Final Project Draft

Peter Antonaros

Packages & Setup

```
#Packages to load
pacman::p_load(
  ggplot2,
  tidyverse,
  data.table,
  R.utils,
  magrittr,
  dplyr,
  testthat,
  YARF,
  lubridate
)

#Memory allocation for Java ~6gb
options(java.parameters = "-Xmx 6000m")
library(rJava)
.jinit()
```

The Data

```
housingDataFilePath = "/home/peterjr/RepoCollections/MATH_342W_FinalProject/Datasets/housing_data_2016_
housingData = data.table(read.csv(housingDataFilePath))

housingData

#Relevant columns begin at the column labeled (URL)
```

Initial Data Preparation I (Dropping Irrelevant Columns & Storing Possible Ones for Later Use)

```
#Dropping Mturk columns that are not relevant to our housing model
housingData[,c(1:27):=NULL]

#Save the urls for later and remove from data frame (might be useful but not immediately)
housingURLS = housingData[,.(URL)]

#Dropping URL from the data table
housingData[,URL:=NULL]
#Dropping other useless url column from data table (ALL NA's)
housingData[,url:=NULL]
#Dropping model_type because similar information is contained in other columns
housingData[,model_type:=NULL]
```

```
housingData
```

Initial Data Preparation II (Writing some notes about Columns)

```
#Getting the column names to write some notes about each column
names(housingData)

#Getting some general information about the table
summary(housingData)
```

Column Name | Information | Notes to Self about column

"approx_year_built" | Integer representing the year the house was built | 40 NA's

"cats_allowed" | Binary decision (0,1) are cats allowed in the home or not | Check for NA's & Factor

"common_charges" | Some sort of charges in dollars ($) | Remove the dollar symbol & Convert to integer & Check for NA's

"community_district_num" | Integer representing the district number of community home is a part of | 19 NA's

"coop_condo" | String representing "Co-op" or "Condo" | Lowercase everything | Check for levels & Factor

"date_of_sale" | String representing the date the home was sold |

"dining_room_type" | String representing "formal" or "combo" dining room type | Lowercase everything & Check for NA's & Factor

"dogs_allowed" | Binary decision (0,1) are dogs allowed in the home or not | Factor this & Check for NA's

"fuel_type" | String representing "gas", "oil", or "other" energy source for the home | Lowercase everything & Check for NA's & factor

"full_address_or_zip_code" | String representing the address of the home |

"garage_exists" | String representing "Yes" if the home has a garage | Check for NA's & Factor this & Missingness column

"kitchen_type" | String representing "Eat-In", "Efficiency", or "Combo" kitchen type | Lowercase everything & Factor this & Check for NA's

"maintenance_cost" | Cost of maintenece for the home in dollars ($) | Remove the dollar symbol & Convert to integer & Check for NA's

"num_bedrooms" | Integer representing number of bedrooms present in the home | 115 NA's

"num_floors_in_building" | Integer representing number of floors present in building containing home | 650 NA's

"num_full_bathrooms" | Integer representing the number of full bathrooms present in the home | No NA's

"num_half_bathrooms" | Integer representing the number of half bathrooms present in the home | 2058 NA's

"num_total_rooms" | Integer representing the number of total rooms present in the home | 2 NA's

"parking_charges" | Parking charges in dollars ($) | Remove the dollar symbol & Convert to integer & Check for NA's

"pct_tax_deductibl" | Integer representing percent of tax deduction | 1754 NA's

"sale_price" | Sale price of the home in dollars ($) | Remove the dollar symbol & Convert to integer & Check for NA's

"sq_footage" | Integer representing the total square footage of the home | 1210 NA's

"total_taxes" | Taxes on the home in dollars ($) | Remove the dollar symbol & Convert to integer & Check for NA's

"walk_score" | Integer representing a walking score for the home |

"listing_price_to_nearest_1000" | Listing price to the nearest 1000 for the home in dollars ($) | Remove the dollar symbol & Convert to integer & Check for NA's

Data Cleaning I (Fixing column types)

```
#First lets deal with the String columns that have $ symbols and convert to integer

#Extract dollar sign columns as subset to operate on
dollarSymbolSubset = housingData[,.(common_charges,maintenance_cost,parking_charges,sale_price,total_ta
#Remove dollar signs based on pattern matching
dollarSymbolSubset[] = lapply(dollarSymbolSubset,gsub,pattern="$",fixed=TRUE,replacement="")


#Replace the columns in housing Data with the new dollarSymbolSubset
housingData[,c("common_charges","maintenance_cost","parking_charges","sale_price","total_taxes","listing
            dollarSymbolSubset[,c("common_charges","maintenance_cost","parking_charges","sale_price","t


#Now we need to convert these columns in housing data to integer type
housingData[,c("common_charges","maintenance_cost","parking_charges","sale_price","total_taxes","listing
```

```
## Warning in lapply(housingData[, c("common_charges", "maintenance_cost", : NAs
## introduced by coercion

## Warning in lapply(housingData[, c("common_charges", "maintenance_cost", : NAs
## introduced by coercion

## Warning in lapply(housingData[, c("common_charges", "maintenance_cost", : NAs
## introduced by coercion

## Warning in lapply(housingData[, c("common_charges", "maintenance_cost", : NAs
## introduced by coercion

## Warning in lapply(housingData[, c("common_charges", "maintenance_cost", : NAs
## introduced by coercion
```

```
##############################################################################################
#Second lets deal with changing cats_allowed and dogs_allowed to factors

housingData[,sum(is.na(cats_allowed))] # No NA values for cats_allowed
housingData[,sum(is.na(dogs_allowed))] # No NA values for dogs_allowed
```

```r
#Changing to factors for cats and dogs allowed
unique(housingData[,cats_allowed]) # 3 "unique" values

#Lets deal with the y instead of a yes
housingData$cats_allowed[grepl("y", housingData$cats_allowed)] = "yes"
length(unique(housingData[,cats_allowed])) # 2 unique values

#Lets do the same for dogs
unique(housingData[,dogs_allowed]) # 3 "unique" values"
housingData$dogs_allowed[grepl("yes89", housingData$dogs_allowed)] = "yes"
length(unique(housingData[,cats_allowed])) # 2 unique values

#Factor them
housingData[,c("cats_allowed","dogs_allowed")] = lapply(housingData[,c("cats_allowed","dogs_allowed")],

levels(housingData$cats_allowed) #Check levels
levels(housingData$dogs_allowed) #Check levels

##############################################################################################
#Third lets deal with the other String columns that need to be factored (track NA's for later)

housingData[,sum(is.na(coop_condo))] # No NA values for coop_condo
length(unique(housingData[,coop_condo])) # 2 unique values

#Factor it
housingData[,coop_condo := factor(coop_condo)]
levels(housingData$coop_condo)


housingData[,sum(is.na(dining_room_type))] # 448 NA values for dining_room_type
length(unique(housingData[,dining_room_type])) # 6 unique values including NA
length(which(housingData$dining_room_type == "none")) #none occurs 2 times
length(which(housingData$dining_room_type == "dining area")) #dining area occurs 2 times

#Lets deal with the issue of "dining area" as the room type and consider it as type other
housingData$dining_room_type[grepl("dining area", housingData$dining_room_type)] = "other"
length(unique(housingData[,dining_room_type])) # 5 unique values including NA

housingData[,dining_room_type := factor(dining_room_type)]
levels(housingData$dining_room_type)


housingData[,sum(is.na(fuel_type))] # 112 NA values for dining_room_type
length(unique(housingData[,fuel_type])) # 7 "unique" values including NA

#Lets deal with the capitalization issues for fuel_typenone
housingData[,fuel_type := tolower(fuel_type)]
length(unique(housingData[,fuel_type])) # 6 unique values including NA
housingData[,fuel_type := factor(fuel_type)]
levels(housingData$fuel_type)


housingData[,sum(is.na(kitchen_type))]# 16 NA values for dining_room_type
```

```
length(unique(housingData[,kitchen_type])) # 14 "unique" values including NA

#Lets deal with the upper case lower case kitchen type differences
housingData[,kitchen_type:=tolower(kitchen_type)] # Lowercase everything to pattern match
length(unique(housingData[,kitchen_type])) # 11 "unique" values including NA

#Lets now deal with spaces creating more unique values
housingData[,kitchen_type := lapply(kitchen_type,gsub,pattern="eat in",fixed=TRUE,replacement="eatin")]
length(unique(housingData[,kitchen_type])) # 10 "unique" values including NA

#Lets lets deal with the misspellings of efficiency kitchen
housingData$kitchen_type[grepl("effic", housingData$kitchen_type)] = "efficiency"
length(unique(housingData[,kitchen_type])) # 6 unique values including NA

#Finally lets deal with 1955 and replace that with NA -> I am assuming here 1955 is wrong and not a typ
housingData[, kitchen_type := sapply(kitchen_type, function(x) replace(x, which(x=="1955"), NA))]
length(unique(housingData[,kitchen_type])) # t unique values including NA (no 1955 -> NA)
housingData[,kitchen_type := factor(kitchen_type)]
levels(housingData$kitchen_type)

#############################################################################################
#Fourth lets deal with the Garage column (track NA's for later)

housingData[,sum(is.na(garage_exists))] # 1826 NA values for garage exists
length(unique(housingData[,garage_exists])) # 7 "unique" values

#Lets deal with the capitalization and misspelling of yes
housingData[,garage_exists := tolower(garage_exists)]
housingData$garage_exists[grepl("y", housingData$garage_exists)] = "yes"
length(unique(housingData[,garage_exists])) # 5 unique values including NA

#Lets treat underground and ug as yes
housingData$garage_exists[grepl("u", housingData$garage_exists)] = "yes"
length(unique(housingData[,garage_exists])) # 3 unique values including NA

#Lets treat 1 as a yes
housingData$garage_exists[grepl("1", housingData$garage_exists)] = "yes"
length(unique(housingData[,garage_exists])) # 2 unique values including NA

housingData[,garage_exists := factor(garage_exists)]
levels(housingData$garage_exists)

#############################################################################################
#Fifth lets take the date column and treat it as an ordinal factor
housingData[,date_of_sale:= factor(date_of_sale,ordered=TRUE)]
length(unique(housingData[,date_of_sale]))


#Lets take a look at our data set now

housingData
summary(housingData)
```

Data Manipulation I (Creating new columns)

```
#First lets just add up all the charges into a single column
chargeCols = c("common_charges","maintenance_cost","parking_charges","total_taxes")

#Assign new column totalCharges to be the row sum of the chargeCols ignoring NA's
housingData[, totalCharges := rowSums(.SD,na.rm=TRUE), .SDcols = c("common_charges","maintenance_cost",

housingData[,sum(is.na(totalCharges))] # No NA's here which is good since

################################################################################################
#Second lets extract the zip codes and assign them to their own column

#Lets use a regular expression to extract the zip code out of this field
housingData[,zip_code := substr(str_extract(full_address_or_zip_code,"[0-9]{5}"),1,6)]

#We can now drop the full_address column since we wont need that
housingData[,full_address_or_zip_code := NULL]




################################################################################################
#Lastly, lets just reorder the columns because I don't like how they are ordered now
#and its annoying to scroll back and forth for similar columns

reorderdCols = c(
  "zip_code","approx_year_built","community_district_num","coop_condo",
  "fuel_type","garage_exists",
  "cats_allowed","dogs_allowed","kitchen_type","dining_room_type",
  "common_charges","maintenance_cost","parking_charges","total_taxes",
  "pct_tax_deductibl","num_floors_in_building","num_bedrooms","num_half_bathrooms",
  "num_full_bathrooms","num_total_rooms",
  "sq_footage","walk_score","listing_price_to_nearest_1000","date_of_sale",
  "sale_price"
)
setcolorder(housingData, reorderdCols)



housingData
```

Imputation Via MissForest on the Data