Final_Project

Libaries

```
pacman::p_load(data.table,tidyverse,magrittr,YARF,skimr,plyr,tidyr,YARF,mltools,caret)
```

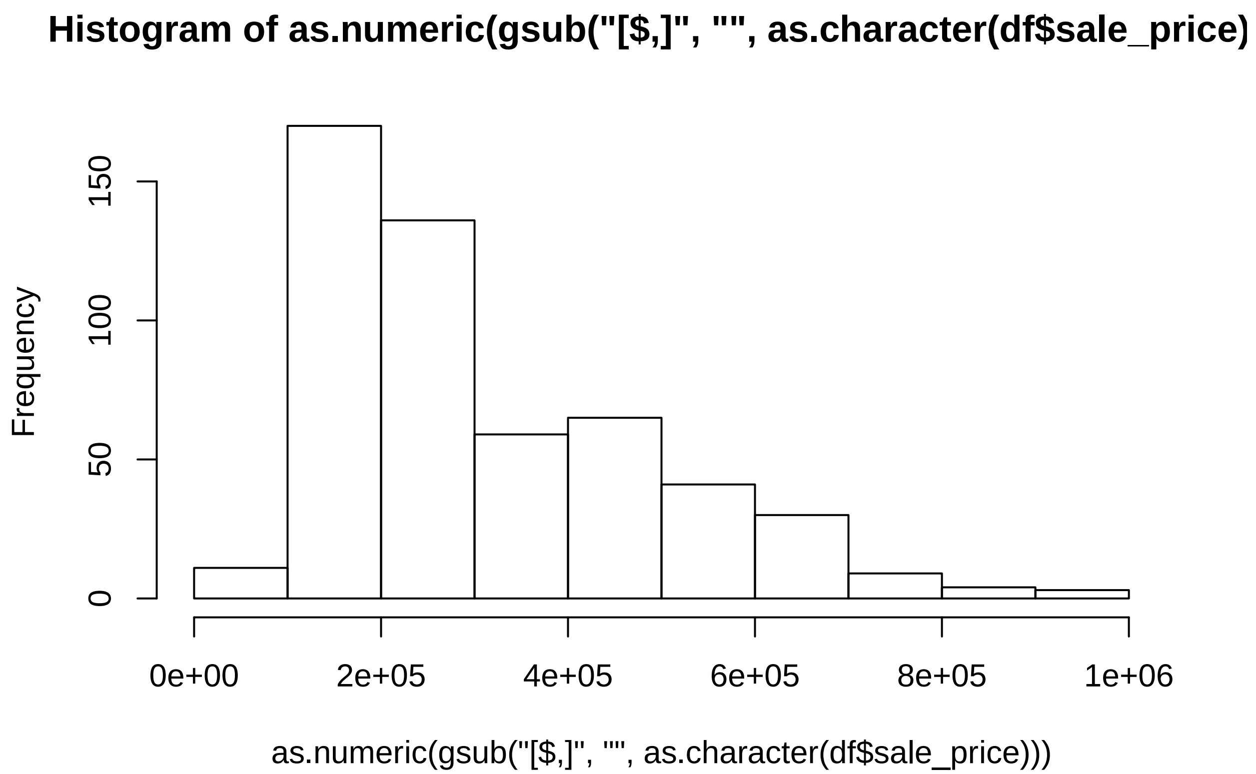
Loading Data

```
df= read.csv("housing_data_2016_2017.csv")
head(df,1)
```

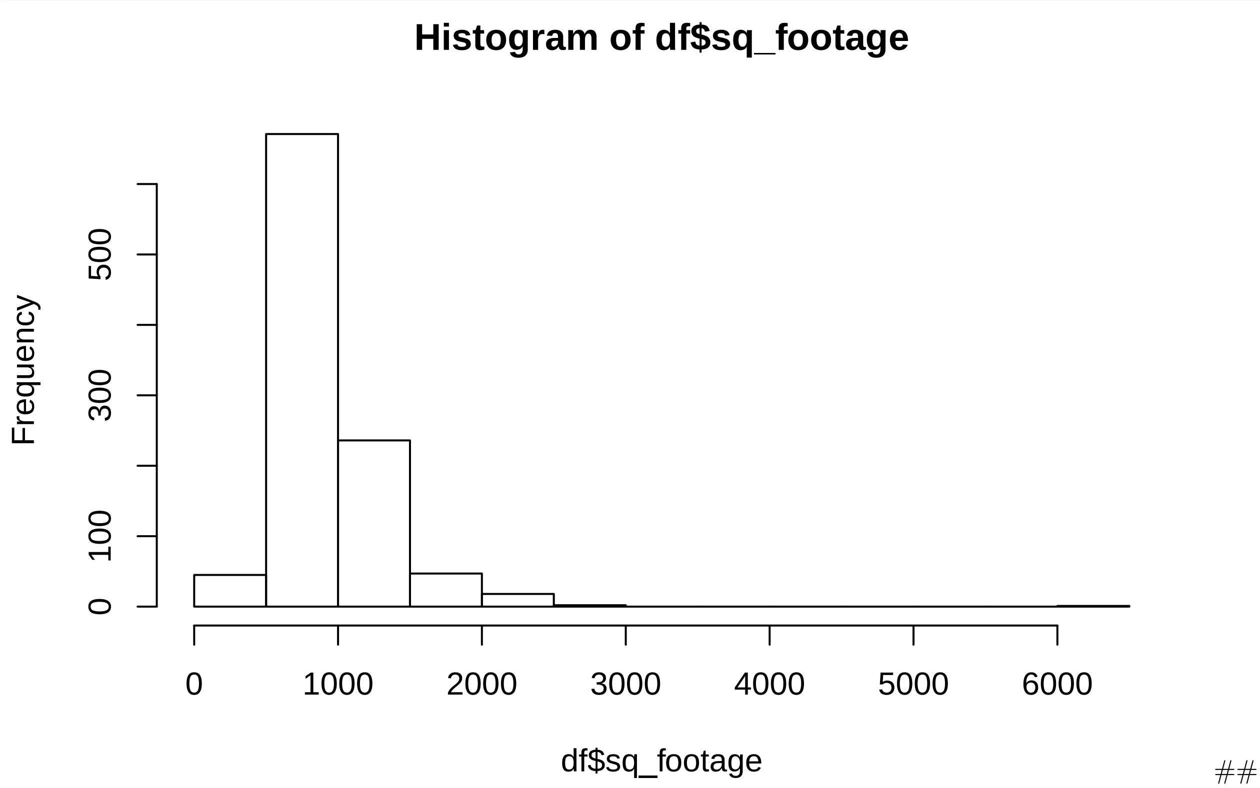
```
##                             HITId                      HITTypeId
## 1 3OID399FXG7F26JWONXF0Y86J90FD4 36BILMLQB75QQNBTYKGYCZWDN8TVAU
##                                                                    Title
## 1 Find Information about Housing To Help a Student Project -- Very easy
##                                           Description Keywords Reward
## 1 Go to a link and copy information into the HIT       NA $0.05
##                 CreationTime MaxAssignments
## 1 Wed Feb 15 22:13:37 PST 2017              1
##                                RequesterAnnotation AssignmentDurationInSeconds
## 1 BatchId:2689947;OriginalHitTemplateId:920937336;                         900
##   AutoApprovalDelayInSeconds                   Expiration NumberOfSimilarHITs
## 1                         60 Wed Feb 22 22:13:37 PST 2017                  NA
##   LifetimeInSeconds                   AssignmentId       WorkerId
## 1                NA 32KTQ2V7RDFCSAWQOW1SXC5AZIC9MB A231MNJJDDF3LS
##   AssignmentStatus                   AcceptTime                   SubmitTime
## 1         Approved Thu Feb 16 05:32:36 PST 2017 Thu Feb 16 05:35:37 PST 2017
##               AutoApprovalTime            ApprovalTime RejectionTime
## 1 Thu Feb 16 05:36:37 PST 2017 2017-02-16 13:37:11 UTC            NA
##   RequesterFeedback WorkTimeInSeconds LifetimeApprovalRate
## 1                NA               181       100% (187/187)
##   Last30DaysApprovalRate Last7DaysApprovalRate
## 1         100% (187/187)        100% (187/187)
##                                                                                                  URL
## 1 http://www.mlsli.com/homes-for-sale/address-not-available-from-broker-Flushing-NY-11355-149238320
##   approx_year_built cats_allowed common_charges community_district_num
## 1              1955           no           $767                     25
##   coop_condo date_of_sale dining_room_type dogs_allowed fuel_type
## 1      co-op    2/16/2016            combo           no       gas
##   full_address_or_zip_code garage_exists kitchen_type maintenance_cost
## 1       Flushing NY, 11355          <NA>       eat in             <NA>
##         model_type num_bedrooms num_floors_in_building num_full_bathrooms
## 1 Mitchell Garden 3            2                      6                  1
##   num_half_bathrooms num_total_rooms parking_charges pct_tax_deductibl
## 1                 NA               5            <NA>                NA
##   sale_price sq_footage total_taxes walk_score listing_price_to_nearest_1000
## 1  $228,000          NA        <NA>         82                          <NA>
##    url
```

```
## 1 <NA>
```

```r
hist(as.numeric(gsub('[$,]','',as.character(df$sale_price))))
```




```r
hist(df$sq_footage)
```



##

Useful Summary of Data This gives a broad overview on the data,

```r
skim(df)
```

Table 1: Data summary

Name	df
Number of rows	2230
Number of columns	55
Column type frequency:	
factor	36
logical	5
numeric	14
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
HITId	758	0.66	FALSE	1472	301: 1, 301: 1, 301: 1, 302: 1
HITTypeId	758	0.66	FALSE	2	310: 944, 36B: 528
Title	758	0.66	FALSE	1	Fin: 1472
Description	758	0.66	FALSE	2	Got: 944, Go : 528
Reward	758	0.66	FALSE	1	$0.: 1472
CreationTime	758	0.66	FALSE	62	Thu: 43, Thu: 40, Wed: 39, Thu: 37
RequesterAnnotation	758	0.66	FALSE	2	Bat: 944, Bat: 528
Expiration	758	0.66	FALSE	62	Thu: 43, Thu: 40, Wed: 39, Thu: 37
AssignmentId	758	0.66	FALSE	1472	301: 1, 301: 1, 304: 1, 304: 1
WorkerId	758	0.66	FALSE	73	A23: 187, A1S: 129, A3C: 124, AHX
AssignmentStatus	758	0.66	FALSE	1	App: 1472
AcceptTime	758	0.66	FALSE	1457	Thu: 2, Thu: 2, Thu: 2, Thu: 2
SubmitTime	758	0.66	FALSE	1460	Thu: 2, Thu: 2, Thu: 2, Thu: 2
AutoApprovalTime	758	0.66	FALSE	1460	Thu: 2, Thu: 2, Thu: 2, Thu: 2
ApprovalTime	758	0.66	FALSE	929	201: 6, 201: 6, 201: 5, 201: 5
LifetimeApprovalRate	758	0.66	FALSE	32	100: 187, 100: 126, 100: 124, 100: 10
Last30DaysApprovalRate	758	0.66	FALSE	32	100: 187, 100: 126, 100: 124, 100: 10
Last7DaysApprovalRate	758	0.66	FALSE	32	100: 187, 100: 126, 100: 124, 100: 10
URL	758	0.66	FALSE	1450	htt: 2, htt: 2, htt: 2, htt: 2
cats_allowed	0	1.00	FALSE	3	no: 1402, yes: 826, y: 2
common_charges	1684	0.24	FALSE	258	$25: 11, $17: 10, $27: 9, $29: 8
coop_condo	0	1.00	FALSE	2	co-: 1661, con: 569
date_of_sale	1702	0.24	FALSE	222	6/3: 7, 10/: 6, 12/: 6, 2/2: 6
dining_room_type	448	0.80	FALSE	5	com: 957, for: 620, oth: 201, din: 2
dogs_allowed	0	1.00	FALSE	3	no: 1684, yes: 544, yes: 2
fuel_type	112	0.95	FALSE	6	gas: 1348, oil: 664, ele: 62, oth: 40
full_address_or_zip_code	0	1.00	FALSE	1177	70-: 22, 269: 17, 270: 16, 73-: 14
garage_exists	1826	0.18	FALSE	6	yes: 361, Yes: 39, 1: 1, eys: 1
kitchen_type	16	0.99	FALSE	13	eat: 733, eff: 505, com: 349, eff: 338
maintenance_cost	623	0.72	FALSE	609	$54: 10, $67: 10, $68: 10, $70: 10
model_type	40	0.98	FALSE	875	1 B: 63, One: 59, 2 B: 50, Hi-: 41
parking_charges	1671	0.25	FALSE	89	$15: 42, $60: 41, $75: 27, $13: 23
sale_price	1702	0.24	FALSE	315	$15: 11, $17: 10, $13: 7, $22: 7

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
total_taxes	1646	0.26	FALSE	293	$13: 13, $25: 12, $4,: 11, $2,: 10
listing_price_to_nearest_1000	534	0.76	FALSE	292	$34: 28, $39: 26, $28: 25, $23: 23
url	758	0.66	FALSE	1450	htt: 2, htt: 2, htt: 2, htt: 2

Variable type: logical

skim_variable	n_missing	complete_rate	mean	count
Keywords	2230	0	NaN	:
NumberOfSimilarHITs	2230	0	NaN	:
LifetimeInSeconds	2230	0	NaN	:
RejectionTime	2230	0	NaN	:
RequesterFeedback	2230	0	NaN	:

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
MaxAssignments	758	0.66	1.00	0.00	1	1	1	1	1	
AssignmentDurationInSeconds	758	0.66	900.00	0.00	900	900	900	900	900	
AutoApprovalDelayInSeconds	758	0.66	60.00	0.00	60	60	60	60	60	
WorkTimeInSeconds	758	0.66	162.39	111.69	22	89	127	197	815	
approx_year_built	40	0.98	1962.71	21.08	1893	1950	1958	1970	2017	
community_district_num	19	0.99	26.33	2.95	3	25	26	28	32	
num_bedrooms	115	0.95	1.65	0.74	0	1	2	2	6	
num_floors_in_building	650	0.71	7.79	7.52	1	3	6	7	34	
num_full_bathrooms	0	1.00	1.23	0.44	1	1	1	1	3	
num_half_bathrooms	2058	0.08	0.95	0.30	0	1	1	1	2	
num_total_rooms	2	1.00	4.14	1.35	0	3	4	5	14	
pct_tax_deductibl	1754	0.21	45.40	6.95	20	40	50	50	75	
sq_footage	1210	0.46	955.36	380.86	100	743	881	1100	6215	
walk_score	0	1.00	83.92	14.75	7	77	89	95	99	

There is a lot of data that is completely missing and some that is heavily missing. I decided to remove them. Some examples below.

Keywords,NumberOfSimilarHITs, LigetimeInSeconds, RejectionTime,RequesterFeedback all completely missing. ommon_charges(missing 1684),garage_exists(missing 1826)

```r
cat("Data has",nrow(df),"number of rows\n")
```

```
## Data has 2230 number of rows
```

```r
cat("Data has",ncol(df), "number of columns")
```

```
## Data has 55 number of columns
```

```r
sort(colMeans(is.na(df)), decreasing = TRUE)
```

```
##                    Keywords           NumberOfSimilarHITs
##                 1.000000000                   1.000000000
##           LifetimeInSeconds                 RejectionTime
```

```
##                        1.000000000                 1.000000000
##                  RequesterFeedback          num_half_bathrooms
##                        1.000000000                 0.922869955
##                      garage_exists           pct_tax_deductibl
##                        0.818834081                 0.786547085
##                       date_of_sale                  sale_price
##                        0.763228700                 0.763228700
##                     common_charges              parking_charges
##                        0.755156951                 0.749327354
##                        total_taxes                         url
##                        0.738116592                 0.660089686
##                         sq_footage                       HITId
##                        0.542600897                 0.339910314
##                          HITTypeId                       Title
##                        0.339910314                 0.339910314
##                        Description                      Reward
##                        0.339910314                 0.339910314
##                       CreationTime              MaxAssignments
##                        0.339910314                 0.339910314
##                RequesterAnnotation AssignmentDurationInSeconds
##                        0.339910314                 0.339910314
##        AutoApprovalDelayInSeconds                  Expiration
##                        0.339910314                 0.339910314
##                       AssignmentId                    WorkerId
##                        0.339910314                 0.339910314
##                   AssignmentStatus                  AcceptTime
##                        0.339910314                 0.339910314
##                         SubmitTime            AutoApprovalTime
##                        0.339910314                 0.339910314
##                       ApprovalTime           WorkTimeInSeconds
##                        0.339910314                 0.339910314
##               LifetimeApprovalRate     Last30DaysApprovalRate
##                        0.339910314                 0.339910314
##               Last7DaysApprovalRate                        URL
##                        0.339910314                 0.339910314
##             num_floors_in_building            maintenance_cost
##                        0.291479821                 0.279372197
## listing_price_to_nearest_1000             dining_room_type
##                        0.239461883                 0.200896861
##                       num_bedrooms                   fuel_type
##                        0.051569507                 0.050224215
##                   approx_year_built                  model_type
##                        0.017937220                 0.017937220
##             community_district_num                kitchen_type
##                        0.008520179                 0.007174888
##                    num_total_rooms                cats_allowed
##                        0.000896861                 0.000000000
##                         coop_condo                dogs_allowed
##                        0.000000000                 0.000000000
##           full_address_or_zip_code          num_full_bathrooms
##                        0.000000000                 0.000000000
##                         walk_score
##                        0.000000000
```

Data Cleaning Remove all missing y

```
df_drops = df %>% drop_na(sale_price)
# skim(df_drops) %>%
#   summary()
skim(df_drops)
```

Table 5: Data summary

Name	df_drops
Number of rows	528
Number of columns	55
Column type frequency:	
factor	36
logical	5
numeric	14
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
HITId	0	1.00	FALSE	528	301: 1, 302: 1, 302: 1, 307: 1
HITTypeId	0	1.00	FALSE	1	36B: 528, 310: 0
Title	0	1.00	FALSE	1	Fin: 528
Description	0	1.00	FALSE	1	Go : 528, Got: 0
Reward	0	1.00	FALSE	1	$0.: 528
CreationTime	0	1.00	FALSE	21	Wed: 39, Wed: 36, Wed: 33, Wed: 3
RequesterAnnotation	0	1.00	FALSE	1	Bat: 528, Bat: 0
Expiration	0	1.00	FALSE	21	Wed: 39, Wed: 36, Wed: 33, Wed: 3
AssignmentId	0	1.00	FALSE	528	301: 1, 301: 1, 308: 1, 308: 1
WorkerId	0	1.00	FALSE	21	A23: 187, AHX: 102, A1K: 80, A3S:
AssignmentStatus	0	1.00	FALSE	1	App: 528
AcceptTime	0	1.00	FALSE	523	Thu: 2, Thu: 2, Thu: 2, Thu: 2
SubmitTime	0	1.00	FALSE	524	Thu: 2, Thu: 2, Thu: 2, Thu: 2
AutoApprovalTime	0	1.00	FALSE	524	Thu: 2, Thu: 2, Thu: 2, Thu: 2
ApprovalTime	0	1.00	FALSE	337	201: 5, 201: 5, 201: 4, 201: 4
LifetimeApprovalRate	0	1.00	FALSE	15	100: 187, 100: 102, 100: 80, 100: 58
Last30DaysApprovalRate	0	1.00	FALSE	15	100: 187, 100: 102, 100: 80, 100: 58
Last7DaysApprovalRate	0	1.00	FALSE	15	100: 187, 100: 102, 100: 80, 100: 58
URL	0	1.00	FALSE	524	htt: 2, htt: 2, htt: 2, htt: 2
cats_allowed	0	1.00	FALSE	2	no: 285, yes: 243, y: 0
common_charges	396	0.25	FALSE	112	$27: 3, $31: 3, $21: 2, $24: 2
coop_condo	0	1.00	FALSE	2	co-: 399, con: 129
date_of_sale	0	1.00	FALSE	222	6/3: 7, 10/: 6, 12/: 6, 2/2: 6
dining_room_type	120	0.77	FALSE	4	com: 241, for: 116, oth: 49, din: 2
dogs_allowed	0	1.00	FALSE	2	no: 381, yes: 147, yes: 0
fuel_type	24	0.95	FALSE	6	gas: 301, oil: 180, ele: 11, oth: 8
full_address_or_zip_code	0	1.00	FALSE	468	70-: 8, 54-: 4, 104: 3, 117: 3
garage_exists	434	0.18	FALSE	6	yes: 51, Yes: 39, 1: 1, eys: 1
kitchen_type	6	0.99	FALSE	7	eff: 231, eat: 190, Com: 50, com: 31

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
maintenance_cost	142	0.73	FALSE	284	$52: 4, $60: 4, $66: 4, $67: 4
model_type	15	0.97	FALSE	356	1 B: 23, One: 19, 2 B: 11, Gar: 11
parking_charges	393	0.26	FALSE	50	$10: 12, $20: 10, $95: 8, $12: 7
sale_price	0	1.00	FALSE	315	$15: 11, $17: 10, $13: 7, $22: 7
total_taxes	397	0.25	FALSE	120	$2,: 3, $4,: 3, $1,: 2, $1,: 2
listing_price_to_nearest_1000	528	0.00	FALSE	0	$1,: 0, $10: 0, $10: 0, $10: 0
url	0	1.00	FALSE	524	htt: 2, htt: 2, htt: 2, htt: 2

Variable type: logical

skim_variable	n_missing	complete_rate	mean	count
Keywords	528	0	NaN	:
NumberOfSimilarHITs	528	0	NaN	:
LifetimeInSeconds	528	0	NaN	:
RejectionTime	528	0	NaN	:
RequesterFeedback	528	0	NaN	:

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
MaxAssignments	0	1.00	1.00	0.00	1	1	1	1	1	
AssignmentDurationInSeconds	0	1.00	900.00	0.00	900	900	900	900	900	
AutoApprovalDelayInSeconds	0	1.00	60.00	0.00	60	60	60	60	60	
WorkTimeInSeconds	0	1.00	150.38	96.99	52	97	123	163	815	
approx_year_built	6	0.99	1962.38	20.56	1915	1950	1957	1968	2016	
community_district_num	1	1.00	26.30	2.99	3	25	26	28	30	
num_bedrooms	0	1.00	1.54	0.75	0	1	1	2	3	
num_floors_in_building	108	0.80	7.08	6.83	1	2	6	7	34	
num_full_bathrooms	0	1.00	1.20	0.42	1	1	1	1	3	
num_half_bathrooms	498	0.06	1.03	0.18	1	1	1	1	2	
num_total_rooms	0	1.00	4.02	1.20	1	3	4	5	8	
pct_tax_deductibl	429	0.19	44.99	8.09	20	40	50	50	65	
sq_footage	315	0.40	965.28	490.42	375	750	874	1010	6215	
walk_score	0	1.00	83.10	13.09	15	76	85	94	99	

Meaningful Features Data Cleaning

Finding meaningful features. These are features I believe are meaningful. df_mutated has all the features that I will be using. I am not looking at whats missing yet or how the data looks like, just looking for features that would be best to predict sales price.

```
colnames(df)
```

```
##  [1] "HITId"                        "HITTypeId"
##  [3] "Title"                        "Description"
##  [5] "Keywords"                     "Reward"
##  [7] "CreationTime"                 "MaxAssignments"
##  [9] "RequesterAnnotation"          "AssignmentDurationInSeconds"
## [11] "AutoApprovalDelayInSeconds"   "Expiration"
```

```
## [13] "NumberOfSimilarHITs"           "LifetimeInSeconds"
## [15] "AssignmentId"                  "WorkerId"
## [17] "AssignmentStatus"              "AcceptTime"
## [19] "SubmitTime"                    "AutoApprovalTime"
## [21] "ApprovalTime"                  "RejectionTime"
## [23] "RequesterFeedback"             "WorkTimeInSeconds"
## [25] "LifetimeApprovalRate"          "Last30DaysApprovalRate"
## [27] "Last7DaysApprovalRate"         "URL"
## [29] "approx_year_built"             "cats_allowed"
## [31] "common_charges"                "community_district_num"
## [33] "coop_condo"                    "date_of_sale"
## [35] "dining_room_type"              "dogs_allowed"
## [37] "fuel_type"                     "full_address_or_zip_code"
## [39] "garage_exists"                 "kitchen_type"
## [41] "maintenance_cost"              "model_type"
## [43] "num_bedrooms"                  "num_floors_in_building"
## [45] "num_full_bathrooms"            "num_half_bathrooms"
## [47] "num_total_rooms"               "parking_charges"
## [49] "pct_tax_deductibl"             "sale_price"
## [51] "sq_footage"                    "total_taxes"
## [53] "walk_score"                    "listing_price_to_nearest_1000"
## [55] "url"
```

```
df_mutated = copy(df_drops)
df_mutated %<>%
  select(cats_allowed,common_charges,coop_condo,dining_room_type,dogs_allowed,fuel_type,garage_exists,ma
sort(colMeans(is.na(df_mutated)), decreasing = TRUE)
```

```
##          garage_exists            total_taxes         common_charges
##            0.821969697            0.751893939            0.750000000
##             sq_footage       maintenance_cost       dining_room_type
##            0.596590909            0.268939394            0.227272727
## num_floors_in_building              fuel_type             model_type
##            0.204545455            0.045454545            0.028409091
##      approx_year_built community_district_num           cats_allowed
##            0.011363636            0.001893939            0.000000000
##             coop_condo           dogs_allowed           num_bedrooms
##            0.000000000            0.000000000            0.000000000
##     num_full_bathrooms        num_total_rooms             sale_price
##            0.000000000            0.000000000            0.000000000
##             walk_score
##            0.000000000
```

Feature Data Cleaning

I am now looking more closely to the data.Looking at this there are too many types of model_types 875 different times from original data with NA sale price this seems difficult to deal with so I will remove this. I discarded data with more than 50% of missinginess.

```
df_mutated_features = copy(df_mutated)
df_mutated_features %<>%
  select(-model_type,-total_taxes, -community_district_num)#,-common_charges,-sq_footage)
skim(df_mutated_features) %>%
  summary()
```

Table 9: Data summary

Name	df_mutated_features
Number of rows	528
Number of columns	16
Column type frequency:	
factor	9
numeric	7
Group variables	None

```r
sort(colMeans(is.na(df_mutated_features)), decreasing = TRUE)
```

```
##          garage_exists         common_charges             sq_footage
##             0.82196970             0.75000000             0.59659091
##       maintenance_cost      dining_room_type num_floors_in_building
##             0.26893939             0.22727273             0.20454545
##              fuel_type      approx_year_built           cats_allowed
##             0.04545455             0.01136364             0.00000000
##             coop_condo           dogs_allowed           num_bedrooms
##             0.00000000             0.00000000             0.00000000
##     num_full_bathrooms        num_total_rooms             sale_price
##             0.00000000             0.00000000             0.00000000
##             walk_score
##             0.00000000
```

Oberservations Data Cleaning

I am okay with the number of features I have now. Now Ill be cleaning the observations.

```r
df_clean = copy(df_mutated_features)


# Fixing y to be just yes and reducing factors to just yes and no.
df_clean %<>%
  mutate(cats_allowed = as.factor(ifelse(cats_allowed =='y' | cats_allowed =='yes','yes','no'))) %>%

#Fixing yes89 to just yes and reducing factors to just yes and no
  mutate(dogs_allowed = as.factor(ifelse(dogs_allowed =='yes89' | dogs_allowed =='yes','yes','no'))) %>

  #mutate(sale_price = as.numeric(gsub('[$]','',as.character(df_clean$sale_price))))
  mutate(sale_price = as.numeric(gsub('[$,]','',as.character(df_clean$sale_price))) )%>%

  mutate(common_charges = as.numeric(gsub('[$,]','',as.character(df_clean$common_charges)))) %>%

  mutate(maintenance_cost = as.numeric(gsub('[$,]','',as.character(df_clean$maintenance_cost)))) %>%

  mutate(garage_exists = ifelse(is.na(garage_exists), 0, 1))

  #mutate(fuel_type = if(is.na(fuel_type)){fuel_type = 'other'})
```

```r
  #Very annoying this best way I found to combine two factor lvels


library(forcats)
df_clean$fuel_type = fct_collapse(df_clean$fuel_type, other = c("other","Other"))
#df_clean$dining_room_type = fct_collapse(df_clean$dining_room_type, other= c('other','none','dining ar



# df_clean_sub = copy(df_clean)
# df_clean_sub = df_clean_sub[df_clean_sub$sale_price < 700000,]
#
# df_clean = df_clean_sub
 #df_clean = df_clean[df_clean$sq_footage < 2500,]
#which(df_clean$sq_footage > 2500)

#df_clean = df_clean[-136,]
```

```r
options(scipen=999)
max(df_clean$sq_footage, na.rm = TRUE)
```

```
## [1] 6215
```

```r
min(df_clean$sq_footage, na.rm = TRUE)
```

```
## [1] 375
```

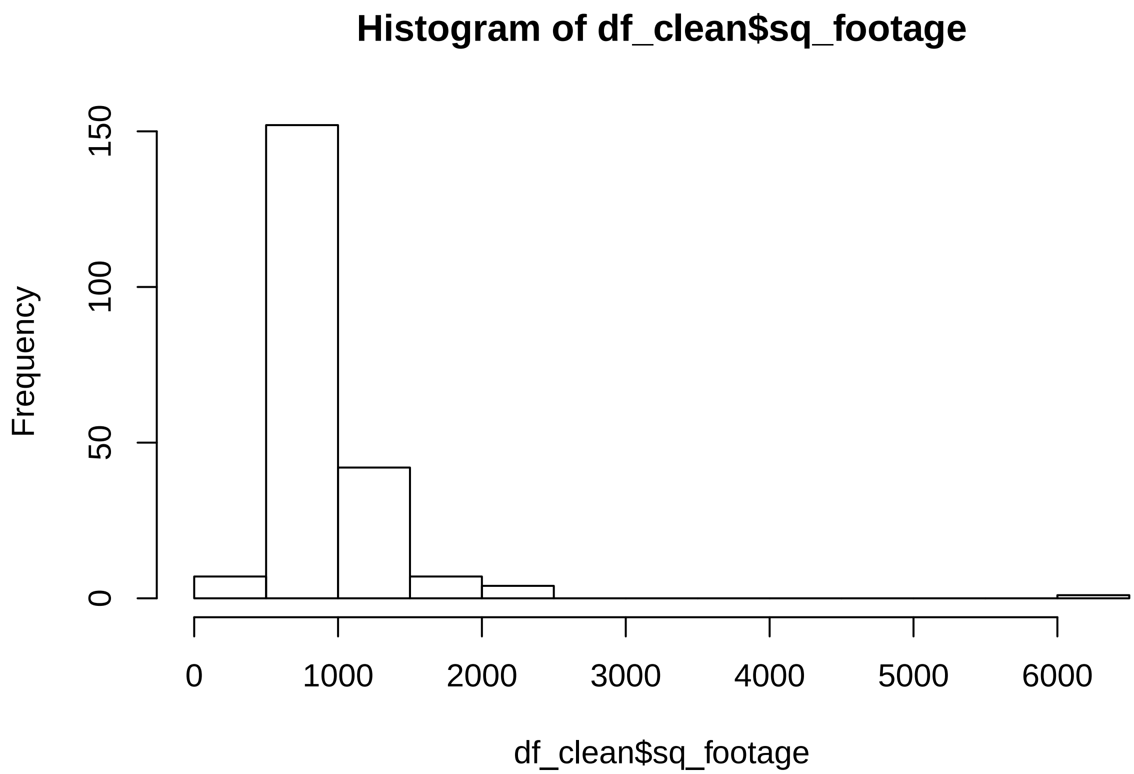
```r
max(df_clean$sale_price, na.rm = TRUE)
```

```
## [1] 999999
```

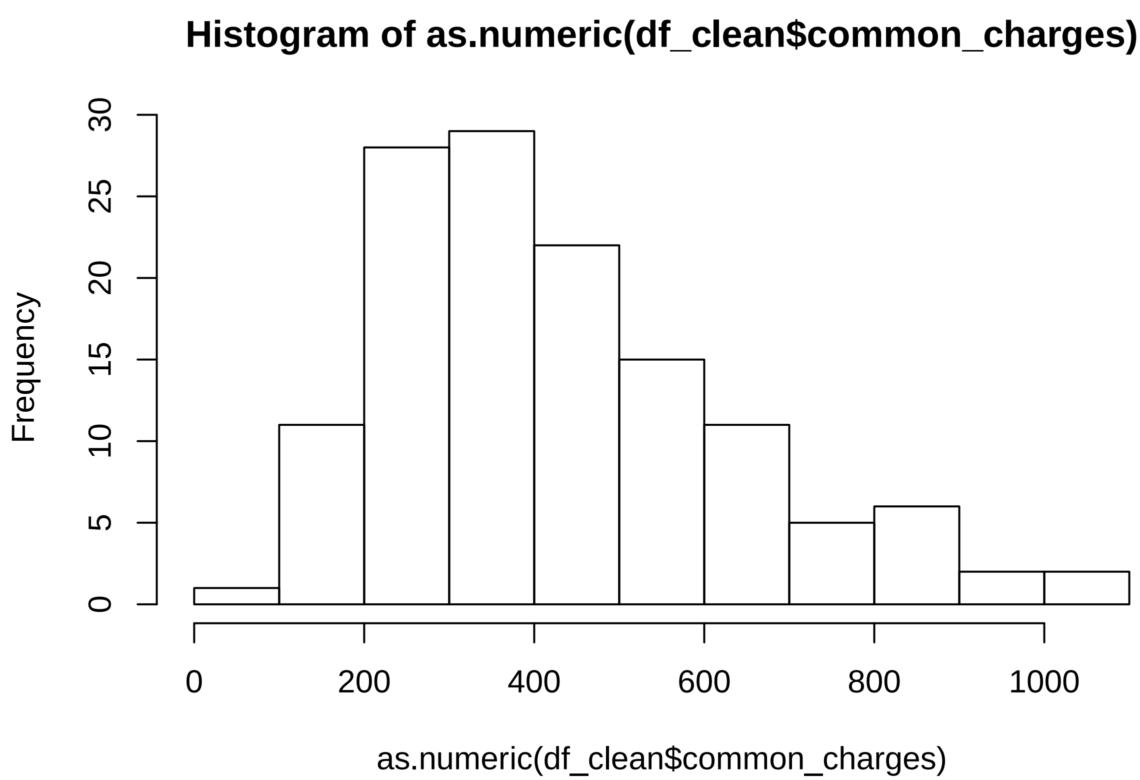
```r
min(df_clean$sale_price, na.rm = TRUE)
```

```
## [1] 55000
```

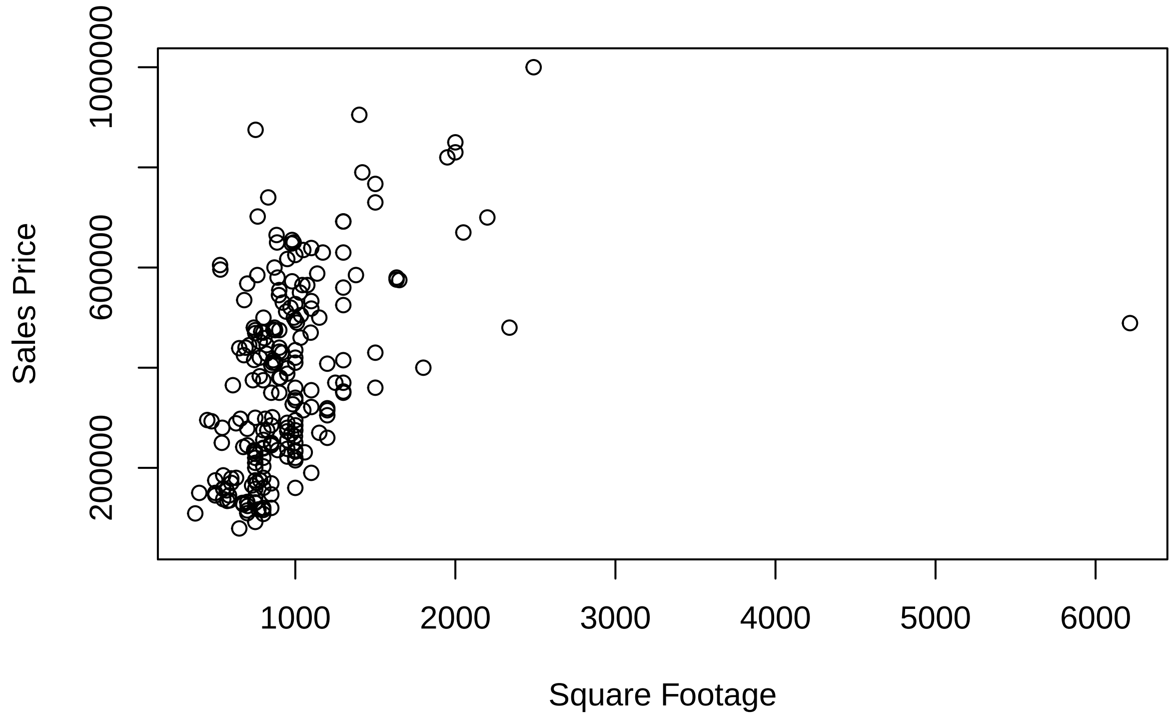
```r
hist(df_clean$sq_footage)
```




```
hist(as.numeric(df_clean$common_charges))
```




```
plot(y=df_clean$sale_price,df_clean$sq_footage, xlab ='Square Footage', ylab= 'Sales Price',)
```




```
library(ggplot2)
#plot(y=df_final$sale_price,df_clean$approx_year_built, xlab ='Approx Year Built', ylab= 'Sales Price',
#geom_rug(sides ="bl")
```

```
#df_clean %<>%
  #select(-sq_footage)
```

```
#Fix issue when using MLR, Notice that there is missingness so ill set it to other
df_clean$fuel_type[df_clean$fuel_type == 'none'] = 'other'
df_clean$fuel_type[is.na(df_clean$fuel_type)] = 'other'
df_clean$fuel_type = factor(df_clean$fuel_type)
#df_cleandd = df_clean[df_clean$sq_footage < 2500,]
#df_clean = subset(df_clean, df_clean$sq_footage <= 2500 & !is.na(df_clean$sq_footage))
#df_clean$num_full_bathrooms = as.factor(df_clean$num_full_bathrooms)
skim(df_clean)
```

Table 10: Data summary

Name	df_clean
Number of rows	528
Number of columns	16
Column type frequency:	
factor	5
numeric	11
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
cats_allowed	0	1.00	FALSE	2	no: 285, yes: 243
coop_condo	0	1.00	FALSE	2	co-: 399, con: 129
dining_room_type	120	0.77	FALSE	4	com: 241, for: 116, oth: 49, din: 2
dogs_allowed	0	1.00	FALSE	2	no: 381, yes: 147
fuel_type	0	1.00	FALSE	4	gas: 301, oil: 180, oth: 36, ele: 11

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p7
common_charges	396	0.25	433.92	205.40	70	288.50	390.5	537.7
garage_exists	0	1.00	0.18	0.38	0	0.00	0.0	0.0
maintenance_cost	142	0.73	821.85	378.77	155	639.25	734.0	880.0
approx_year_built	6	0.99	1962.38	20.56	1915	1950.00	1957.0	1968.0
num_bedrooms	0	1.00	1.54	0.75	0	1.00	1.0	2.0
num_floors_in_building	108	0.80	7.08	6.83	1	2.00	6.0	7.0
num_full_bathrooms	0	1.00	1.20	0.42	1	1.00	1.0	1.0
num_total_rooms	0	1.00	4.02	1.20	1	3.00	4.0	5.0
sq_footage	315	0.40	965.28	490.42	375	750.00	874.0	1010.0
sale_price	0	1.00	314956.56	179526.60	55000	171500.00	259500.0	428875.0
walk_score	0	1.00	83.10	13.09	15	76.00	85.0	94.0

```
set.seed(28)
M = tbl_df(apply(is.na(df_clean), 2, as.numeric))
colnames(M) = paste("is_missing_", colnames(df_clean), sep = "")
M %<>%
  select_if(function(x){sum(x) > 0})
head(M)
```

```
## # A tibble: 6 x 6
##   is_missing_comm~ is_missing_dini~ is_missing_main~ is_missing_appr~
##              <dbl>            <dbl>            <dbl>            <dbl>
## 1                0                0                1                0
## 2                1                0                0                0
## 3                0                0                1                0
## 4                0                0                1                0
## 5                1                0                0                0
## 6                1                0                0                0
## # ... with 2 more variables: is_missing_num_floors_in_building <dbl>,
## #   is_missing_sq_footage <dbl>
```

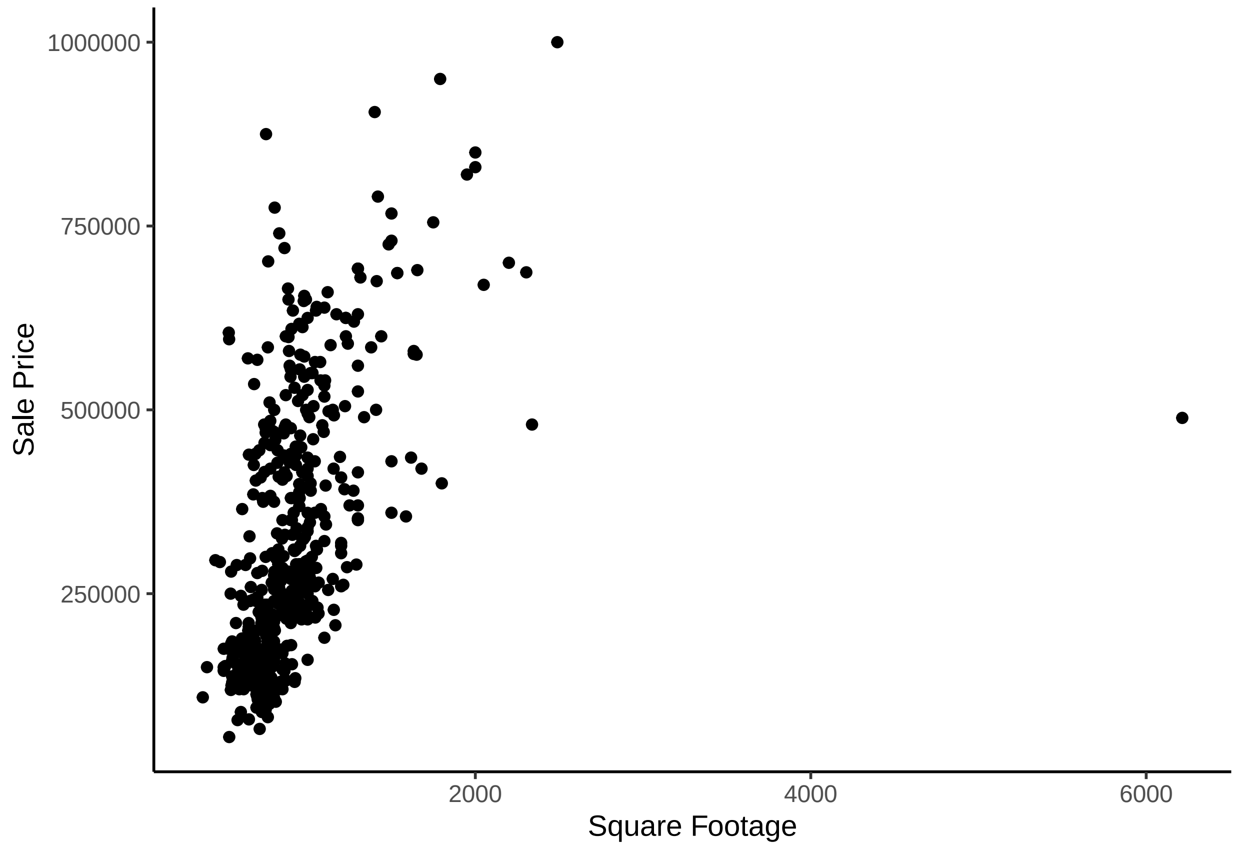
```
pacman::p_load(missForest)
dfimp = missForest(data.frame(df_clean))$ximp
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
##   missForest iteration 4 in progress...done!
##   missForest iteration 5 in progress...done!
##   missForest iteration 6 in progress...done!
##   missForest iteration 7 in progress...done!
```

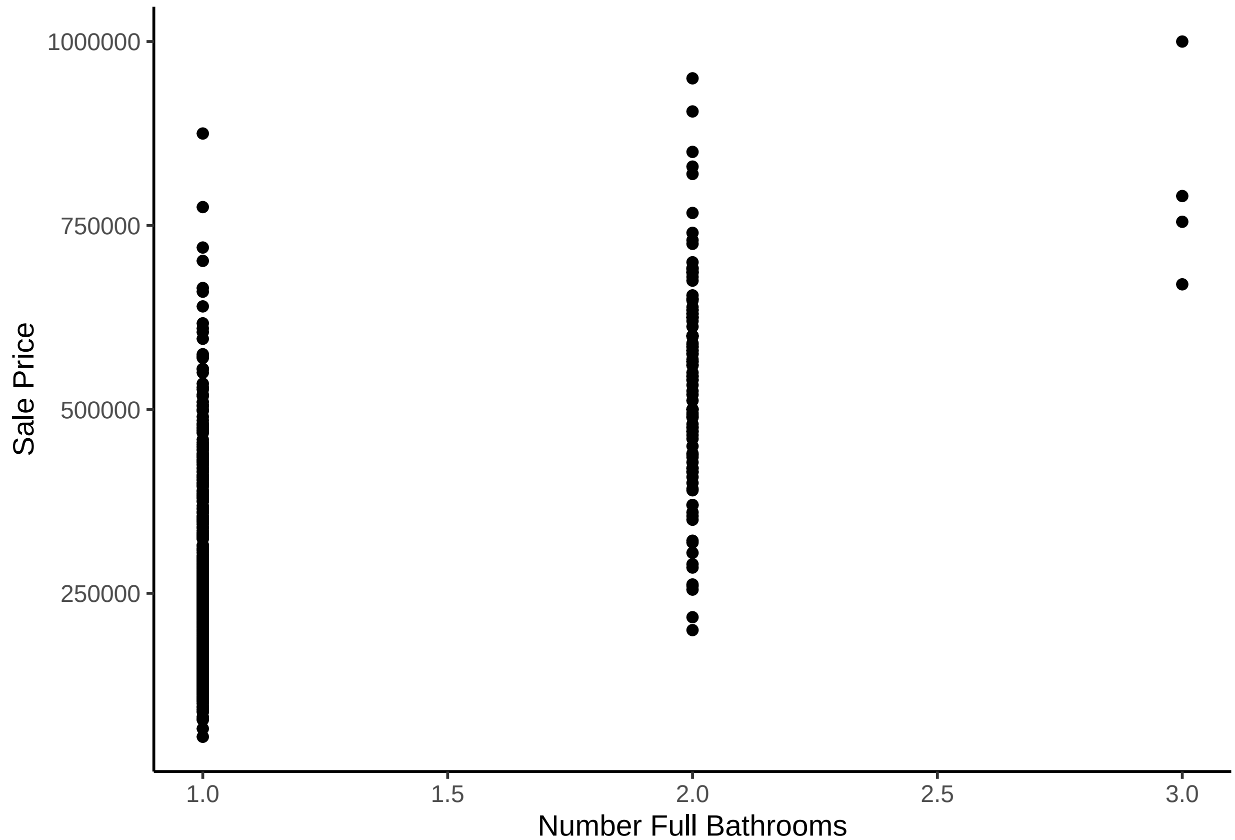
```r
df_final = cbind(dfimp, M)
```

```r
#skim(df_final)
```

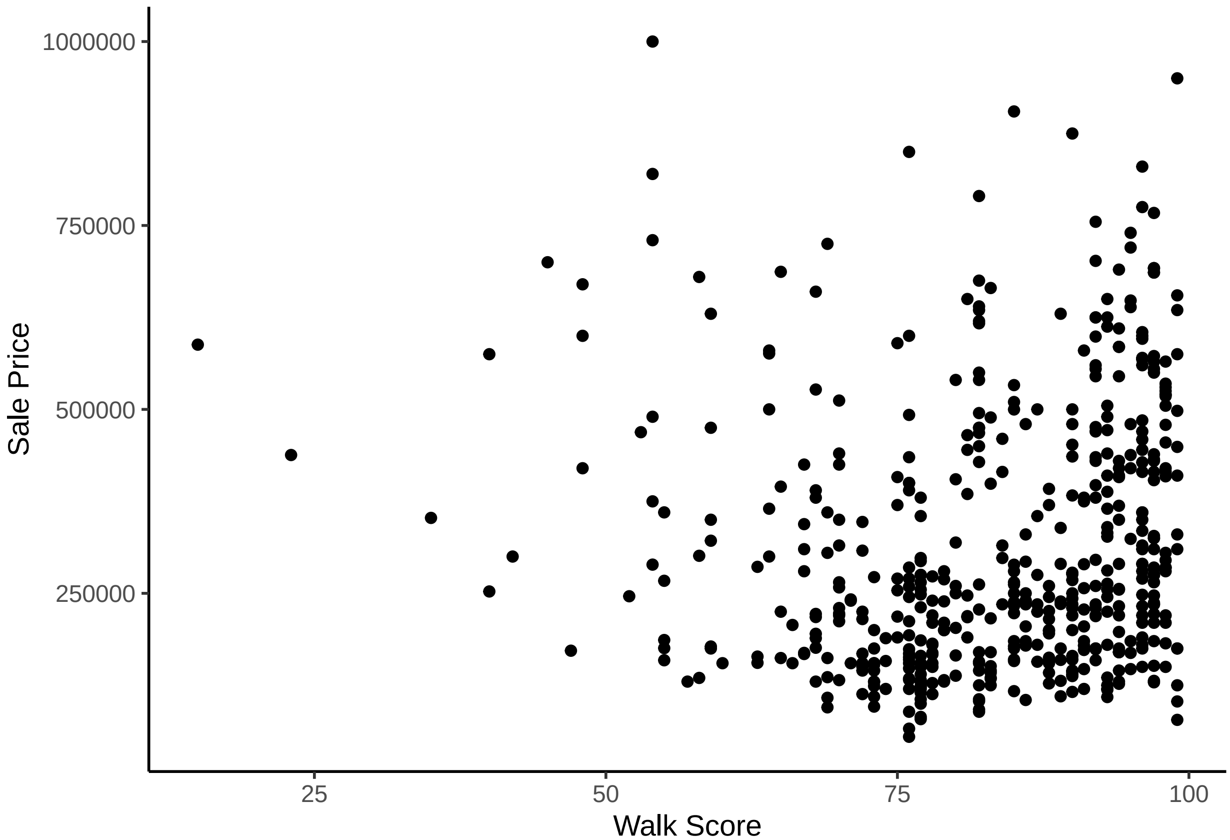
```r
ggplot(df_final, aes(x=sq_footage, y=sale_price)) + geom_point() +  theme_classic()  +
  labs(
       x="Square Footage", y = "Sale Price")
```



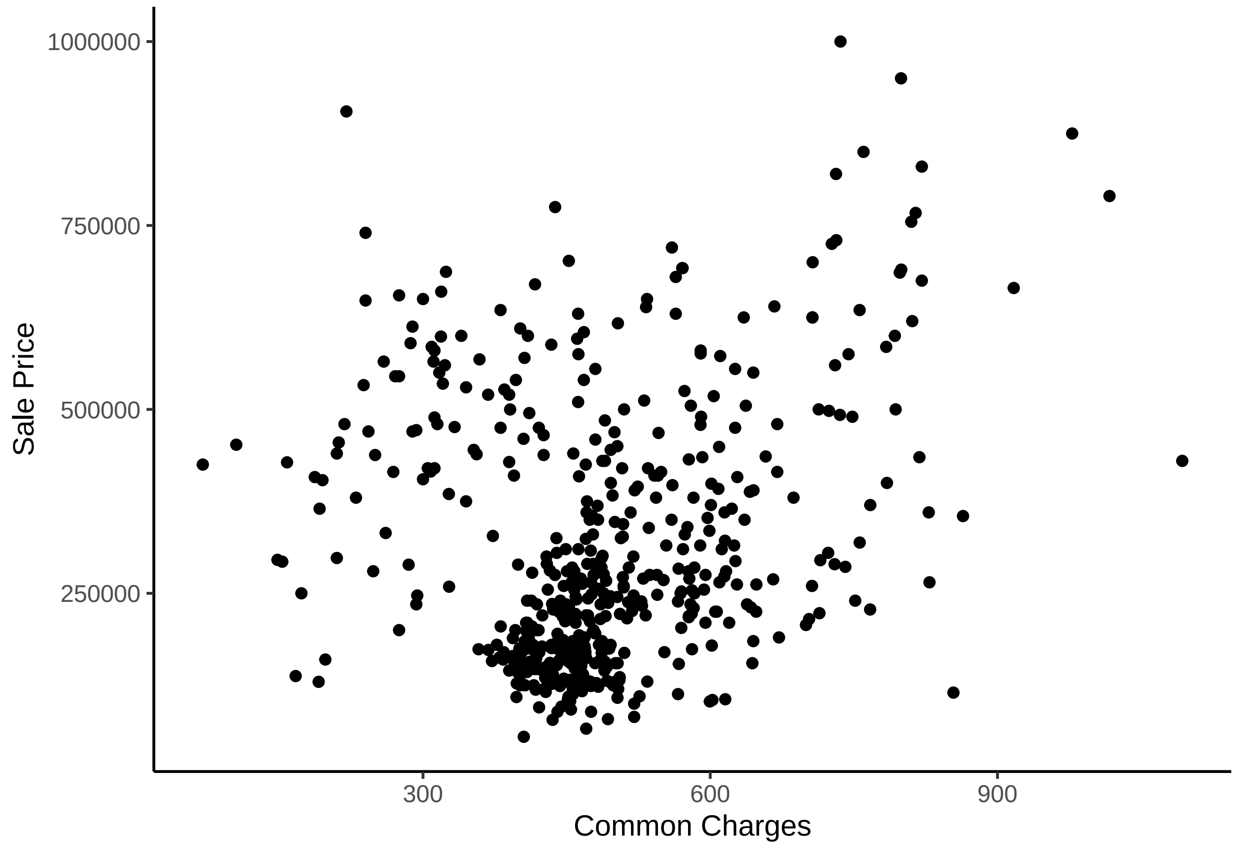

```r
ggplot(df_final, aes(x=num_full_bathrooms, y=sale_price)) + geom_point() +  theme_classic()  +
  labs(
       x="Number Full Bathrooms", y = "Sale Price")
```



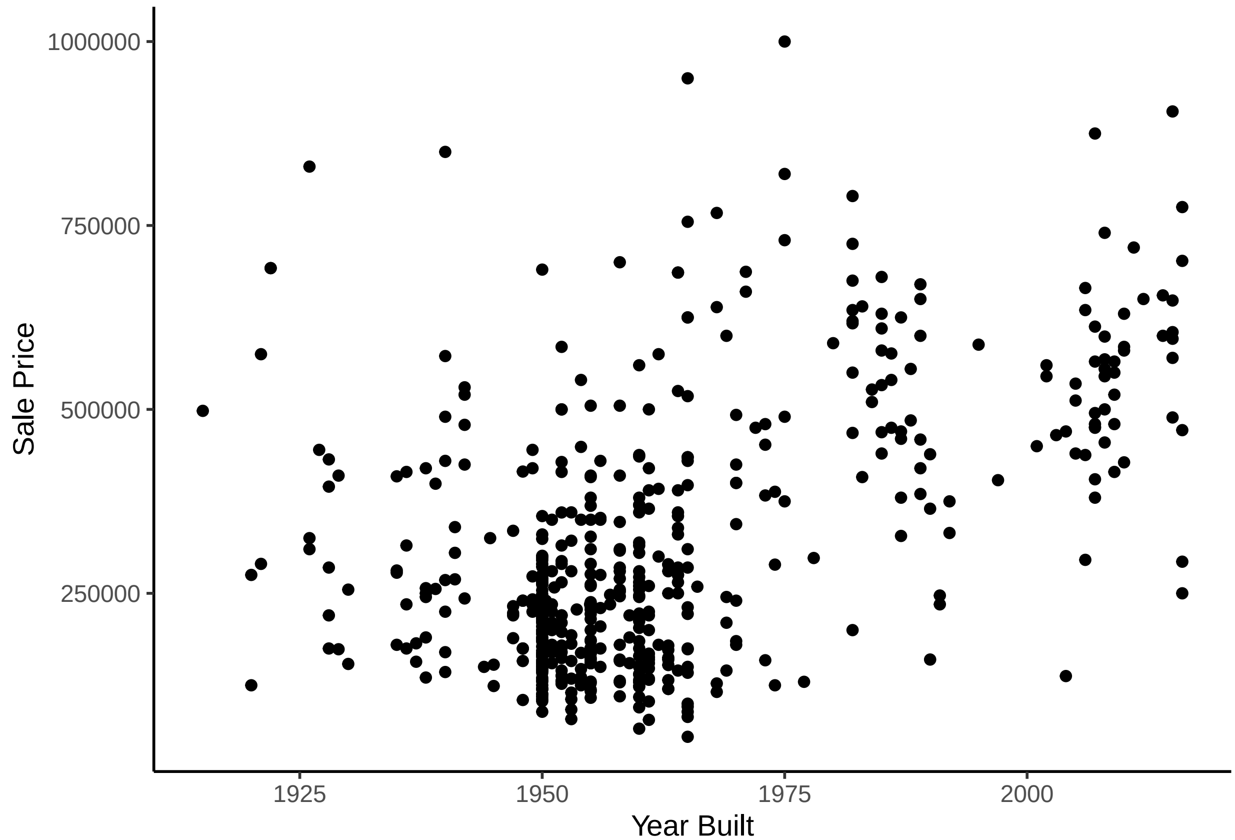

```
ggplot(df_final, aes(x=walk_score, y=sale_price)) + geom_point() +  theme_classic()  +
  labs(
       x="Walk Score", y = "Sale Price")
```



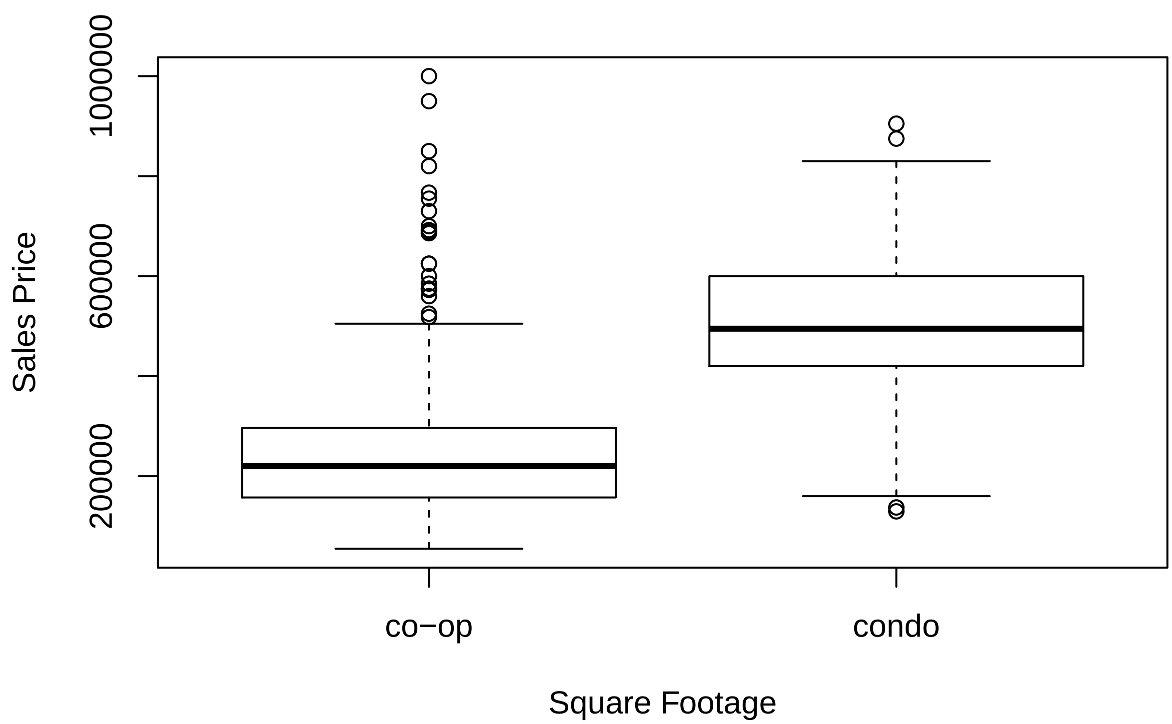

```
ggplot(df_final, aes(x=common_charges, y=sale_price)) + geom_point() +  theme_classic()  +
  labs(
       x="Common Charges", y = "Sale Price")
```




```
ggplot(df_final, aes(x=approx_year_built, y=sale_price)) + geom_point() +  theme_classic()   +
  labs(
        x="Year Built", y = "Sale Price")
```




```
plot(y=df_clean$sale_price,df_clean$coop_condo, xlab ='Square Footage', ylab= 'Sales Price',)
```



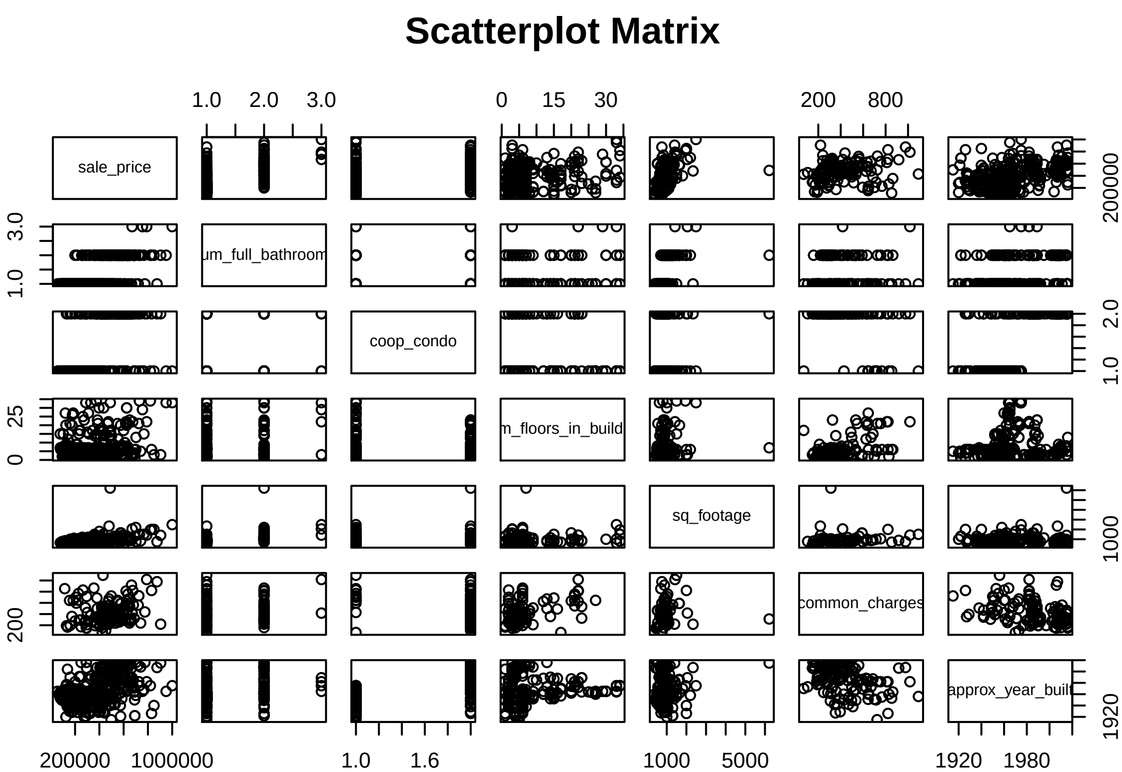

```
# Tried to one hot incode data. MAJOR FAIL. R takes care of this since data is already factors
# df_dummy = copy(df_final)
```

```r
# df_dummy$cats_allowed = model.matrix(~df_dummy$cats_allowed + 0)
# df_dummy$coop_condo = model.matrix(~df_dummy$coop_condo + 0)
# df_dummy$dining_room_type = model.matrix(~df_dummy$dining_room_type + 0)
# df_dummy$dogs_allowed = model.matrix(~df_dummy$dogs_allowed + 0)
# df_dummy$fuel_type = model.matrix(~df_dummy$fuel_type + 0)
# library(data.table,mltools)
# something = copy(df_final)
#
# something$fuel_type = cbind(model.matrix(~something$fuel_type))
```

```r
colnames(df_final)
```

```
##  [1] "cats_allowed"                     "common_charges"
##  [3] "coop_condo"                       "dining_room_type"
##  [5] "dogs_allowed"                     "fuel_type"
##  [7] "garage_exists"                    "maintenance_cost"
##  [9] "approx_year_built"                "num_bedrooms"
## [11] "num_floors_in_building"           "num_full_bathrooms"
## [13] "num_total_rooms"                  "sq_footage"
## [15] "sale_price"                       "walk_score"
## [17] "is_missing_common_charges"        "is_missing_dining_room_type"
## [19] "is_missing_maintenance_cost"      "is_missing_approx_year_built"
## [21] "is_missing_num_floors_in_building" "is_missing_sq_footage"
```

```r
pairs(~sale_price+num_full_bathrooms+coop_condo+num_floors_in_building+sq_footage+common_charges+approx_
  main="Scatterplot Matrix")
```




```r
skim(df_final)
```

Table 13: Data summary

Name	df_final
Number of rows	528
Number of columns	22
Column type frequency:	
factor	5
numeric	17
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
cats_allowed	0	1	FALSE	2	no: 285, yes: 243
coop_condo	0	1	FALSE	2	co-: 399, con: 129
dining_room_type	0	1	FALSE	4	com: 332, for: 139, oth: 55, din: 2
dogs_allowed	0	1	FALSE	2	no: 381, yes: 147
fuel_type	0	1	FALSE	4	gas: 301, oil: 180, oth: 36, ele: 11

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p
common_charges	0	1	489.07	138.73	70	417.43	469.
garage_exists	0	1	0.18	0.38	0	0.00	0.
maintenance_cost	0	1	810.71	361.55	155	605.50	722.
approx_year_built	0	1	1962.25	20.48	1915	1950.00	1956.
num_bedrooms	0	1	1.54	0.75	0	1.00	1.
num_floors_in_building	0	1	7.08	6.33	1	3.00	6.
num_full_bathrooms	0	1	1.20	0.42	1	1.00	1.
num_total_rooms	0	1	4.02	1.20	1	3.00	4.
sq_footage	0	1	901.76	364.21	375	722.87	835.
sale_price	0	1	314956.56	179526.60	55000	171500.00	259500.
walk_score	0	1	83.10	13.09	15	76.00	85.
is_missing_common_charges	0	1	0.75	0.43	0	0.75	1.
is_missing_dining_room_type	0	1	0.23	0.42	0	0.00	0.
is_missing_maintenance_cost	0	1	0.27	0.44	0	0.00	0.
is_missing_approx_year_built	0	1	0.01	0.11	0	0.00	0.
is_missing_num_floors_in_building	0	1	0.20	0.40	0	0.00	0.
is_missing_sq_footage	0	1	0.60	0.49	0	0.00	1.

```r
set.seed(28)
train.control <- trainControl(method = "repeatedcv",
                              number = 10, repeats = 3)
prop_test = 0.10
test_indices = sample(1 : nrow(df_final), round((prop_test) * nrow(df_final)))
df_test = df_final[test_indices, ]
y_test = df_test$sale_price
X_test = cbind(1, df_test)
```

```r
#X_test$sale_price = NULL


train_indices = setdiff(1 : nrow(df_final), test_indices)
df_train = df_final[train_indices, ]
y_train = df_train$sale_price
X_train = cbind(1, df_train)
#X_train$sale_price = NULL

n_train = nrow(X_train)
```

```r
#mod = train(sale_price ~ ., df_final, trControl = train.control,method = "lm")
mod = lm(sale_price ~ ., df_final)
summary(mod)$r.squared
```

```
## [1] 0.77106
```

```r
summary(mod)$sigma
```

```
## [1] 88012.24
```

```r
summary(mod)$sigma
```

```
## [1] 88012.24
```

```r
set.seed(28)
mod =lm(sale_price ~., data.frame(df_train),set.seed(28))
summary(mod)$r.squared
```

```
## [1] 0.7649951
```

```r
summary(mod)$sigma
```

```
## [1] 88712.26
```

```r
y_hat = predict(mod,data.frame(X_test))
e = y_test - y_hat
Rsq_oos = (var(y_test) - var(e)) / var(y_test)

cat("My R Squared in sample is ",summary(mod)$r.squared, "My RSME is:", summary(mod)$sigma)
```

```
## My R Squared in sample is  0.7649951 My RSME is: 88712.26
```

```r
cat("\nMy R Squared out of sample is ",Rsq_oos, "My RSME is:", sd(e))
```

```
##
## My R Squared out of sample is  0.8046578 My RSME is: 85564.51
```

```r
#plot(y_test,y_hat)
summary(mod)$coefficients
```

```
##                                      Estimate   Std. Error     t value
## (Intercept)                      -298395.10004 659717.39749 -0.45230746
## cats_allowedyes                     9080.70174  11575.96889  0.78444421
## common_charges                        31.45106     52.66910  0.59714451
## coop_condocondo                   268580.85658  25419.09719 10.56610526
## dining_room_typedining area        23700.90178  89632.03081  0.26442446
## dining_room_typeformal              7681.67180  11135.25553  0.68985142
## dining_room_typeother              32171.80378  14691.73745  2.18978891
```

```
## dogs_allowedyes                     19982.52480  12690.44037  1.57461240
## fuel_typegas                        -3832.20897  29743.63410 -0.12884132
## fuel_typeoil                        -4827.69028  30620.41595 -0.15766247
## fuel_typeother                       5305.86760  33823.14796  0.15687090
## garage_exists                         802.16809  12224.26107  0.06562099
## maintenance_cost                      121.28753     26.86624  4.51449583
## approx_year_built                      33.65822    336.41994  0.10004822
## num_bedrooms                        54268.57388  10137.46922  5.35326645
## num_floors_in_building               4939.69734    918.37444  5.37873998
## num_full_bathrooms                  82887.84110  14540.51399  5.70047532
## num_total_rooms                       378.61425   6557.41605  0.05773833
## sq_footage                             23.78621     16.31750  1.45771128
## walk_score                           1134.48219    336.43376  3.37208186
## is_missing_common_charges           42305.47657  25406.07304  1.66517181
## is_missing_dining_room_type          7221.66106  10162.44063  0.71062271
## is_missing_maintenance_cost        -26899.32499  22966.63777 -1.17123478
## is_missing_approx_year_built        23817.48782  37540.64641  0.63444533
## is_missing_num_floors_in_building   14964.93013  10575.49891  1.41505666
## is_missing_sq_footage              -15340.57219   8925.11107 -1.71881023
##                                                          Pr(>|t|)
## (Intercept)                       0.651265944570144084124763139700
## cats_allowedyes                   0.433193164251291951138966851430
## common_charges                    0.550711901176169194016551955430
## coop_condocondo                   0.000000000000000000000001868493
## dining_room_typedining area       0.791574194208011006068659298760
## dining_room_typeformal            0.490644019556724209074616283030
## dining_room_typeother             0.029053391015062355468012711410
## dogs_allowedyes                   0.116050256420487843400657368420
## fuel_typegas                      0.897540903681133706015771167590
## fuel_typeoil                      0.874793655357398280969505321990
## fuel_typeother                    0.875417108445347036393968664920
## garage_exists                     0.947708785772300998928585613610
## maintenance_cost                  0.000008119343805649964699428480
## approx_year_built                 0.920350700630993445727767721110
## num_bedrooms                      0.000000138097342205014205699100
## num_floors_in_building            0.000000120943931183074434272820
## num_full_bathrooms                0.000000021672360783895491256320
## num_total_rooms                   0.953982747816195852941234534230
## sq_footage                        0.145619195490151648320065191910
## walk_score                        0.000810572945462633060766111100
## is_missing_common_charges         0.096576038379283746238002095200
## is_missing_dining_room_type       0.477687014966515555158821371150
## is_missing_maintenance_cost       0.242125592297690900300821681410
## is_missing_approx_year_built      0.526113351066712864323449139190
## is_missing_num_floors_in_building 0.157745044275308193837403791800
## is_missing_sq_footage             0.086338212313594620006540480970
```

```r
sort(summary(mod)$coefficients)
```

```
##   [1] -298395.100035177893005311489105224610
##   [2]  -26899.324988928445236524567008018490
##   [3]  -15340.572192303185147466138005256650
##   [4]   -4827.690280038509627047460526227950
##   [5]   -3832.208971922430009726667776703830
##   [6]      -1.718810228396836059516772365900
```

```
##  [7]      -1.17123478240953704521132294758
##  [8]      -0.45230745947942546658637752444
##  [9]      -0.15766246573039013889605541863
## [10]      -0.12884131641719678840196650071
## [11]       0.00000000000000000000001868493
## [12]       0.00000002167236078389549125632
## [13]       0.00000012094393118307443427282
## [14]       0.00000013809734220501420569910
## [15]       0.00000811934380564996469942848
## [16]       0.00081057294546263306076611110
## [17]       0.02905339101506235546801271141
## [18]       0.05773832994050270145391934307
## [19]       0.06562098834916842149400650896
## [20]       0.08633821231359462000654048097
## [21]       0.09657603837928374623800209520
## [22]       0.10004821760408874609105112086
## [23]       0.11605025642048784340065736842
## [24]       0.14561919549015164832006519191
## [25]       0.15687089811831364527527910013
## [26]       0.15774504427530819383740379180
## [27]       0.24212559229769090030082168141
## [28]       0.26442446478970765832983147448
## [29]       0.43319316425129195113896685143
## [30]       0.47768701496651555515882137115
## [31]       0.49064401955672420907461628303
## [32]       0.52611335106671286432344913919
## [33]       0.55071190117616919401655195543
## [34]       0.59714450991216483366486045270
## [35]       0.63444533058002450243151315590
## [36]       0.65126594457014408412476313970
## [37]       0.68985141639581148975679525392
## [38]       0.71062270619985379305205697165
## [39]       0.78444420772045220235924034569
## [40]       0.79157419420801100606865929876
## [41]       0.87479365535739828096950532199
## [42]       0.87541710844534703639396866492
## [43]       0.89754090368113370601577116759
## [44]       0.92035070063099344572776772111
## [45]       0.94770878577230099892858561361
## [46]       0.95398274781619585294123453423
## [47]       1.41505665740450825573759630061
## [48]       1.45771128142618344725178758381
## [49]       1.57461240156352300090247808839
## [50]       1.66517180727883817858980819437
## [51]       2.18978891278662723962611380557
## [52]       3.37208185790195003050939703826
## [53]       4.51449583345036398185357029433
## [54]       5.35326644953967356599378035753
## [55]       5.37873997553210880795404591481
## [56]       5.70047531579415878155714381137
## [57]      10.56610526001139049867560970597
## [58]      16.31750256262787246441803290509
## [59]      23.78620757024330956141966453288
## [60]      26.86624107365656755064264871180
```

```
##  [61]      31.45106224537411065966807655059
##  [62]      33.65821516798138191006728447974
##  [63]      52.66909721735582650126161752269
##  [64]     121.28753338749559986808890243992
##  [65]     336.41993804601116835328866727650
##  [66]     336.43376249591602800137479789555
##  [67]     378.61425158612763652854482643306
##  [68]     802.16809356022804422536864876747
##  [69]     918.37444519046392088057473301 89
##  [70]    1134.48218689817190352187026292086
##  [71]    4939.69733724168963817646726965904
##  [72]    5305.86759736097155837342143058777
##  [73]    6557.41605232219535537296906113625
##  [74]    7221.66106224240047595230862498283
##  [75]    7681.67179779537946160417050123215
##  [76]    8925.11106744553217140492051839828
##  [77]    9080.70174103995486802887171506882
##  [78]   10137.46922402964446519035845994949
##  [79]   10162.44063022016416653059422969818
##  [80]   10575.49890782208785822149366140366
##  [81]   11135.25552781922124268021434545517
##  [82]   11575.96888557304737332742661237717
##  [83]   12224.26107470222814299631863832474
##  [84]   12690.44037477945857972372323274612
##  [85]   14540.51399411636884906329214572906
##  [86]   14691.73745167259403388015925884247
##  [87]   14964.93013488775068253744393587112
##  [88]   19982.52479543017761898227035999298
##  [89]   22966.63776803953805938363075256348
##  [90]   23700.90177559171934262849390506744
##  [91]   23817.48782108449086081236600875854
##  [92]   25406.07304495342759764753282070160
##  [93]   25419.09719491797659429721534252167
##  [94]   29743.63409570794101455248892307281
##  [95]   30620.41594791543684550561010837555
##  [96]   32171.80378124470007605850696563721
##  [97]   33823.14795800577121553942561149597
##  [98]   37540.64640890333976130932569503784
##  [99]   42305.47656812327477382495999336243
## [100]   54268.57388023888051975518465042114
## [101]   82887.84110241988673806190490722656
## [102]   89632.03081243125780019909143447876
## [103]  268580.85657596361124888062477111816
## [104]  659717.39749463775660842657089233398
```

```r
# pacman::p_load(ggplot2, mlr3,mlr)
# library(mlr3)
# library(mlr)
# modeling_task = makeRegrTask(data=data.frame(X_train), target='sale_price')
# algorithm = makeLearner("regr.lm")
# validation = makeResampleDesc("CV", iters = 5)
# #Having issues with fuel_type none
# res = resample(algorithm, modeling_task, validation,measures = list(rmse))
# res
```

```
# #average rsme somehow worse than above
# mean(res$measures.test$rmse)
```

REGRESSION TREEES.

Here the trees overfit in sample but they did pretty decent out of sample but not better than OLS

```
options(java.parameters = "-Xmx4000m")

X_train_CART = X_train
X_train_CART$sale_price = NULL

X_test_CART = X_test
X_test_CART$sale_price = NULL
tree_model = YARFCART(X_train_CART, y_train, bootstrap_indices = 1 : n_train, calculate_oob_error = TRUE
```

```
## YARF initializing with a fixed 1 trees...
## YARF factors created...
## YARF after data preprocessed... 31 total features...
## Beginning YARF regression model construction...done.
## Calculating OOB error...done.
```

```
illustrate_trees(tree_model, max_depth = 4, open_file = TRUE,margin_in_px=200,font_size=20,length_in_px
get_tree_num_nodes_leaves_max_depths(tree_model)
```

```
## $num_nodes
## [1] 375
##
## $num_leaves
## [1] 188
##
## $max_depths
## [1] 25
```

```
#In Sample Error
y_hat_train = predict(tree_model,X_train)
```

```
## Warning in predict.YARF(tree_model, X_train): Prediction set column names did not match training set
## Attempting to subset to training set columns.
```

```
e_train = y_train - y_hat_train
rsme_train = sd(e_train)
rsquared_train = (var(y_train) - var(e_train)) / var(y_train)

#Out of Sample Error
y_hat_test = predict(tree_model, X_test)
```

```
## Warning in predict.YARF(tree_model, X_test): Prediction set column names did not match training set c
## Attempting to subset to training set columns.
```

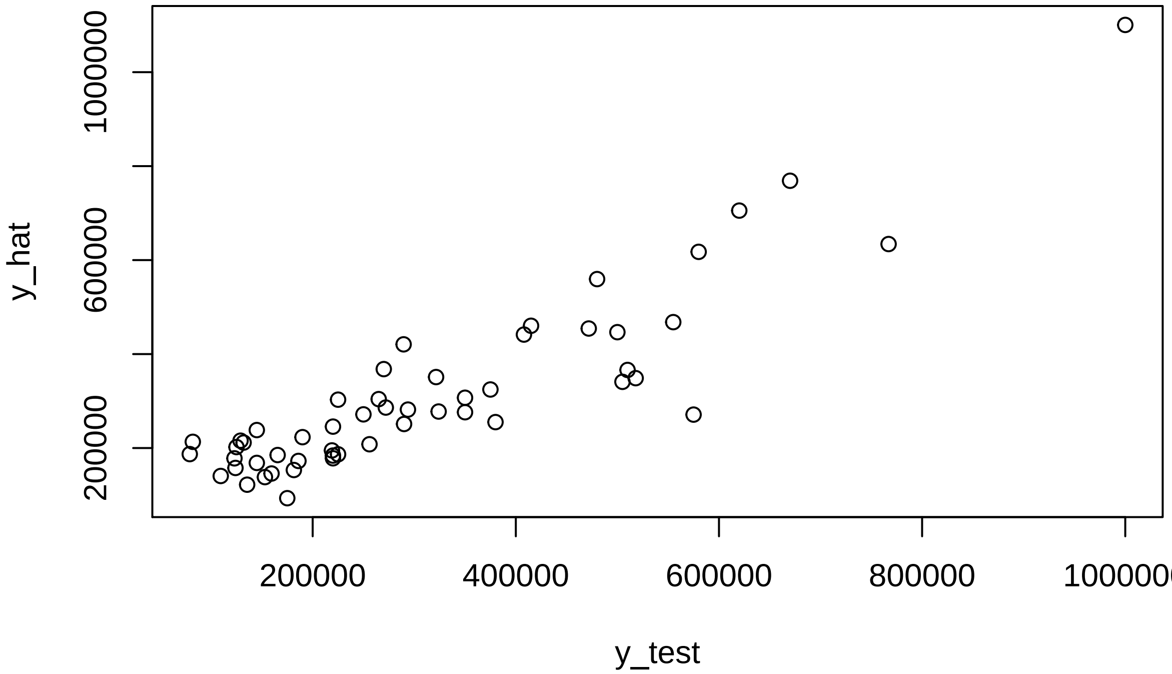
```
e_test = y_test - y_hat_test
rsme_test = sd(e_test)
rsquared_test = (var(y_test) - var(e_test)) / var(y_test)

cat("My R Squared in sample is ",rsquared_train, "My RSME is:", rsme_train)
```

```
## My R Squared in sample is  0.9882321 My RSME is: 19320.98
```

```r
cat("\nMy R Squared out of sample is ",rsquared_test, "My RSME is:",rsme_test)
```

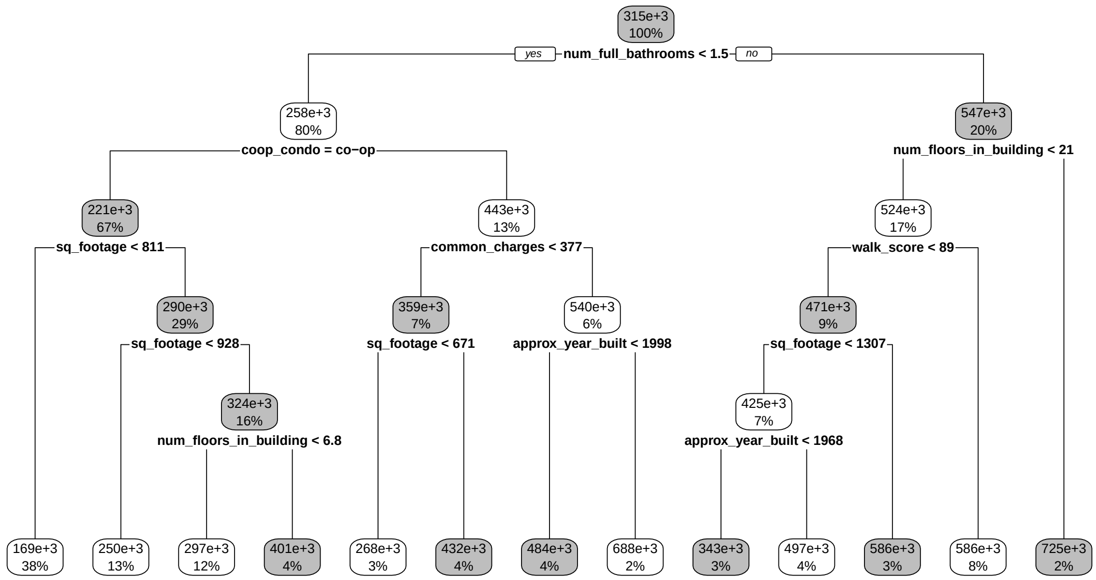
```
##
## My R Squared out of sample is  0.7961257 My RSME is: 87413.18
```

```r
plot(y_test,y_hat)
```




```r
library(rpart,mlr)
library("rpart.plot")
rpart_fit = rpart(sale_price ~., data.frame(X_train),method="anova")
rpart.plot(rpart_fit,box.col=c("grey", "white"))
```

```
## Warning: Bad 'data' field in model 'call' (expected a data.frame or a matrix).
## To silence this warning:
##     Call rpart.plot with roundint=FALSE,
##     or rebuild the rpart model with model=TRUE.
```




```r
rpart_fit
```

```
## n= 475
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 475 15036150000000 314877.8
##    2) num_full_bathrooms< 1.5 381  6589647000000 257531.4
##      4) coop_condo=co-op 318  2568349000000 220853.0
##        8) sq_footage< 811.1617 181   389403400000 168632.4 *
##        9) sq_footage>=811.1617 137  1033253000000 289845.1
##         18) sq_footage< 928.3983 63   336600400000 249738.1 *
##         19) sq_footage>=928.3983 74   509036800000 323990.2
##           38) num_floors_in_building< 6.765 55   307402000000 297298.0 *
##           39) num_floors_in_building>=6.765 19    49014940000 401257.3 *
##      5) coop_condo=condo 63  1434082000000 442670.1
##       10) common_charges< 377 34   494643300000 359439.4
##         20) sq_footage< 671.3365 15    92026640000 267975.9 *
##         21) sq_footage>=671.3365 19   178067100000 431647.4 *
##       11) common_charges>=377 29   427770100000 540250.9
##         22) approx_year_built< 1997.5 21   111454700000 483780.4 *
##         23) approx_year_built>=1997.5 8    73558990000 688486.0 *
##    3) num_full_bathrooms>=1.5 94  2115051000000 547313.8
##      6) num_floors_in_building< 20.5 83  1618691000000 523704.8
##       12) walk_score< 88.5 45  1045441000000 471400.0
##         24) sq_footage< 1307.457 32   432133500000 425031.2
##           48) approx_year_built< 1967.5 15    98365230000 343366.7 *
##           49) approx_year_built>=1967.5 17   145464100000 497088.2 *
##         25) sq_footage>=1307.457 13   375147200000 585538.5 *
##       13) walk_score>=88.5 38   304349500000 585644.7 *
##      7) num_floors_in_building>=20.5 11   101022700000 725454.5 *
```

```r
library(mlr,mlr3)
```

```
## Loading required package: ParamHelpers
```

```
## 'mlr' is in maintenance mode since July 2019. Future development
## efforts will go into its successor 'mlr3' (<https://mlr3.mlr-org.com>).
##
## Attaching package: 'mlr'
## The following object is masked from 'package:caret':
##
##     train
```

```r
modeling_task = makeRegrTask(data=data.frame(X_train), target='sale_price')
```

```
## Warning in makeTask(type = type, data = data, weights = weights, blocking =
## blocking, : Empty factor levels were dropped for columns: dining_room_type
```

```r
algorithm = makeLearner("regr.rpart")
validation = makeResampleDesc("CV", iters = 5)
#Having issues with fuel_type none
res = resample(algorithm, modeling_task, validation,measures = list(rmse))
```

```
## Resampling: cross-validation
## Measures:             rmse
## [Resample] iter 1:    77933.2398912
## [Resample] iter 2:    98589.0648772
## [Resample] iter 3:    99565.4579709
## [Resample] iter 4:    113289.5639943
## [Resample] iter 5:    106979.1754515
##
## Aggregated Result: rmse.test.rmse=99985.7422881
##
```

```r
res
```

```
## Resample Result
## Task: data.frame(X_train)
## Learner: regr.rpart
## Aggr perf: rmse.test.rmse=99985.7422881
## Runtime: 0.0736251
```

```r
#average rsme somehow worse than above
mean(res$measures.test$rmse)
```

```
## [1] 99271.3
```

```r
X_train_RF = X_train
X_train_RF$sale_price = NULL

X_test_RF = X_test
X_test_RF$sale_price = NULL
Bag_model = YARFBAG(X_train_RF, y_train, num_trees = 250, seed = 1 ,calculate_oob_error = TRUE)
```

```
## YARF initializing with a fixed 250 trees...
## YARF factors created...
## YARF after data preprocessed... 31 total features...
## Beginning YARF regression model construction...done.
```

```
## Calculating OOB error...done.
```

```r
Bag_model$rmse_oob
```

```
## [1] 77425.65
```

```r
y_hat_test_bag = predict(Bag_model, X_test)
```

```
## Warning in predict.YARF(Bag_model, X_test): Prediction set column names did not match training set c
## Attempting to subset to training set columns.
```

```r
s_e_bag = sd(y_test - y_hat_test_bag)
s_e_bag
```

```
## [1] 76296.75
```

```r
#In Sample Error
y_hat_train = predict(Bag_model,X_train)
```

```
## Warning in predict.YARF(Bag_model, X_train): Prediction set column names did not match training set
## Attempting to subset to training set columns.
```

```r
e_train = y_train - y_hat_train
rsme_train = sd(e_train)
rsquared_train = (var(y_train) - var(e_train)) / var(y_train)

#Out of Sample Error
y_hat_test = predict(Bag_model, X_test)
```

```
## Warning in predict.YARF(Bag_model, X_test): Prediction set column names did not match training set c
## Attempting to subset to training set columns.
```

```r
e_test = y_test - y_hat_test
rsme_test = sd(e_test)
rsquared_test = (var(y_test) - var(e_test)) / var(y_test)

cat("My R Squared in sample is ",rsquared_train, "My RSME is:", rsme_train,"\n" )
```

```
## My R Squared in sample is  0.9702822 My RSME is: 30703.46
```

```r
cat("\nMy R Squared out of sample is ",rsquared_test, "My RSME is:",rsme_test,"\n" )
```

```
##
## My R Squared out of sample is  0.8446824 My RSME is: 76296.75
```

```r
df_final_bag_all = copy(df_final)
y_all = df_final_bag_all$sale_price
df_final_bag_all$sale_price = NULL
mod_bag_all = YARFBAG(df_final_bag_all, y_all, num_trees = 250, seed = 28)
```

```
## YARF initializing with a fixed 250 trees...
## YARF factors created...
## YARF after data preprocessed... 30 total features...
## Beginning YARF regression model construction...done.
## Calculating OOB error...done.
```

```r
mod_bag_all$rmse_oob
```

```
## [1] 76421.52
```

RANDOM FOREST

```
set.seed(2)
X_train_RF = X_train
X_train_RF$sale_price = NULL


X_test_RF = X_test
X_test_RF$sale_price = NULL


copy_df = copy(df_final)
total_dfX = copy_df
total_dfY = copy_df$sale_price
total_dfX$sale_price = NULL
RF_other = YARF(total_dfX, total_dfY, num_trees = 250, seed = 1 ,calculate_oob_error = TRUE)
```

```
## YARF initializing with a fixed 250 trees...
## YARF factors created...
## YARF after data preprocessed... 30 total features...
## Beginning YARF regression model construction...done.
## Calculating OOB error...done.
```

```
RF_model = YARF(X_train_RF, y_train, num_trees = 250, seed = 1 ,calculate_oob_error = TRUE)
```

```
## YARF initializing with a fixed 250 trees...
## YARF factors created...
## YARF after data preprocessed... 31 total features...
## Beginning YARF regression model construction...done.
## Calculating OOB error...done.
```

```
#In Sample Error
y_hat_train = predict(RF_model,X_train)
```

```
## Warning in predict.YARF(RF_model, X_train): Prediction set column names did not match training set co
## Attempting to subset to training set columns.
```

```
e_train = y_train - y_hat_train
rsme_train = sd(e_train)
rsquared_train = (var(y_train) - var(e_train)) / var(y_train)

#Out of Sample Error
y_hat_test = predict(RF_model, X_test)
```

```
## Warning in predict.YARF(RF_model, X_test): Prediction set column names did not match training set co
## Attempting to subset to training set columns.
```

```
e_test = y_test - y_hat_test
rsme_test = sd(e_test)
rsquared_test = (var(y_test) - var(e_test)) / var(y_test)

cat("My R Squared in sample is ",rsquared_train, "My RSME is:", rsme_train,"\n" )
```

```
## My R Squared in sample is  0.9657972 My RSME is: 32938.96
```

```
cat("\nMy R Squared out of sample is ",rsquared_test, "My RSME is:",rsme_test,"\n" )
```

```
##
```

```
## My R Squared out of sample is  0.820339 My RSME is: 82058.32
```

```r
cat("OOB RSME:",RF_model$rmse_oob,"\n" )
```

```
## OOB RSME: 78439.13
```

```r
cat("GAIN OVER TREES",(mod_bag_all$rmse_oob - RF_model$rmse_oob) / mod_bag_all$rmse_oob * 100, "%\n" )
```

```
## GAIN OVER TREES -2.640113 %
```

```r
RF_model$rmse_oob
```

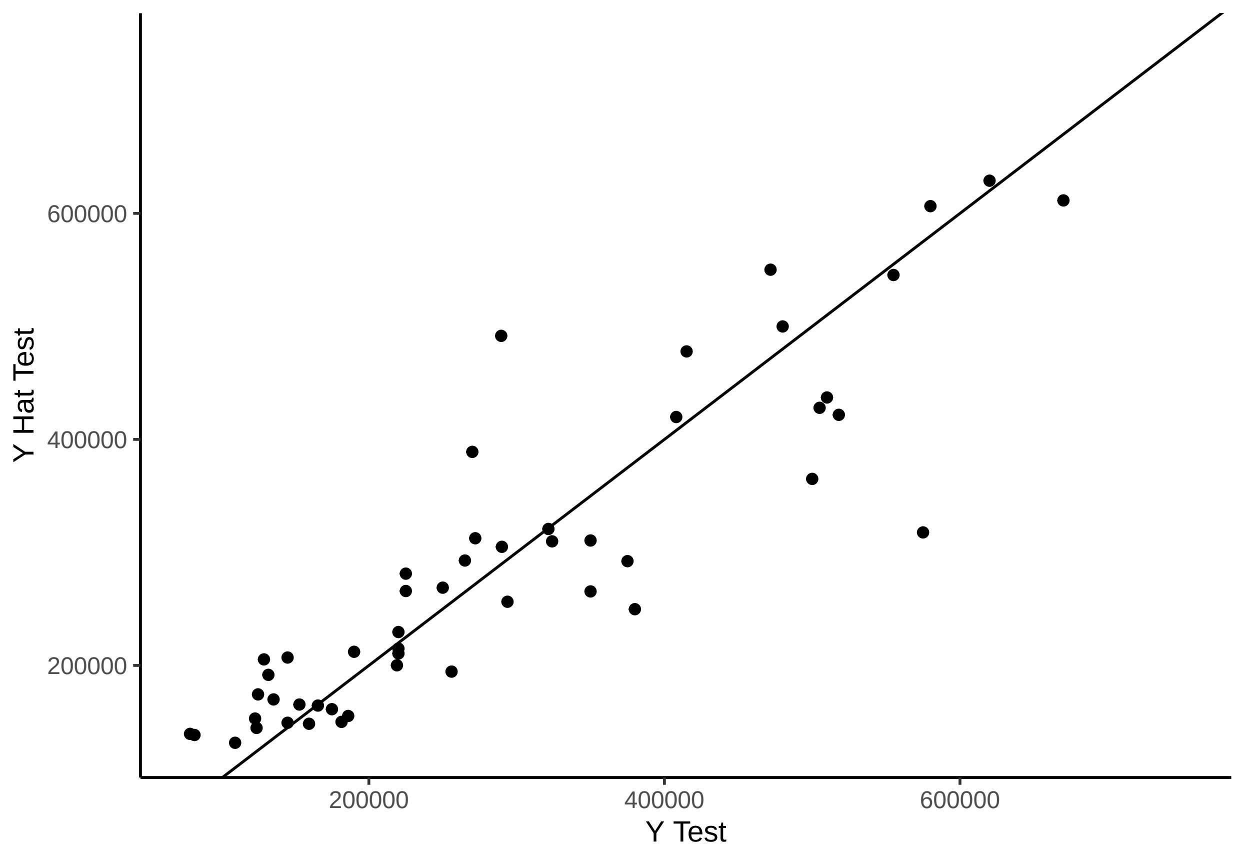
```
## [1] 78439.13
```

```r
RF_model
```

```
## YARF v1.0 for regression
## Missing data feature ON.
## 250 trees, training data n = 475 and p = 31
## Model construction completed within 0.05 minutes.
## OOB results on all observations:
##   R^2: 0.80563
##   RMSE: 78439.13
##   MAE: 53379.91
##   L2: 2922531075753
##   L1: 25355456
```

```r
ggplot( data.frame(y_test,y_hat_test),aes(x=y_test, y=y_hat_test)) + geom_point() + theme_classic()   +
  labs(x="Y Test", y = "Y Hat Test") + geom_abline() +xlim(NA,750000)
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



```r
set.seed(28)
modeling_task = makeRegrTask(data=data.frame(X_train), target='sale_price')
```

```
## Warning in makeTask(type = type, data = data, weights = weights, blocking =
## blocking, : Empty factor levels were dropped for columns: dining_room_type
```

```r
algorithm = makeLearner("regr.randomForest")
validation = makeResampleDesc("CV", iters = 5)
#Having issues with fuel_type none
res = resample(algorithm, modeling_task, validation,measures = list(rmse))
```

```
## Resampling: cross-validation

## Measures:             rmse

## [Resample] iter 1:    87545.1912787

## [Resample] iter 2:    75434.2169342

## [Resample] iter 3:    73396.4503821

## [Resample] iter 4:    82743.6328139

## [Resample] iter 5:    55317.9247182

##

## Aggregated Result: rmse.test.rmse=75694.2561831

##
```

```r
res
```

```
## Resample Result
## Task: data.frame(X_train)
## Learner: regr.randomForest
## Aggr perf: rmse.test.rmse=75694.2561831
## Runtime: 2.56153
```

```r
#average rsme somehow worse than above
mean(res$measures.test$rmse)
```

```
## [1] 74887.48
```

```r
set.seed(28)
modeling_task = makeRegrTask(data=data.frame(X_train), target='sale_price')
```

```
## Warning in makeTask(type = type, data = data, weights = weights, blocking =
## blocking, : Empty factor levels were dropped for columns: dining_room_type
```

```r
algorithm = makeLearner("regr.randomForest")
holdout = makeResampleDesc("Holdout")
#Having issues with fuel_type none
res = resample(algorithm, modeling_task, holdout,measures = list(rmse))
```

```
## Resampling: holdout

## Measures:             rmse

## [Resample] iter 1:    85520.2547615

##

## Aggregated Result: rmse.test.rmse=85520.2547615
```

```
##
```

```r
res
```

```
## Resample Result
## Task: data.frame(X_train)
## Learner: regr.randomForest
## Aggr perf: rmse.test.rmse=85520.2547615
## Runtime: 0.377138
```

```r
#average rsme somehow worse than above
mean(res$measures.test$rmse)
```

```
## [1] 85520.25
```

```r
# library(rpart)
# library(rpart.plot)
# fit = rpart(sale_price ~., data.frame(X_train),method="anova")
# rpart.plot(fit)
# summary(fit)
# pred
#
# in_e = y_train - pred
# sd(in_e)
# (var(y_train) - var(e)) / var(y_train)
# e = y_test - pred
# sd(e)
#
# Rsq_oos = (var(y_test) - var(e)) / var(y_test)
# #sd(e)
# cat("My R Squared in sample is ",summary(mod)$r.squared, "My RSME is:", sd(in_e))
# cat("\nMy R Squared out of sample is ",Rsq_oos, "My RSME is:", sd(e))
# ```
# ```{r}
# library(randomForest)
# control <- trainControl(method="cv", number=10)
#
#
# RegressionTree1 = train(sale_price~., data=data.frame(X_train), method="rpart", trControl=control)
# y_hat = predict(object = RegressionTree1,newdata = data.frame(X_test))
# sqrt(mean((y_hat-y_test)^2))
#
# RegressionTree = train(sale_price~., data=df_final, method="rpart", trControl=control)
# print(RegressionTree)
#
#
# ##
#
# fit = rpart(sale_price ~., data.frame(X_train),method = 'anova')
# printcp(fit)
# rpart.plot(fit)
# summary(fit)
# y_hat = predict(object = fit,newdata = data.frame(X_test))
# sqrt(mean((y_hat-y_test)^2))
```

```r
# RandomForest = train(sale_price~., data=df_final, method="rf", trControl=control)
# print(RandomForest)
```

```
#
#
```