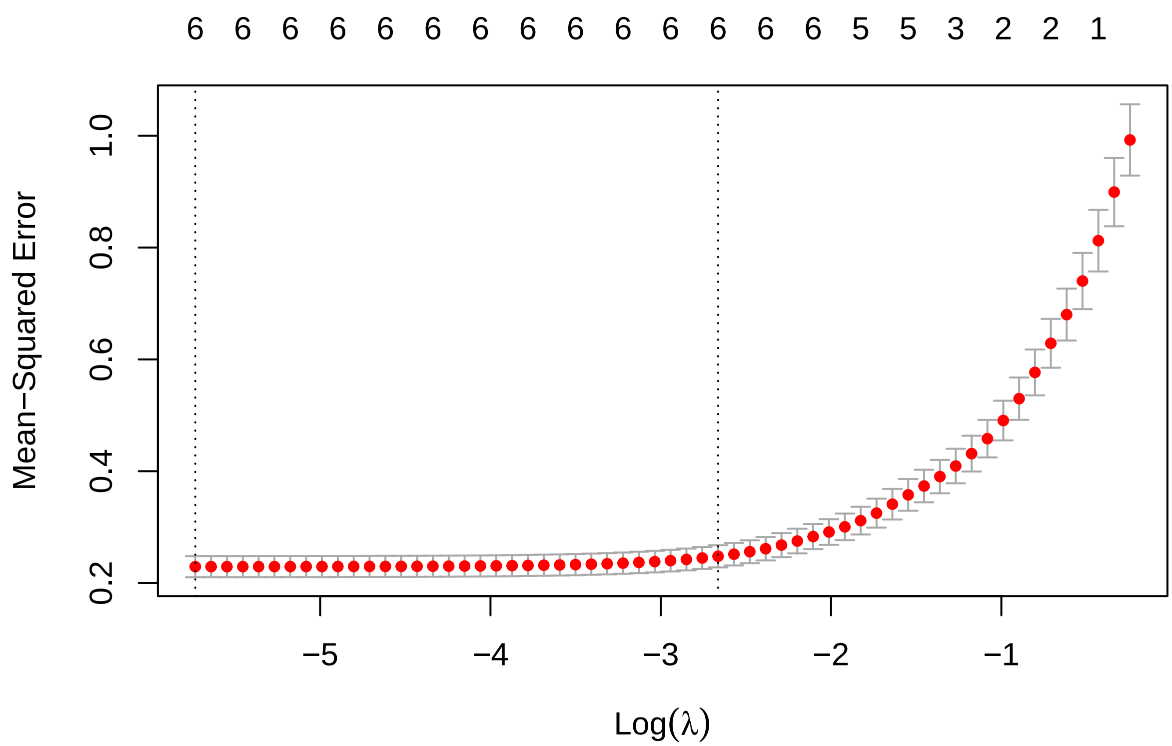
Model 1

```
target_data = real_estate %>%
  select(
    price_per_area,
    trans_date,
    house_age,
    dist_mrt,
    num_conven,
    long,
    lat
  ) %>%
  mutate_all(scale)

x <- model.matrix(price_per_area~.,target_data)[,-1]
y = target_data$price_per_area

lassoRes <- glmnet(x,y,alpha=1)
```

```r
plot(lassoRes, xvar = "lambda")
```




```r
cvLassoRes <- cv.glmnet(x,y,alpha=1)
plot(cvLassoRes)
```




```r
predict(lassoRes,type="coefficients",s=cvLassoRes$lambda.min)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) -3.304028e-13
## trans_date   1.268612e-01
## house_age   -2.081912e-01
## dist_mrt    -4.490818e-01
## num_conven   1.352608e-01
## long         1.206391e-01
## lat          2.975428e-01
```

```r
predict(lassoRes,type="coefficients",s=cvLassoRes$lambda.1se)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) -2.086305e-13
## trans_date   5.593546e-02
## house_age   -1.428667e-01
## dist_mrt    -4.445972e-01
## num_conven   1.052264e-01
## long         8.840268e-02
## lat          2.636808e-01
```

```r
predict(lassoRes,type="coefficients",s=0.4)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)  2.997108e-14
## trans_date   .
## house_age    .
## dist_mrt    -3.539910e-01
## num_conven   .
## long         .
## lat          6.466845e-02
```

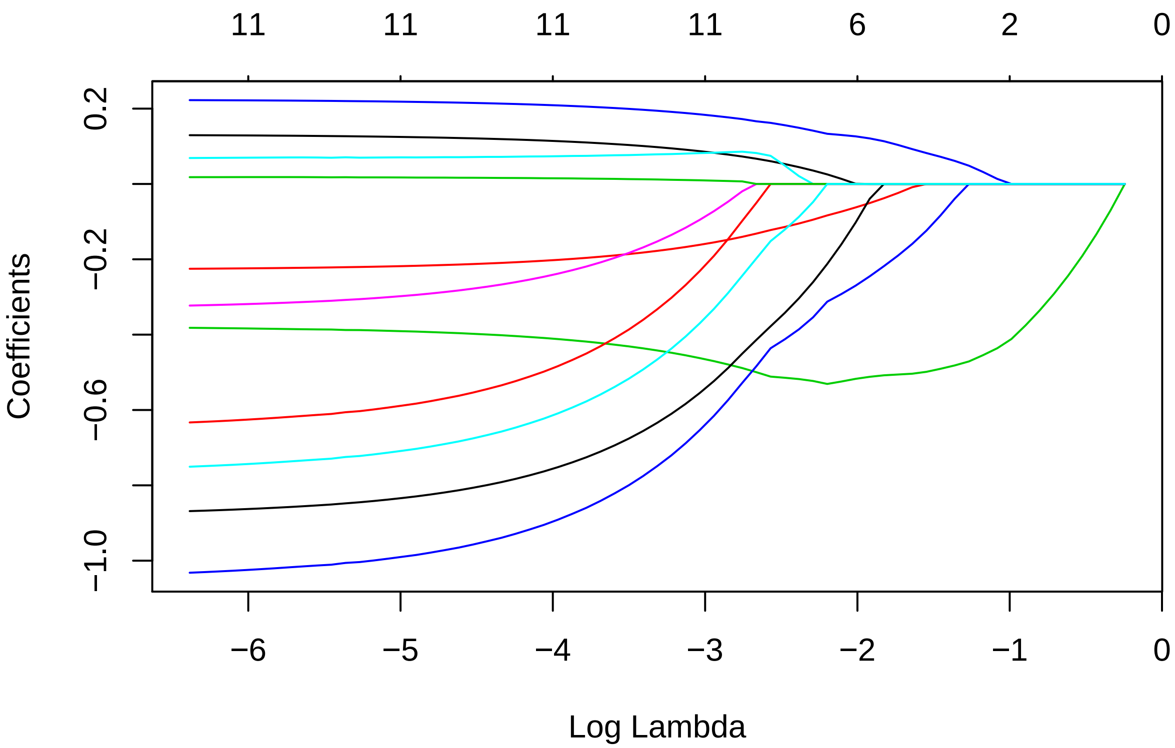
```r
predict(lassoRes,type="coefficients",s=0.25)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)  6.934142e-14
## trans_date   .
## house_age    .
## dist_mrt    -4.485366e-01
## num_conven   1.382731e-02
## long         1.785177e-03
## lat          1.640823e-01
```

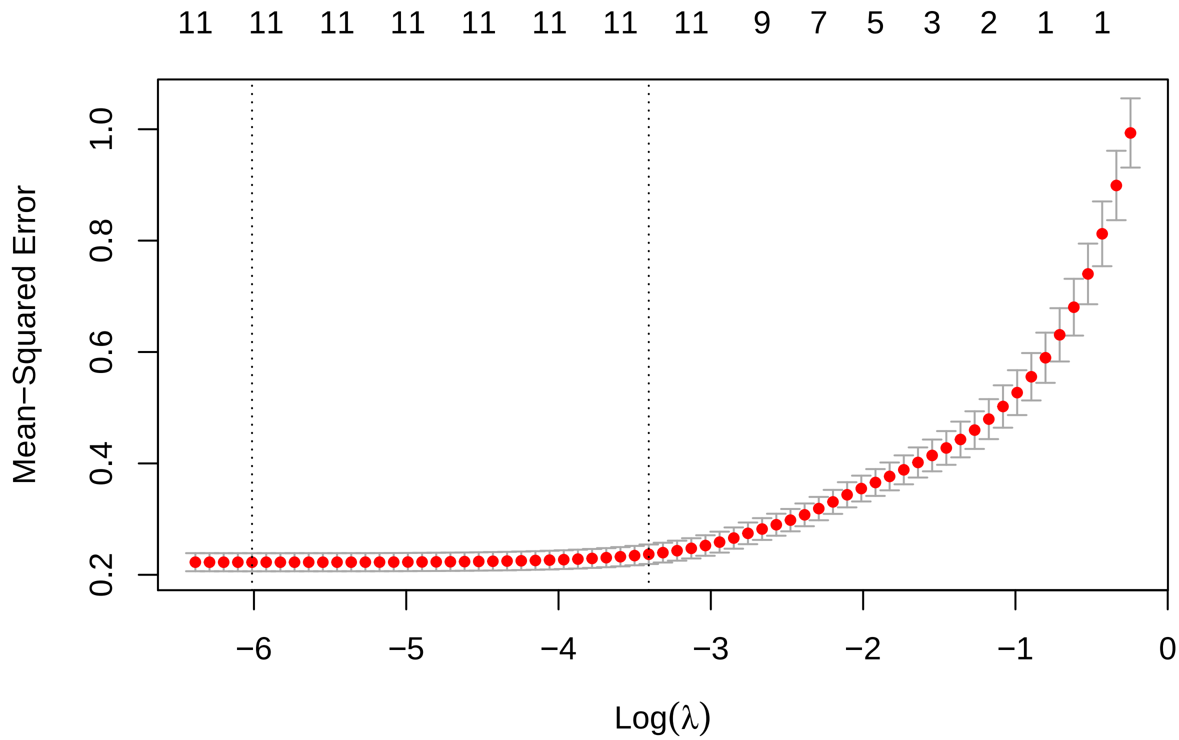
To do the lasso analysis we scale the variables so that the coefficients are more evenly weighted in the model. When we examine the lasso selection on the variables we see confirmation as before in the regsubsets and resampling validation that the lowest cross-validation error occurs with all 6 variables included in the model. The lasso model selects two variables as being very important within the first order of magnitude of lambda. These variables are the distance from the MRT station and the latitude of the house. We see that the trans_date variable is not picked up in the first 4 most important variables by the lasso regularization, which is in line with the low predictive power we saw when exploring the variable.

Model 2

```
target_data = real_estate %>%
  select(
    price_per_area,
    trans_date,
    house_age,
    dist_mrt,
    num_conven
  ) %>%
  mutate_all(scale) %>%
  mutate(cluster = real_estate$cluster)

x <- model.matrix(price_per_area~.,target_data)[,-1]
y = target_data$price_per_area

lassoRes <- glmnet(x,y,alpha=1)
plot(lassoRes, xvar = "lambda")
```




```
cvLassoRes <- cv.glmnet(x,y,alpha=1)
plot(cvLassoRes)
```




```r
predict(lassoRes,type="coefficients",s=cvLassoRes$lambda.min)
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  0.32685497
## trans_date   0.12881862
## house_age   -0.22401614
## dist_mrt    -0.38351772
## num_conven   0.22198204
## cluster2     0.06976986
## cluster3    -0.31889226
## cluster4    -0.86308760
## cluster5    -0.62573897
## cluster6     0.01823482
## cluster7    -1.02478831
## cluster8    -0.74363388
```

```r
predict(lassoRes,type="coefficients",s=cvLassoRes$lambda.1se)
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  0.21666799
## trans_date   0.10072994
## house_age   -0.18200343
## dist_mrt    -0.43632658
## num_conven   0.19699746
## cluster2     0.07767134
## cluster3    -0.16850372
## cluster4    -0.65587925
## cluster5    -0.36073836
## cluster6     0.01243654
```

```
## cluster7    -0.77546572
## cluster8    -0.49257182
```

```
predict(lassoRes,type="coefficients",s=0.2)
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  0.012919553
## trans_date   .
## house_age   -0.005824572
## dist_mrt    -0.502069698
## num_conven   0.089208210
## cluster2     .
## cluster3     .
## cluster4     .
## cluster5     .
## cluster6     .
## cluster7    -0.147857111
## cluster8     .
```

```
predict(lassoRes,type="coefficients",s=0.24)
```

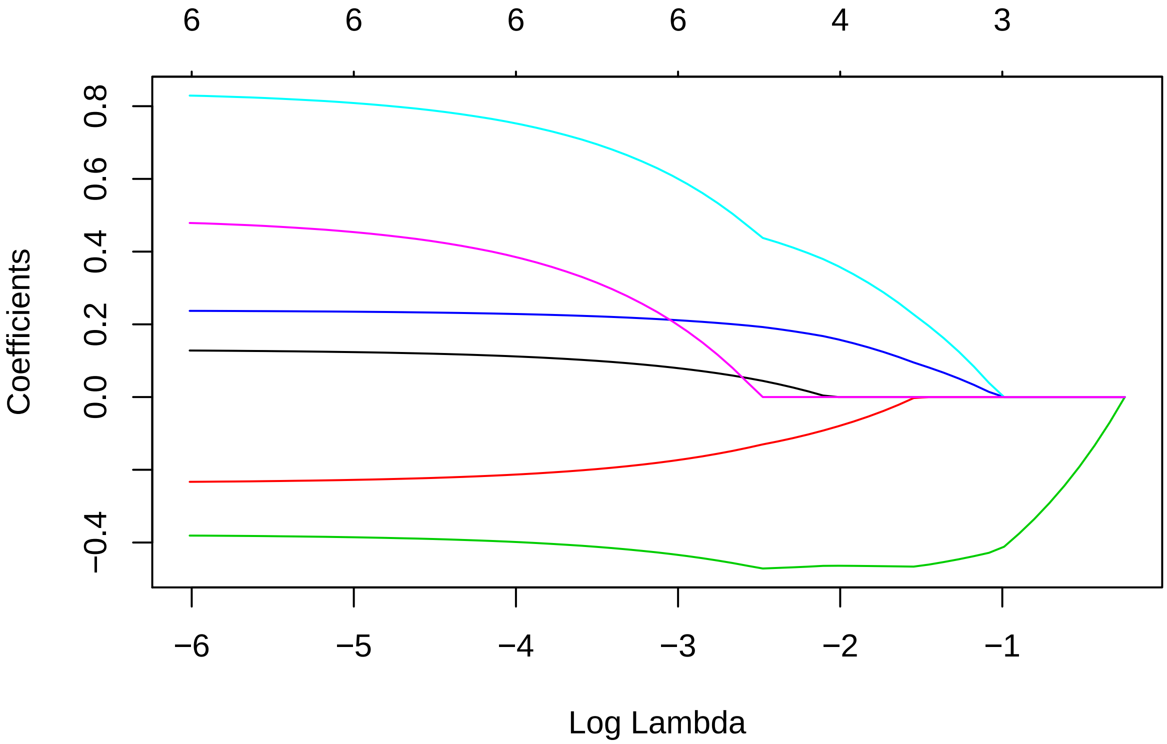
```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  0.006203863
## trans_date   .
## house_age    .
## dist_mrt    -0.488106125
## num_conven   0.069065533
## cluster2     .
## cluster3     .
## cluster4     .
## cluster5     .
## cluster6     .
## cluster7    -0.070999764
## cluster8     .
```

We scale the variables as in the previous lasso analysis. The lasso analysis in this subset of variables is different from the regsubsets and resampling validation methods. The lasso model selects all the variables for obtaining the lowest cross-validation error, unlike the previous methods which selected 9 variables. We see that here too trans_date is not among the first 4 variables to be selected.

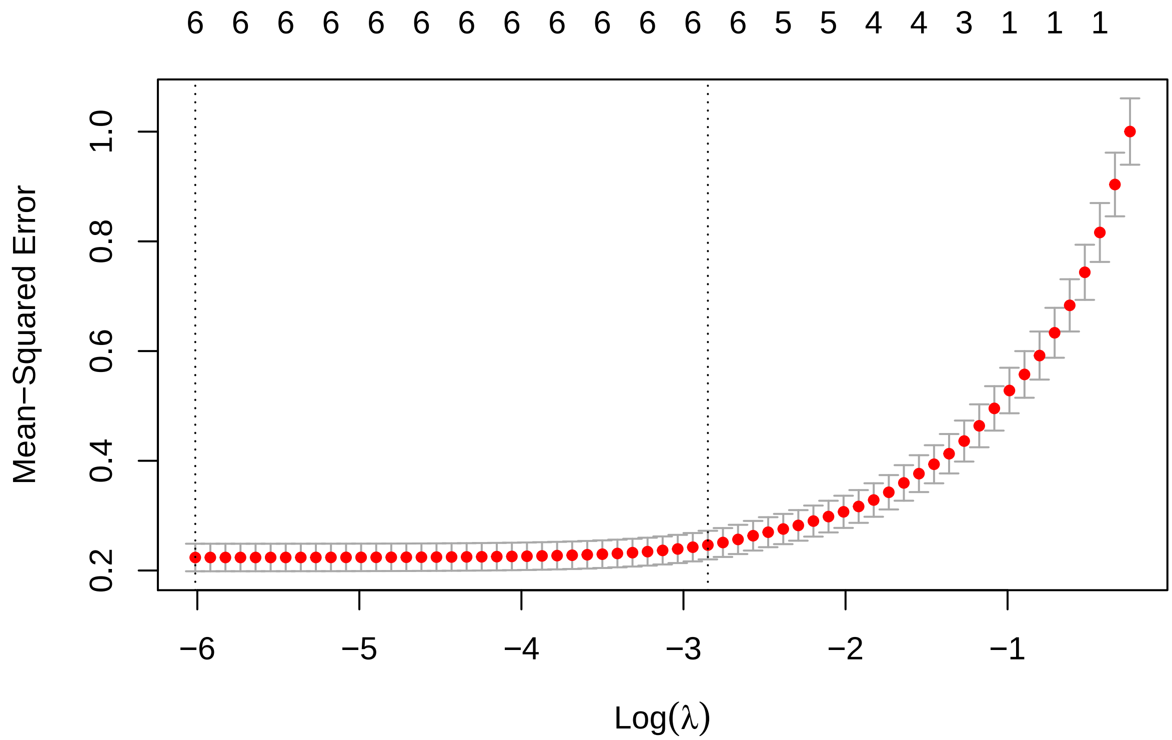
Model 3

```
target_data = real_estate %>%
  select(
    price_per_area,
    trans_date,
    house_age,
    dist_mrt,
    num_conven
  ) %>%
  mutate_all(scale) %>%
  mutate(urban = real_estate$urban)

x <- model.matrix(price_per_area~.,target_data)[,-1]
```

```r
y = target_data$price_per_area

lassoRes <- glmnet(x,y,alpha=1)
plot(lassoRes, xvar = "lambda")
```




```r
cvLassoRes <- cv.glmnet(x,y,alpha=1)
plot(cvLassoRes)
```




```r
predict(lassoRes,type="coefficients",s=cvLassoRes$lambda.min)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept) -0.4868673
## trans_date   0.1280173
## house_age   -0.2330881
## dist_mrt    -0.3809626
## num_conven   0.2370833
## urban1       0.8291076
## urban2       0.4788246
```

```r
predict(lassoRes,type="coefficients",s=cvLassoRes$lambda.1se)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) -0.29441653
## trans_date   0.07125995
## house_age   -0.16297522
## dist_mrt    -0.44318326
## num_conven   0.20690183
## urban1       0.56106048
## urban2       0.15004672
```

```r
predict(lassoRes,type="coefficients",s=0.2)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) -0.11720320
## trans_date   .
## house_age   -0.01540831
## dist_mrt    -0.46584678
```

```
## num_conven   0.10546473
## urban1       0.24890577
## urban2       .
```

```r
predict(lassoRes,type="coefficients",s=0.3)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) -0.04566308
## trans_date   .
## house_age    .
## dist_mrt    -0.44026805
## num_conven   0.03910842
## urban1       0.09697519
## urban2       .
```

We scale the variables for the lasso analysis as before. We see that the lasso cross validation selects all 6 variables as the other validation methods have done. We see again that date is not among the top 4 variables chosen by the lasso regularization. It seems that in all cases, the location of the house is more important than the age of the house as determined by the lasso regularization.

Discussion

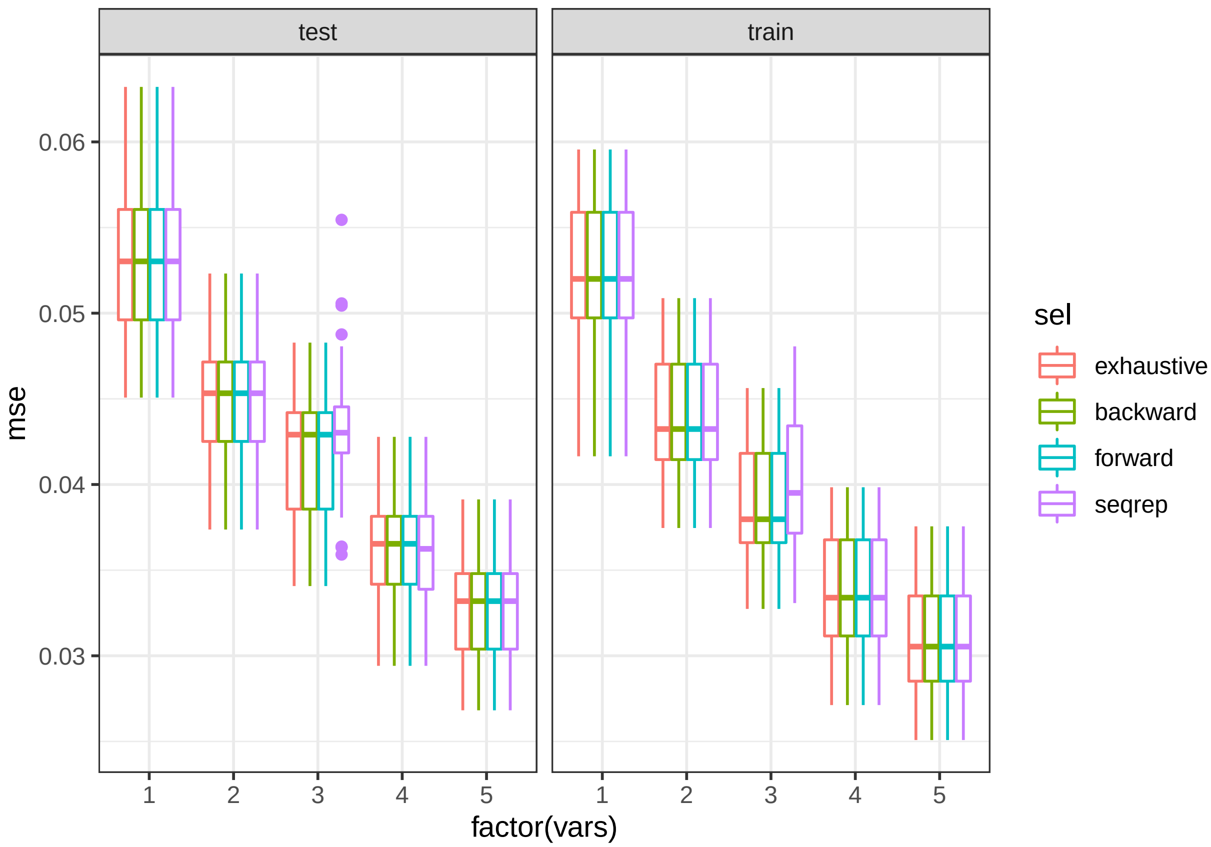
In the analysis of the various models we used in the prediction problem for determining the price per unit area of a house in several locations in Taiwan, we found that all 3 models did around the same. However, after visualizing the geographic data we saw that the way the data is spread, the latitude effectively measures the proximity of a house to the urban area pictured in the visualization. In a more general dataset where the distribution of housing is less easily separable by latitude, I believe the clustering method would be much more effective at predicting the price per unit area than the latitude and longitude approach.

One assumption we made during the exploration phase was that the date variable would not be meaningful in the regression but in our various cross-validation exploration we find that the various regression models do put weight on the date variable. To address this we look at the change in the error term when we omit the date variable on Model 3.

Discussion Analysis Extension

```r
target_data = real_estate %>%
  select(
    price_per_area,
    house_age,
    dist_mrt,
    num_conven,
    urban
  )

dfTmp <- NULL
whichSum <- array(0,dim=c(5,6,4),
  dimnames=list(NULL,colnames(model.matrix(price_per_area~.,target_data)),
      c("exhaustive", "backward", "forward", "seqrep")))
# Split data into training and test 30 times:
nTries <- 30
for ( iTry in 1:nTries ) {
  bTrain <- sample(rep(c(TRUE,FALSE),length.out=nrow(target_data)))
  # Try each method available in regsubsets
```

```
  # to select the best model of each size:
  for ( jSelect in c("exhaustive", "backward", "forward", "seqrep") ) {
    rsTrain <- regsubsets(price_per_area~.,target_data[bTrain,],nvmax=5,method=jSelect)
    # Add up variable selections:
    whichSum[,,jSelect] <- whichSum[,,jSelect] + summary(rsTrain)$which
    # Calculate test error for each set of variables
    # using predict.regsubsets implemented above:
    for ( kVarSet in 1:5 ) {
      # make predictions:
      testPred <- predict(rsTrain,target_data[!bTrain,],id=kVarSet)
      # calculate MSE:
      mseTest <- mean((testPred-target_data[!bTrain,"price_per_area"])^2 %$% price_per_area)
      # add to data.frame for future plotting:
      dfTmp <- rbind(dfTmp,data.frame(sim=iTry,sel=jSelect,vars=kVarSet,
      mse=c(mseTest,summary(rsTrain)$rss[kVarSet]/sum(bTrain)),trainTest=c("test","train")))
    }
  }
}
# plot MSEs by training/test, number of
# variables and selection method:
ggplot(dfTmp,aes(x=factor(vars),y=mse,colour=sel)) + geom_boxplot()+facet_wrap(~trainTest)+theme_bw()
```




```
reg_no_date = summary(lm(price_per_area ~ ., data = target_data))


target_data = real_estate %>%
  select(
```

```r
    price_per_area,
    trans_date,
    house_age,
    dist_mrt,
    num_conven,
    urban
  )

reg_date = summary(lm(price_per_area ~ ., data = target_data))

increase = round((reg_no_date$sigma - reg_date$sigma)/reg_date$sigma * 100, 2)

reg_date
```

```
##
## Call:
## lm(formula = price_per_area ~ ., data = target_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5851 -0.1084  0.0083  0.1077  0.7652
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -338.75080   61.32621  -5.524 5.95e-08 ***
## trans_date     0.17041    0.03047   5.593 4.12e-08 ***
## house_age     -0.09853    0.01004  -9.814  < 2e-16 ***
## dist_mrt      -0.12459    0.01244 -10.016  < 2e-16 ***
## num_conven     0.11460    0.01472   7.783 5.95e-14 ***
## urban1         0.31046    0.02488  12.478  < 2e-16 ***
## urban2         0.18225    0.02999   6.076 2.84e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1722 on 405 degrees of freedom
## Multiple R-squared:  0.7851, Adjusted R-squared:  0.7819
## F-statistic: 246.6 on 6 and 405 DF,  p-value: < 2.2e-16
```

```r
reg_no_date
```

```
##
## Call:
## lm(formula = price_per_area ~ ., data = target_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52342 -0.10932  0.00525  0.10607  0.82338
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.23367    0.10000  42.336  < 2e-16 ***
## house_age   -0.09723    0.01041  -9.345  < 2e-16 ***
## dist_mrt    -0.11514    0.01278  -9.013  < 2e-16 ***
## num_conven   0.12266    0.01519   8.075 7.76e-15 ***
## urban1       0.31874    0.02575  12.380  < 2e-16 ***
```

```
## urban2       0.18613    0.03108   5.988 4.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1785 on 406 degrees of freedom
## Multiple R-squared:  0.7685, Adjusted R-squared:  0.7657
## F-statistic: 269.6 on 5 and 406 DF,  p-value: < 2.2e-16
```

The two model summaries presented above are in order: regression model 3 with the transaction date and regression model 3 without the transaction date. We see that the increase in the residual standard error is only `3.66` percent. Given that we see very low correlation with the outcome variable both graphically and numerically I believe we should avoid including the varible in the final model until we can obtain more data over a longer period of time. There may be some cyclical effects based on the time of year, such as if there is a typical "moving season" where demand is higher than other points in the year, and in such a case then some type of date modeling would be prudent but without this additional information and given the low additional explanatory power of the date variable, shown here numerically and demonstrated as one of the last variables to be included by the lasso regularization, it does not seem wise to build our final model with this variable.