Feature_Engineering2 & Linear.Model

Yue Wen

This RMD file is about continue performing Feature engineering and build linear model to predict independent variable.

1. Missing data imputation

I will summarize what I did:

1) For tax/area related missing data, there is relationship between different tax featres, it works the same way for the area feature. Therefore, we used the related feature to impute the missing data by simple calculation.
2) For zip code/city, what we did here is to use longitude/latitude to find its nearest neighbor and use the same zip/city.
3) For the missing features that contains more then 80% NA, delete them.
4) For categorical features that contains a lot of NA, create new level missing.
5) For total area, using library(mice) to impute.

Because it takes long to run the imputation part, I save the result to the csv, and we don't evaluate the code below anymore. Instead, we directly import the CSV data generated.

```
# 1. read/prepare data
################################################################################
setwd("D:/data_camp/zillow_project")
train <- read.csv('train_property.csv', stringsAsFactors = F)
train$trans_year <- sapply(strsplit(train$transactiondate, '-'), '[[', 1)
train$trans_month <- sapply(strsplit(train$transactiondate, '-'), '[[', 2)
train$trans_day <- sapply(strsplit(train$transactiondate, '-'), '[[', 3)
train$trans_weekday <- weekdays(as.Date(train$transactiondate))
train$trans_DATE <- as.Date(train$transactiondate)
# train rename
train <- plyr::rename(train,
                      c("parcelid"="id_parcel",
                        "transactiondate" = "trans_date",
                        "yearbuilt" = "build_year",
                        "basementsqft"="area_base_living",
                        "yardbuildingsqft17"="area_patio",
                        "yardbuildingsqft26"="area_shed",
                        "poolsizesum"="area_pool",
                        "lotsizesquarefeet"="area_lot",
                        "garagetotalsqft"="area_garage",
                        "finishedfloor1squarefeet" = "area_firstfloor_finished",
                        "calculatedfinishedsquarefeet" = "area_total_calc",
                        "finishedsquarefeet6" = "area_base",
                        "finishedsquarefeet12" = "area_live_finished",
                        "finishedsquarefeet13" = "area_liveperi_finished",
                        "finishedsquarefeet15" = "area_total_finished",
                        "finishedsquarefeet50" = "area_unknown",
                        "unitcnt" = "num_unit",
```

```
                         "numberofstories" = "num_story",
                         "roomcnt" = "num_room",
                         "bathroomcnt" = "num_bathroom",
                         "bedroomcnt" = "num_bedroom",
                         "calculatedbathnbr" = "num_bathroom_calc",
                         "fullbathcnt" = "num_bath",
                         "threequarterbathnbr" = "num_75_bath",
                         "fireplacecnt" = "num_fireplace",
                         "poolcnt" = "num_pool",
                         "garagecarcnt" = "num_garage",
                         "regionidcounty" = "region_county",
                         "regionidcity" = "region_city",
                         "regionidzip" = "region_zip",
                         "regionidneighborhood" = "region_neighbor",
                         "taxvaluedollarcnt" = "tax_total",
                         "structuretaxvaluedollarcnt" = "tax_building",
                         "landtaxvaluedollarcnt" = "tax_land",
                         "taxamount" = "tax_property",
                         "assessmentyear" = "tax_year",
                         "taxdelinquencyflag" = "tax_delinquency",
                         "taxdelinquencyyear" = "tax_delinquency_year",
                         "propertyzoningdesc" = "zoning_property",
                         "propertylandusetypeid" = "zoning_landuse",
                         "propertycountylandusecode" = "zoning_landuse_county",
                         "fireplaceflag" = "flag_fireplace",
                         "hashottuborspa" = "flag_tub",
                         "buildingqualitytypeid" = "quality",
                         "buildingclasstypeid" = "framing",
                         "typeconstructiontypeid" = "material",
                         "decktypeid" = "deck",
                         "storytypeid" = "story",
                         "heatingorsystemtypeid" = "heating",
                         "airconditioningtypeid" = "aircon",
                         "architecturalstyletypeid" = "architectural_style",
                         "pooltypeid10" = "flag_spa",
                         "pooltypeid2" = "flag_pool_spa",
                         "pooltypeid7" = "flag_pool_tub",
                         "fips"="county"))
# numerical variable
variable_numeric = c("area_firstfloor_finished",
                     "area_base", "area_base_living",
                     "area_garage",
                     "area_live_finished",
                     "area_liveperi_finished",
                     "area_lot",
                     "area_patio",
                     "area_pool",
                     "area_shed",
                     "area_total_calc",
                     "area_total_finished",
                     "area_unknown",
                     "tax_building",
                     "tax_land",
                     "tax_property",
```

```
                     "tax_total",
                     "latitude",
                     "longitude")
# discrete
variable_discrete = c("num_75_bath",
                      "num_bath",
                      "num_bathroom",
                      "num_bathroom_calc",
                      "num_bedroom",
                      "num_fireplace",
                      "num_garage",
                      "num_pool",
                      "num_room",
                      "num_story",
                      "num_unit")
variable_binary = c("flag_fireplace",
                    "flag_tub",
                    "flag_spa",
                    "flag_pool_spa",
                    "flag_pool_tub",
                    "tax_delinquency")

# categorical variable
variable_nominal = c("aircon",
                     "architectural_style",
                     "county",
                     "deck",
                     "framing",
                     "heating",
                     "id_parcel",
                     "material",
                     "region_city",
                     "region_county",
                     "region_neighbor",
                     "region_zip",
                     "story",
                     "zoning_landuse",
                     "zoning_landuse_county")
variable_ordinal = c("quality")

# date
variable_date = c("tax_year",
                  "build_year",
                  "tax_delinquency_year",
                  "trans_year",
                  "trans_month",
                  "trans_day",
                  "trans_date",
                  "trans_weekday")

# others
variable_unstruct = c("zoning_property")

# don't understand
```

```
variable_unknown = c('censustractandblock',
                     'rawcensustractandblock')

# Conversion
# - convert some binary to 0, 1
# - convert to date to int
# - convert to numeric to double
# - convert to discrete to int
# - convert to categorical to character


train[train$flag_fireplace == "", "flag_fireplace"] = 0
train[train$flag_fireplace == "true", "flag_fireplace"] = 1
train[train$flag_tub == "", "flag_tub"] = 0
train[train$flag_tub == "true", "flag_tub"] = 1
train[train$tax_delinquency == "", "tax_delinquency"] = 0
train[train$tax_delinquency == "Y", "tax_delinquency"] = 1
# convert to date to int
train[,variable_date] = sapply(train[,variable_date], as.character)
# convert to numeric to double
train[,variable_numeric] = sapply(train[,variable_numeric], as.numeric)
# convert to discrete to int
train[,c(variable_discrete, variable_binary)] = sapply(train[,c(variable_discrete, variable_binary)], as
# convert to categorical to character
train[,c(variable_nominal, variable_ordinal)] = sapply(train[,c(variable_nominal, variable_ordinal)], as

#drop  the redundant value
train <- train[, !(names(train) %in% c("num_bath","num_bathroom_calc","region_county","tax_year"))]

#deleting missing value
num.NA <- sort(colSums(sapply(train, is.na)))
remain.col <- names(num.NA)[which(num.NA <= 0.8 * dim(train)[1])] #
train <- train[,remain.col]
####################################################################################

#2. missing value imputation
check.na <- function(train){
mis.col <- colSums(is.na(train))
mis.col <- mis.col[mis.col>0]
return(mis.col)
}

##(1) handle those have relative small missing value -> tax/total, tax_land/ tax_property

train[which(is.na(train$tax_total)),"tax_total"] <- train[which(is.na(train$tax_total)),"tax_property"]/
        quantile(train$tax_property/train$tax_total, 0.5, na.rm = T)

train[which(is.na(train$tax_property)),"tax_property"] <- train[which(is.na(train$tax_property)),"tax_to
  quantile(train$tax_property/train$tax_total, 0.5, na.rm = T)

train[which(is.na(train$tax_land)),"tax_land"] <- train[which(is.na(train$tax_land)),"tax_total"]*
  quantile(train$tax_land/train$tax_total, 0.5, na.rm = T)

train[which(is.na(train$tax_building)),"tax_building"] <- train[which(is.na(train$tax_building)),"tax_to
```

```
  train[which(is.na(train$tax_building)),"tax_land"]
check.na(train)

#2. then impute  region_zip/region_neighbor/region_city based on latitude/longtitude

#actually we might just delete region neighbor, because  the missing value does  not contain
#any information, and we have  to bother to impute it

with(train,t.test(logerror~ is.na(region_neighbor)))

# p-value is 0.07, we  can accept the  null hypothesis, region_neighbor does not contain info
#for those who are interested, take a look at how to find the nearest datapoint
#https://stackoverflow.com/questions/21977720/r-finding-closest-neighboring-point-and-number-of-neighbo

geo.df <- train[,c("latitude","longitude","region_zip","region_city")]

library(sp)
library(rgeos)

Impute.Zip<- function(row,search.range=1000){
    geo.df.trunc <-subset(geo.df, abs(geo.df$longitude + geo.df$latitude -
                                                          unlist(row["longitude"])-unlist(row["latitude

    coordinates(geo.df.trunc) <- ~longitude+latitude
    d <- gDistance(geo.df.trunc, byid=T)
    k = 2
    min.d <- apply(d, 1, function(x) order(x, decreasing=F)[k])
    neighbor <- min.d[rownames(row)]
    while(is.na(geo.df.trunc[neighbor,"region_zip"]$region_zip)){
      k = k + 1
      min.d <- apply(d, 1, function(x) order(x, decreasing=F)[k])
      neighbor <- min.d[rownames(row)]

    }
    return(geo.df.trunc[neighbor,"region_zip"]$region_zip)
}

na.zip.rowname <- which(is.na(train$region_zip))

for (i in na.zip.rowname){
  row <-  geo.df[i,]
  geo.df[i,"region_zip"] = Impute.Zip(row)
}

#similarly, impute city

Impute.city<- function(row,search.range=1000){
  geo.df.trunc <-subset(geo.df, abs(geo.df$longitude + geo.df$latitude -
                                      unlist(row["longitude"])-unlist(row["latitude"]))<search.range)

  coordinates(geo.df.trunc) <- ~longitude+latitude
  d <- gDistance(geo.df.trunc, byid=T)
  k = 2
  min.d <- apply(d, 1, function(x) order(x, decreasing=F)[k])
```

```
  neighbor <- min.d[rownames(row)]
  while(is.na(geo.df.trunc[neighbor,"region_city"]$region_city)){
    k = k + 1
    min.d <- apply(d, 1, function(x) order(x, decreasing=F)[k])
    neighbor <- min.d[rownames(row)]

  }
  return(geo.df.trunc[neighbor,"region_city"]$region_city)
}

na.city.rowname <- which(is.na(train$region_city))

count = 0

for (i in na.city.rowname){
  count = count + 1
  print(count)
  print(i)
  row <-  geo.df[i,]
  geo.df[i,"region_city"] = Impute.city(row)
}


train[,c("region_zip","region_city")] = geo.df[,c("region_zip","region_city")]
train <- train[,!names(train) != "region_neighbor"]

#delete censustractandblock, region_neighbor -> do not have explain power

train <- train[,!names(train) %in%c("X.1","X","censustractandblock","region_neighbor")]
check.na(train)
names(train)
plot(train$area_lot,train$area_lot)

#reduandant variable, delete area_live_finished
train <- train[,!names(train) == "area_live_finished"]

#fisrt handle  some simple one, we create a new level to indicate missing value

train$aircon <- ifelse(is.na(train$aircon),"missing",train$aircon)
train$num_garage <- ifelse(is.na(train$num_garage),"missing",train$num_garage)
train$area_garage <- ifelse(is.na(train$area_garage),"missing",train$area_garage)
train$heating<- ifelse(is.na(train$heating),"missing",train$heating)
train$num_story<- ifelse(is.na(train$num_story),"missing",train$num_story)
train$quality<- ifelse(is.na(train$quality),"missing",train$quality)
train$build_year<- ifelse(is.na(train$build_year),"missing",train$build_year)
check.na(train)

# then let us impute the remaining missing value using library(mice)
# we have aleady got number infomation, for now, I did not see the explanation power in num_unit

train <- train[,!names(train) == "num_unit"]
library("mice")

#put all the columns related to area
```

```
area.col <- c("area_lot","area_total_calc","num_bedroom","num_bathroom","num_room","tax_total")
imputed_Data <- mice(train[,area.col], m=1, maxit = 10, method = 'pmm', seed = 500)
plot(train$area_lot,complete(imputed_Data,1)$area_lot)
train$area_lot = complete(imputed_Data,1)$area_lot
abline(0,1)

train$area_total_calc = complete(imputed_Data,1)$area_total_calc

#check missing value
check.na(train)

# here, numerical value can't contain NA
plot(table(train$num_garage))

#keep one, delete  garage_area
train <- train[,!names(train) == "area_garage"]

#delete num_story
with(train, t.test(logerror~ is.na(num_story)))
train <- train[,!names(train) == "num_story"]
with(train, t.test(logerror~ is.na(num_garage)))
train <- train[,!names(train) == "num_garage"]
```

2 Feature Engineering

We seperate all the features into several different categories, then perform the feature engineering within categories. Here is a summary of feature engineering.

(1) Date-realted features

(a) "weekend": whether the sold day is weekend

(b) "Q2": whether the sold day belongs to Q2

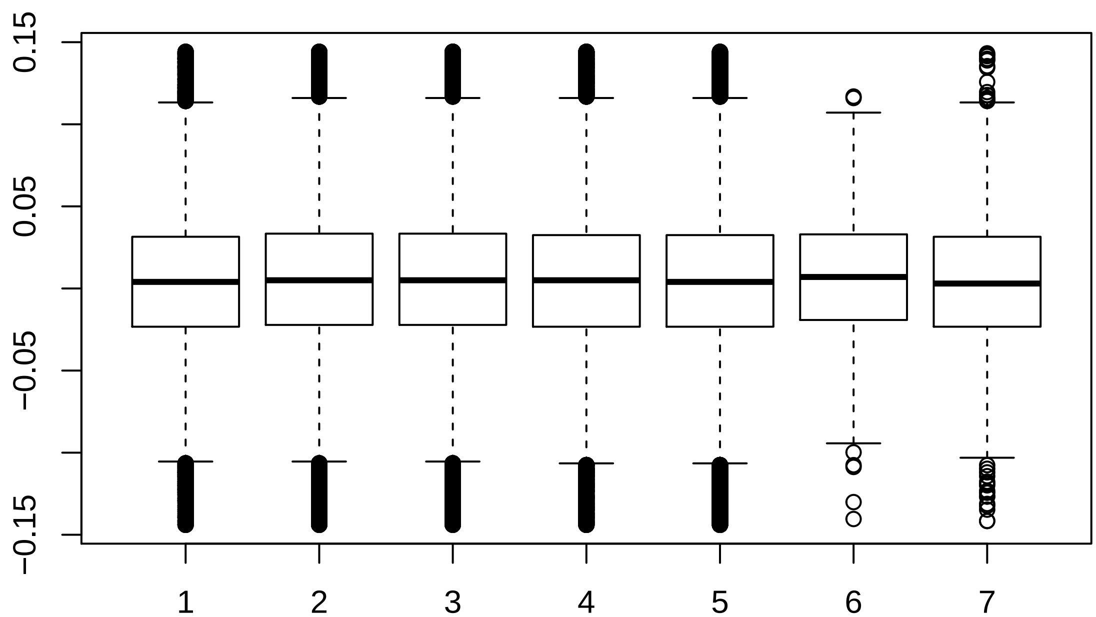
(c) "ancient level": categorize built year by cut (1940, 1995)

The above three features cover up day - week - season -year.

Note: the reasons I came up with the coming three features is due to visualization result by plotting mean(error)~ category.

```r
variable_date = c("build_year",
                  "trans_year",
                  "trans_month",
                  "trans_day",
                  "trans_date",
                  "trans_weekday")


# trans_weekday
par(mfrow = c(1,1))
boxplot(subset(train,trans_weekday == "Monday" &
          abs(logerror)<0.145)$logerror,
        subset(train,trans_weekday == "Tuesday" &
```

```r
              abs(logerror)<0.145)$logerror,
        subset(train,trans_weekday == "Wednesday" &
              abs(logerror)<0.145)$logerror,
        subset(train,trans_weekday == "Thursday" &
              abs(logerror)<0.145)$logerror,
        subset(train,trans_weekday == "Friday" &
              abs(logerror)<0.145)$logerror,
        subset(train,trans_weekday == "Saturday" &
              abs(logerror)<0.145)$logerror,
        subset(train,trans_weekday == "Sunday" &
              abs(logerror)<0.145)$logerror
        )
```



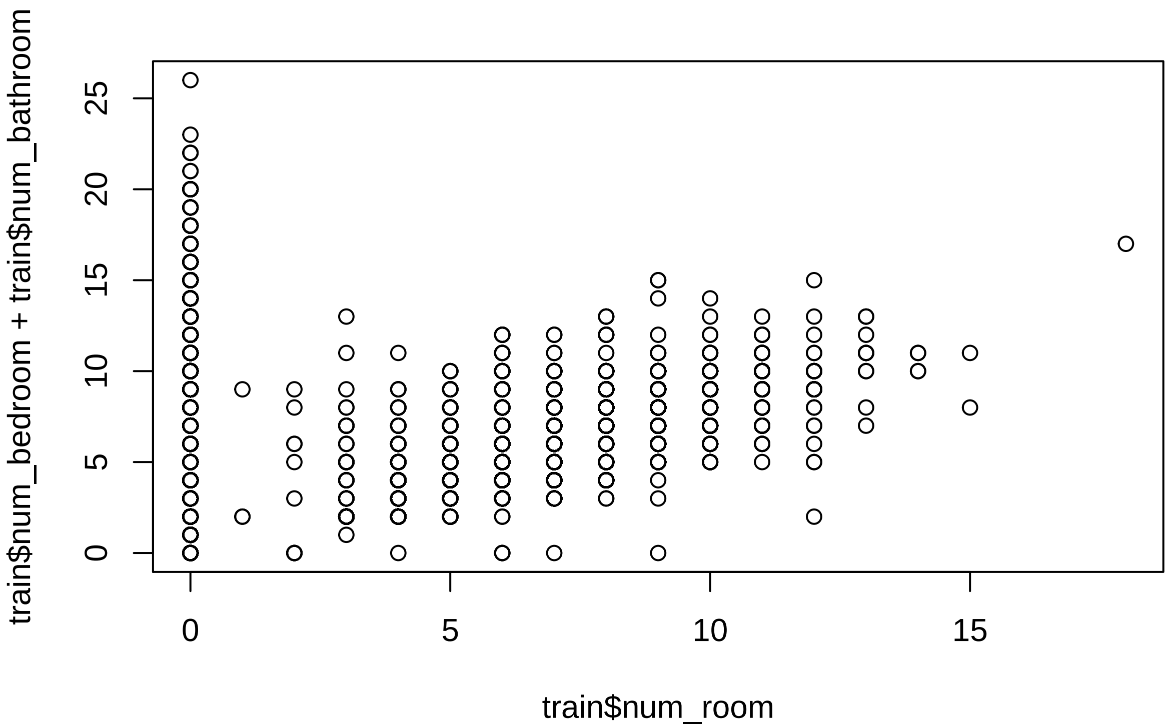

```r
with(train,t.test(logerror~ trans_weekday == "Saturday"|trans_weekday == "Sunday"))
```

```
##
##  Welch Two Sample t-test
##
## data:  logerror by trans_weekday == "Saturday" | trans_weekday == "Sunday"
## t = 3.941, df = 1195.1, p-value = 8.586e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.005499136 0.016402573
## sample estimates:
## mean in group FALSE  mean in group TRUE
##        0.0115915048        0.0006406504
```

```r
#new feature 1.
train$weekend =ifelse(train$trans_weekday== "Saturday"|train$trans_weekday == "Sunday",
                      1,0)

train$ancient.level <- ifelse(train$build_year == "missing","missing",
                              ifelse(as.integer(train$build_year)>1995,"new",
                                ifelse(as.integer(train$build_year)>1940,"old","ancient")))

train$Q2 = with(train,ifelse(trans_month %in% c("4","5","6"),"1","0"))
date.feature <- c("weekend","Q2","ancient.level")
```

(2) room/area related features

```r
plot(train$num_room,  train$num_bedroom + train$num_bathroom)
```




```r
with(train,t.test(abs(logerror)~ num_room >= num_bedroom + num_bathroom))
```

```
##
##  Welch Two Sample t-test
##
## data:  abs(logerror) by num_room >= num_bedroom + num_bathroom
## t = 5.7024, df = 34396, p-value = 1.191e-08
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.004327230 0.008859982
## sample estimates:
```

```
## mean in group FALSE  mean in group TRUE
##          0.06997593          0.06338233
```

```r
#whether the number of room is larger than the sum of the number of bedroom/bathroom

train$right_room <- with(train,ifelse(num_room >= num_bedroom + num_bathroom,1,0))

# bedroom/ bathroom ratio
train$bed_to_bath <- with(train,ifelse(num_bathroom == 0, 0,num_bedroom/num_bathroom))

#examine the correlation to see whether the  the features are informative features
cor(train$bed_to_bath,train$logerror)
```

```
## [1] -0.006295695
```

```r
cor(train$bed_to_bath,abs(train$logerror))
```

```
## [1] 0.004323492
```

```r
cor(train$num_bathroom,abs(train$logerror))
```

```
## [1] 0.001316901
```

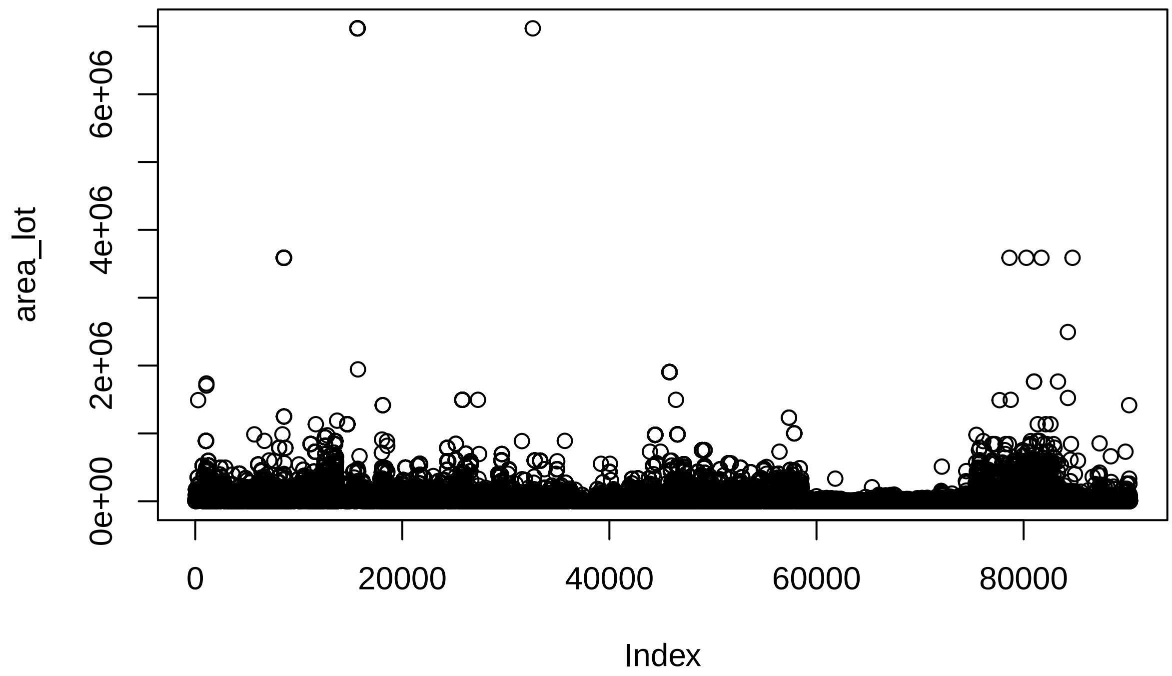
```r
#Generate  other features

train$are_per_room<- with(train,ifelse(num_room == 0, 0, area_total_calc/num_room))

#dummy variable indicating whether the count of room is zero
train$zero_room <- with(train, ifelse(num_room  ==0 ,1,0))

cor(train$are_per_room,abs(train$logerror))
```

```
## [1] -0.02167194
```

```r
cor(train$area_total_calc,abs(train$logerror))
```

```
## [1] 0.03953064
```

```r
par(mfrow = c(1,1))

with(train,plot(area_lot),plot(area_total_calc))
```




```r
room.area.feature <- c("num_room","num_bedroom","num_bathroom","right_room","bed_to_bath",
                       "are_per_room","zero_room","area_total_calc")
```

(3) facility realted features

```r
table(train$aircon)
```

```
##
##       1      11      13       3       5       9 missing
##   26668      63    1833       1     215       1   61494
```

```r
by(train,train$aircon,function(x){mean(x$logerror)})
```

```
## train$aircon: 1
## [1] 0.01285241
## --------------------------------------------------------
## train$aircon: 11
## [1] 0.02629365
## --------------------------------------------------------
## train$aircon: 13
## [1] 0.01546552
## --------------------------------------------------------
## train$aircon: 3
## [1] 0.0917
## --------------------------------------------------------
## train$aircon: 5
```

```
## [1] 0.01568233
## --------------------------------------------------------
## train$aircon: 9
## [1] 0.01
## --------------------------------------------------------
## train$aircon: missing
## [1] 0.01070144
```

```r
#collapse categorical features with too many levels
#rebuild level based on mean  log_error
train$new.aircon = with(train,ifelse(aircon=="missing",aircon,
                                ifelse(aircon=="5"|aircon=="13","5/13",
                                  ifelse(aircon=="11","11","1/3/9"))))

table(train$heating)
```

```
##
##       1      10      11      12      13      14      18       2      20
##      13       2       1       1      76       2      25   38303      97
##      24       6       7 missing
##    1071     970   15519   34195
```

```r
by(train,train$heating,function(x){mean(x$logerror)})
```

```
## train$heating: 1
## [1] 0.02457692
## --------------------------------------------------------
## train$heating: 10
## [1] -0.0293
## --------------------------------------------------------
## train$heating: 11
## [1] -0.0151
## --------------------------------------------------------
## train$heating: 12
## [1] -0.0131
## --------------------------------------------------------
## train$heating: 13
## [1] -1.184211e-05
## --------------------------------------------------------
## train$heating: 14
## [1] -0.0049
## --------------------------------------------------------
## train$heating: 18
## [1] 0.027436
## --------------------------------------------------------
## train$heating: 2
## [1] 0.01324539
## --------------------------------------------------------
## train$heating: 20
## [1] -0.0006814433
## --------------------------------------------------------
## train$heating: 24
## [1] -0.01097722
## --------------------------------------------------------
## train$heating: 6
```

```
## [1] 0.009798144
## --------------------------------------------------------
## train$heating: 7
## [1] 0.007491887
## --------------------------------------------------------
## train$heating: missing
## [1] 0.01205165
```

```
# capllase heating's level to new heating
train$new.heating = with(train,ifelse(heating == "missing",heating,
                                  ifelse(heating %in% c("10","11","12","13","14","20","24"),"neg",
                                         ifelse(heating %in% c("6","7"),"6/7","2/1/18"))))

#Create new level to indicate whether the categorical variavle is missing
train$missing.heating <- with(train,ifelse(heating == "missing",1,0))
train$missing.aircon <- with(train,ifelse(aircon == "missing",1,0))

facility.feature <- c("flag_tub","flag_fireplace","missing.aircon","new.heating")
```

(4) tax realted features

```
with(train,cor(tax_delinquency,logerror))
```

```
## [1] 0.01893559
```

```
# Create tax percentage indicating the percentage of tax paid
train$tax.perc <- train$tax_property/ train$tax_total
with(train,cor(tax.perc,logerror))
```

```
## [1] -0.003674848
```

```
train$right.tax <-ifelse(train$tax.perc < 0.1,1,0)
train$build.to.total <- train$tax_building/train$tax_total
with(train,cor(build.to.total,logerror))
```

```
## [1] 0.01948048
```

```
train$tax.per.area <-  train$tax_property/train$area_total_calc
tax.feature <- c("tax_delinquency","tax_total","tax_property","tax_land","right.tax","tax.per.area","bu
```

(5)Overall evaluation feature(Collapse quality)

```
quality.error <- by(train,train$quality, function(x){mean(x$logerror)})
table(train$quality)
```

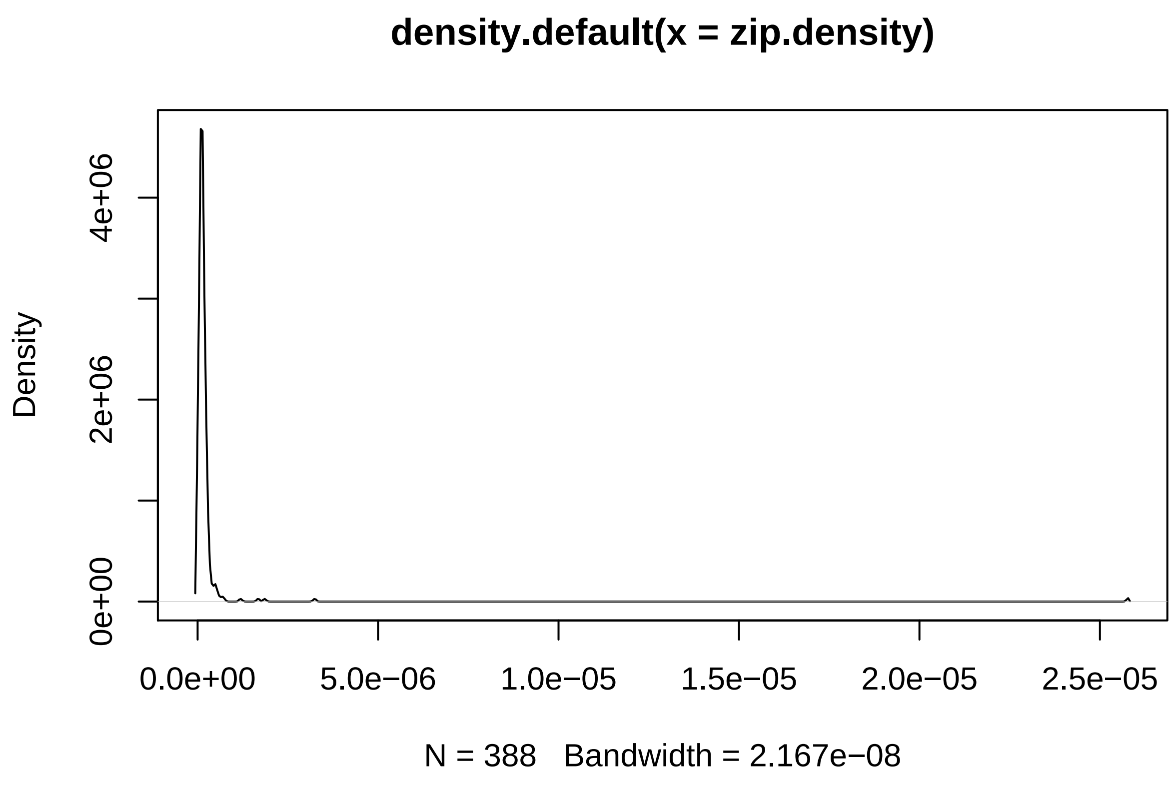
```
##
##       1      10      11      12       4       6       7       8 missing
##    2627    1461       1     119   23839       2   29310       5   32911
```

```
#collapse quality features based on their  influence  on response
train$new.quality <- with(train,ifelse(quality=="missing",quality,
                                ifelse(quality %in% c("12","7","11"),"12-7-11",
                                ifelse(quality %in% c("4","1"), "4-1","6-8-10")
                                       )
```

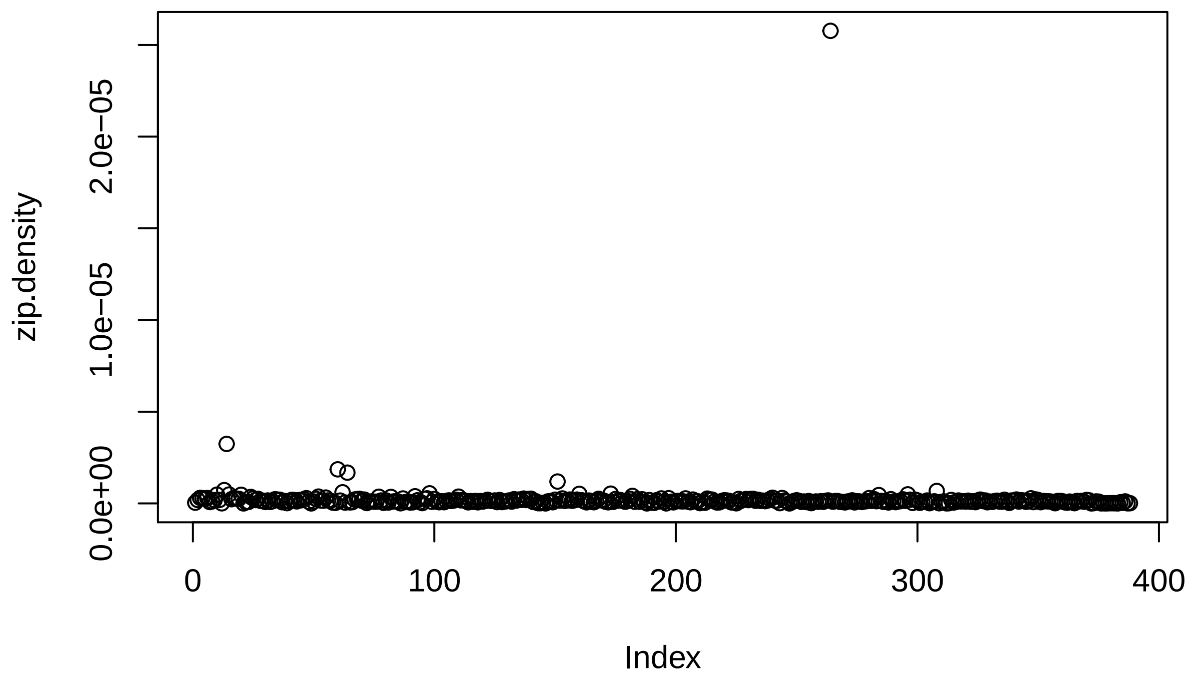
```
))

quality.feature <- c("new.quality")
```

(6) geometry related feature

```
zip.num <- by(train, train$region_zip, function(x) nrow(x))
train$zip.num <- zip.num[train$region_zip]

zip.area  <- by(train, train$region_zip, function(x)
{abs((max(x$longitude) - min(x$longitude))*(max(x$latitude) - min(x$latitude)))})
train$zip.area <- zip.area[train$region_zip]

zip.density <- ifelse(zip.area  == 0 , 0 ,zip.num/zip.area)
zip.density.accuracy <- ifelse(zip.num>50,1,0)
zip.density.revised  <- zip.density*zip.density.accuracy
train$zip.density.revised <- zip.density.revised[train$region_zip]

plot(density(zip.density))
```



```r
plot(zip.density)
```




```r
train$zip.density <- zip.density[train$region_zip]
with(train,cor(zip.density,logerror))
```

```
## [1] -0.01249644
```

```r
with(train,cor(zip.density,abs(logerror)))
```

```
## [1] 0.0162063
```

```r
train$zip.density <- zip.density[train$region_zip]

##
city.zip <- by(train,train$region_city, function(x){length(unique(x$region_zip))})
train$zip.per.city <- city.zip[train$region_city]
with(train,cor(zip.per.city ,logerror))
```

```
## [1] -0.006879891
```

```r
city.area <- by(train, train$region_city, function(x){abs((max(x$longitude) - min(x$longitude))*(max(x$l

city.num  <- by(train, train$region_city, function(x) nrow(x))
city.density <- city.area/city.num
train$city.density <- city.density[train$region_city]
with(train,cor(city.density ,abs(logerror)))
```

```
## [1] 0.01449351
```

```r
train$county6111 <- with(train, ifelse(county == "6111",1,0))
####SELECT features
geo.feature<- c("county6111","zip.density.revised")
```

build linear model

Here, we do not list every linear model built. Instead, we just give an example to show the basic idea of how to transform features using residual diagnosis.

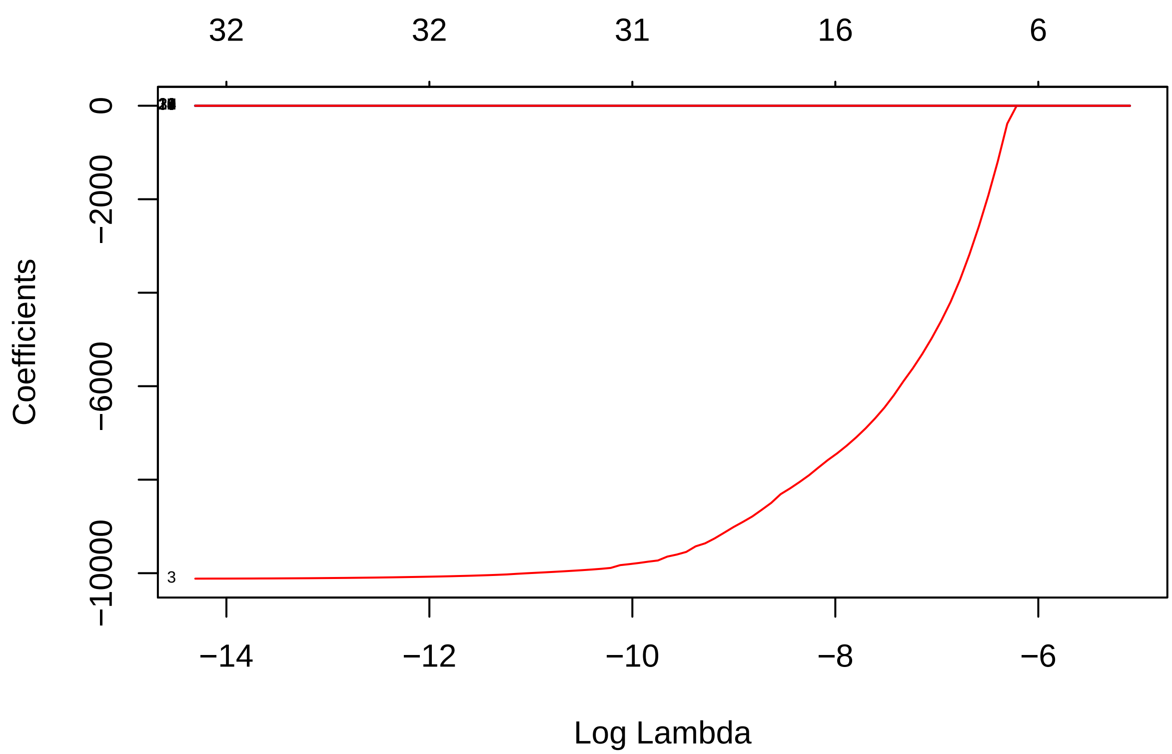
```r
selected.feature <- c("logerror",geo.feature,quality.feature,tax.feature,facility.feature,room.area.fea
reg.df <- train[,selected.feature]

full.model <- lm(logerror~.,data = reg.df)
summary(full.model)
```

```
##
## Call:
## lm(formula = logerror ~ ., data = reg.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6260 -0.0378 -0.0059  0.0275  4.7276
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -5.332e-04  1.216e-02  -0.044  0.96503
## county6111            1.418e-03  2.495e-03   0.568  0.56995
## zip.density.revised  -1.010e+04  4.008e+03  -2.518  0.01179 *
## new.quality4-1       -9.122e-04  1.780e-03  -0.512  0.60839
## new.quality6-8-10    -2.195e-03  4.834e-03  -0.454  0.64985
## new.qualitymissing    1.201e-03  2.944e-03   0.408  0.68324
## tax_delinquency       2.450e-02  3.862e-03   6.344 2.25e-10 ***
## tax_total             3.995e-08  5.688e-09   7.024 2.18e-12 ***
## tax_property         -2.850e-06  2.738e-07 -10.411  < 2e-16 ***
## tax_land             -1.866e-08  6.175e-09  -3.021  0.00252 **
## right.tax             4.260e-02  6.262e-03   6.803 1.03e-11 ***
## tax.per.area         -5.028e-05  7.097e-05  -0.708  0.47869
## build.to.total        2.306e-04  3.680e-03   0.063  0.95003
## tax.perc             -1.795e-04  1.232e-03  -0.146  0.88420
## flag_tub             -1.518e-02  3.493e-03  -4.344 1.40e-05 ***
## flag_fireplace       -1.030e-03  1.118e-02  -0.092  0.92662
## missing.aircon        2.773e-03  1.597e-03   1.736  0.08253 .
## new.heating6/7       -7.506e-04  2.098e-03  -0.358  0.72051
## new.heatingmissing   -6.748e-03  2.801e-03  -2.409  0.01600 *
## new.heatingneg       -2.652e-02  5.404e-03  -4.908 9.24e-07 ***
## num_room             -1.133e-03  8.110e-04  -1.398  0.16225
## num_bedroom           4.710e-04  1.184e-03   0.398  0.69080
## num_bathroom         -2.737e-03  1.595e-03  -1.716  0.08615 .
## right_room           -1.581e-02  5.451e-03  -2.900  0.00373 **
## bed_to_bath          -2.943e-03  1.967e-03  -1.496  0.13457
## are_per_room         -4.017e-05  1.372e-05  -2.929  0.00340 **
## zero_room            -3.568e-02  8.755e-03  -4.076 4.59e-05 ***
## area_total_calc       1.107e-05  1.245e-06   8.895  < 2e-16 ***
```

```
## weekend               -1.229e-02  4.866e-03  -2.526  0.01153 *
## Q21                   -6.899e-03  1.135e-03  -6.077 1.23e-09 ***
## ancient.levelmissing  1.148e-02  7.586e-03   1.514  0.13009
## ancient.levelnew      4.555e-03  2.508e-03   1.816  0.06935 .
## ancient.levelold      3.468e-03  1.891e-03   1.834  0.06661 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1606 on 90242 degrees of freedom
## Multiple R-squared:  0.006293,   Adjusted R-squared:  0.005941
## F-statistic: 17.86 on 32 and 90242 DF,  p-value: < 2.2e-16
```

Then let us start the model selection using lasso regression.

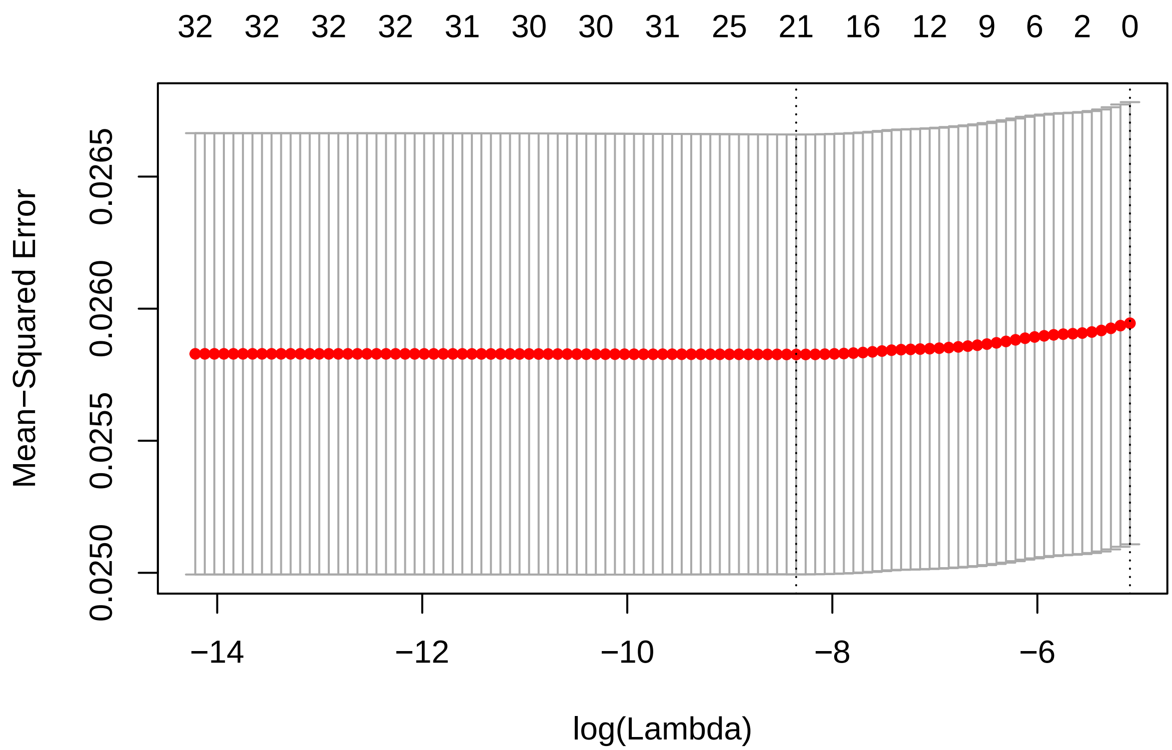
```r
library(glmnet)
ind = model.matrix(~.,reg.df[,-1])
dep = reg.df$logerror
fit <- glmnet(x=ind, y=dep)
plot(fit, xvar = "lambda", label = T)
```




```r
cvfit <- cv.glmnet(ind, dep)
cvfit$lambda.min
```

```
## [1] 0.0002357268
```

```r
plot(cvfit)
```




```r
x = coef(cvfit, s = "lambda.min")
x
```

```
## 34 x 1 sparse Matrix of class "dgCMatrix"
##                                   1
## (Intercept)          -4.669815e-02
## (Intercept)            .
## county6111             1.990462e-03
## zip.density.revised  -8.050513e+03
## new.quality4-1         .
## new.quality6-8-10      .
## new.qualitymissing     1.845913e-04
## tax_delinquency        2.214832e-02
## tax_total              1.574528e-08
## tax_property         -1.960022e-06
## tax_land               .
## right.tax              4.792814e-02
## tax.per.area         -4.684791e-05
## build.to.total         3.312448e-03
## tax.perc               .
## flag_tub             -1.166651e-02
## flag_fireplace         .
## missing.aircon         5.931688e-04
## new.heating6/7         .
```

```
## new.heatingmissing   -2.173368e-03
## new.heatingneg       -1.767372e-02
## num_room              .
## num_bedroom          -1.579857e-04
## num_bathroom          .
## right_room            .
## bed_to_bath          -4.846860e-05
## are_per_room          .
## zero_room            -2.082931e-04
## area_total_calc       9.285067e-06
## weekend              -1.007241e-02
## Q21                  -6.325344e-03
## ancient.levelmissing  .
## ancient.levelnew      2.723897e-03
## ancient.levelold      1.901514e-03
```

We can see lasso regression did not really help us choose some features.

However, we want to use residual plot to seek for new opportunies to improve the mdoel. I will just give an example for the feature bedroom to bathroom rate.

After performing the square root transfromation, the residual plot will look better. And we trained the model agian, the adjusted R-square does improve as well.

Then, for each feature, we tried to perform the some transformation, also include some intersection terms.

After trying different transformations and intersactions, I showed the best result bewlow.

```r
mod <- lm(logerror~.-zip.density.revised +I((train$zip.density.revised)^(1/18))
           -are_per_room + I(are_per_room^(1/15))
           -county6111-new.quality
           #-tax_land +I(tax_land^(1/2))
           -tax_property + I(tax_property^(1/60))
           -num_room
           -bed_to_bath + I((bed_to_bath)^(1/2))
           -tax.per.area + I(tax.per.area^(1/50))
           -flag_fireplace
           -missing.aircon
           -tax.perc + I(tax.perc^(1/12))
           -right.tax
           + right.tax:I(tax_land^(1/3))
           +I(build.to.total^(1/60))
          + flag_tub : new.heating
          +right_room : zero_room
          -flag_tub
          +Q2:weekend

           ,data = reg.df)
summary(mod)
```

```
##
## Call:
## lm(formula = logerror ~ . - zip.density.revised + I((train$zip.density.revised)^(1/18)) -
##     are_per_room + I(are_per_room^(1/15)) - county6111 - new.quality -
##     tax_property + I(tax_property^(1/60)) - num_room - bed_to_bath +
##     I((bed_to_bath)^(1/2)) - tax.per.area + I(tax.per.area^(1/50)) -
##     flag_fireplace - missing.aircon - tax.perc + I(tax.perc^(1/12)) -
```

```
##     right.tax + right.tax:I(tax_land^(1/3)) + I(build.to.total^(1/60)) +
##     flag_tub:new.heating + right_room:zero_room - flag_tub +
##     Q2:weekend, data = reg.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6221 -0.0381 -0.0058  0.0284  4.7295
##
## Coefficients:
##                                         Estimate Std. Error t value
## (Intercept)                            1.847e+00  1.420e-01  13.009
## tax_delinquency                        2.456e-02  3.856e-03   6.371
## tax_total                              2.173e-08  5.454e-09   3.984
## tax_land                              -3.180e-08  6.898e-09  -4.610
## build.to.total                        -2.168e-02  7.113e-03  -3.049
## new.heating6/7                        -1.276e-03  1.814e-03  -0.703
## new.heatingmissing                    -5.660e-03  1.620e-03  -3.494
## new.heatingneg                        -2.974e-02  4.939e-03  -6.022
## num_bedroom                            4.983e-03  1.267e-03   3.933
## num_bathroom                          -6.229e-03  1.686e-03  -3.695
## right_room                            -1.496e-02  6.528e-03  -2.291
## zero_room                             -1.878e-01  8.083e-02  -2.324
## area_total_calc                        1.732e-05  2.058e-06   8.416
## weekend                               -1.243e-02  5.171e-03  -2.404
## Q21                                   -6.639e-03  1.136e-03  -5.842
## ancient.levelmissing                   3.348e-02  9.264e-03   3.614
## ancient.levelnew                       1.269e-02  2.381e-03   5.329
## ancient.levelold                       5.460e-03  1.813e-03   3.012
## I((train$zip.density.revised)^(1/18)) -4.459e-02  1.171e-02  -3.808
## I(are_per_room^(1/15))                -1.185e-01  5.378e-02  -2.203
## I(tax_property^(1/60))                -1.798e+00  2.325e-01  -7.734
## I((bed_to_bath)^(1/2))                -2.377e-02  5.328e-03  -4.462
## I(tax.per.area^(1/50))                 4.524e-01  1.940e-01   2.333
## I(tax.perc^(1/12))                    -1.625e-01  5.241e-02  -3.100
## I(build.to.total^(1/60))               7.430e-02  1.073e-02   6.923
## right.tax:I(tax_land^(1/3))            3.703e-04  1.587e-04   2.333
## flag_tub:new.heating2/1/18             1.678e-03  6.872e-03   0.244
## flag_tub:new.heating6/7               -8.010e-03  1.422e-02  -0.563
## flag_tub:new.heatingmissing           -1.952e-02  4.145e-03  -4.708
## flag_tub:new.heatingneg                4.665e-02  1.135e-01   0.411
## right_room:zero_room                  -1.593e-02  1.151e-02  -1.384
## weekend:Q21                            5.243e-03  1.506e-02   0.348
##                                       Pr(>|t|)
## (Intercept)                            < 2e-16 ***
## tax_delinquency                       1.89e-10 ***
## tax_total                             6.78e-05 ***
## tax_land                              4.04e-06 ***
## build.to.total                        0.002299 **
## new.heating6/7                        0.481892
## new.heatingmissing                    0.000475 ***
## new.heatingneg                        1.73e-09 ***
## num_bedroom                           8.39e-05 ***
## num_bathroom                          0.000220 ***
## right_room                            0.021942 *
```

```
## zero_room                              0.020149 *
## area_total_calc                         < 2e-16 ***
## weekend                                0.016205 *
## Q21                                    5.16e-09 ***
## ancient.levelmissing                   0.000301 ***
## ancient.levelnew                       9.89e-08 ***
## ancient.levelold                       0.002594 **
## I((train$zip.density.revised)^(1/18)) 0.000140 ***
## I(are_per_room^(1/15))                 0.027566 *
## I(tax_property^(1/60))                 1.05e-14 ***
## I((bed_to_bath)^(1/2))                 8.15e-06 ***
## I(tax.per.area^(1/50))                 0.019670 *
## I(tax.perc^(1/12))                     0.001937 **
## I(build.to.total^(1/60))               4.46e-12 ***
## right.tax:I(tax_land^(1/3))            0.019644 *
## flag_tub:new.heating2/1/18             0.807115
## flag_tub:new.heating6/7                0.573320
## flag_tub:new.heatingmissing            2.50e-06 ***
## flag_tub:new.heatingneg                0.680953
## right_room:zero_room                   0.166344
## weekend:Q21                            0.727687
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1603 on 90243 degrees of freedom
## Multiple R-squared:  0.01029,    Adjusted R-squared:  0.009954
## F-statistic: 30.28 on 31 and 90243 DF,  p-value: < 2.2e-16
```