Assignment: Week 4 - Housing

Name: Ramani, Aarti

Date: 2023-01-05

```
library(readxl)
setwd("C:/Masters/GitHub/Winter2022/Ramani-DSC520")
housing_df <- read_excel(path = "C:/Masters/GitHub/Winter2022/Ramani-DSC520/data/week-6-housing.xlsx",
                         .name_repair = function(col){ gsub(" ", "_", col) })
names(housing_df)
```

```
##  [1] "Sale_Date"                "Sale_Price"
##  [3] "sale_reason"              "sale_instrument"
##  [5] "sale_warning"             "sitetype"
##  [7] "addr_full"                "zip5"
##  [9] "ctyname"                  "postalctyn"
## [11] "lon"                      "lat"
## [13] "building_grade"           "square_feet_total_living"
## [15] "bedrooms"                 "bath_full_count"
## [17] "bath_half_count"          "bath_3qtr_count"
## [19] "year_built"               "year_renovated"
## [21] "current_zoning"           "sq_ft_lot"
## [23] "prop_type"                "present_use"
```

```
#survey_df <- read.csv(file="C:/Masters/GitHub/Winter2022/Ramani-DSC520/data/acs-14-1yr-s0201.csv")
#survey_df

#Use the apply function on a variable in your dataset
apply(housing_df,2,range)
```

```
##      Sale_Date    Sale_Price sale_reason sale_instrument sale_warning sitetype
## [1,] "2006-01-03" "    698"  " 0"        " 0"            NA           "A1"
## [2,] "2016-12-16" "4400000"  "19"        "27"            NA           "R4"
##      addr_full            zip5    ctyname postalctyn lon         lat
## [1,] "10002 242ND WAY NE" "98052" NA      "REDMOND"  "-121.9499" "47.45635"
## [2,] "9985 185TH CT NE"   "98074" NA      "REDMOND"  "-122.1643" "47.73255"
##      building_grade square_feet_total_living bedrooms bath_full_count
## [1,] " 2"           "  240"                  " 0"     " 0"
## [2,] "13"           "13540"                  "11"     "23"
##      bath_half_count bath_3qtr_count year_built year_renovated current_zoning
## [1,] "0"             "0"             "1900"     "   0"         "A10"
## [2,] "8"             "8"             "2016"     "2016"         "URPSO"
##      sq_ft_lot prop_type present_use
## [1,] "    785" "R"       "  0"
## [2,] "1631322" "R"       "300"
```

```
#Use the aggregate function on a variable in your dataset
aggregate(cbind(Sale_Price, bedrooms) ~ ctyname + zip5, housing_df, mean)
```

```
##     ctyname  zip5 Sale_Price bedrooms
## 1   REDMOND 98052   644803.2 3.683380
## 2 SAMMAMISH 98074   972480.3 4.090909
```

```
#PRACTICE
#list(housing_df$ctyname)
#aggregate(cbind(housing_df$Sale.Price, housing_df$bedrooms), list(housing_df$ctyname, housing_df$zip5)
#Validate mean of sale price from aggregrate function for the city REDMOND
#aggregate(Sale.Price ~ ctyname, housing_df, mean)
#mean(subset(housing_df, housing_df$ctyname=="REDMOND")$Sale.Price)
#mean(housing_df[housing_df$ctyname=="REDMOND",]$Sale.Price)
#aggregate(cbind(Sale.Price, bedrooms) ~ ctyname + zip5, upd_housing_df, mean)
#aggregate(cbind(Sale.Price, bedrooms) ~ ctyname + zip5, housing_df, mean)

#Use the plyr function on a variable in your dataset - more specifically, I want to see you split some
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#library(tidyr)
#upd2_houseing_df <- housing_df %>% separate(sale_warning , c("sale_warning_1", "sale_warning_2","sale_
#housing_df %>% filter(ctyname=='' & zip5==98052)
#upd_housing_df <- housing_df %>% mutate(ctyname = replace(ctyname, zip5==98052 & ctyname=='', "REDMOND
library(plyr)
```

```
## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
zip_df = subset(housing_df, zip5==98052)
nonzip_df = subset(housing_df, zip5!=98052)
#zip_df[zip_df$zip5==98052 & is.na(zip_df$ctyname),]
zip_df <- ddply(zip_df, .(zip5), mutate, ctyname = case_when(is.na(ctyname)&zip5==98052 ~ "REDMOND", TRU
housing_df <- full_join(zip_df, nonzip_df)
```

```
## Joining, by = c("Sale_Date", "Sale_Price", "sale_reason", "sale_instrument",
## "sale_warning", "sitetype", "addr_full", "zip5", "ctyname", "postalctyn",
## "lon", "lat", "building_grade", "square_feet_total_living", "bedrooms",
## "bath_full_count", "bath_half_count", "bath_3qtr_count", "year_built",
## "year_renovated", "current_zoning", "sq_ft_lot", "prop_type", "present_use")
```

```
#Check distributions of the data
library(fitdistrplus)
```

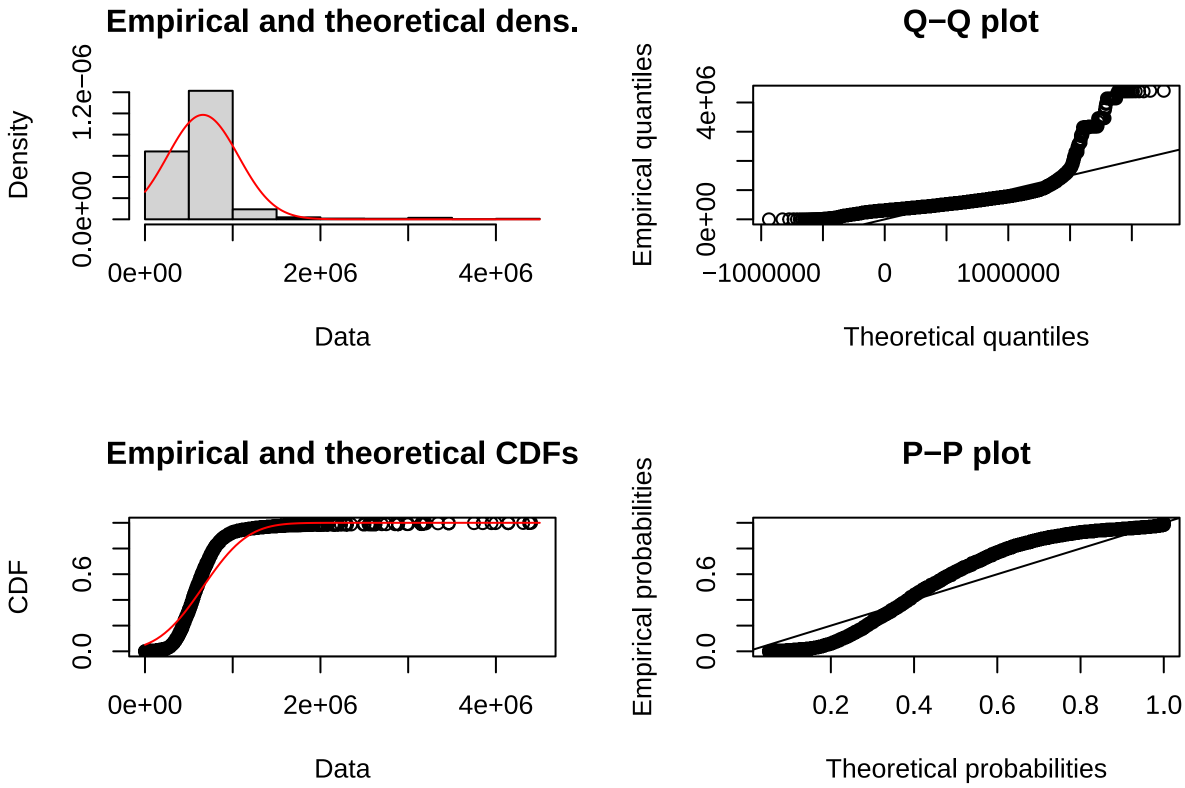
```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## Loading required package: survival
```

```
#descdist(housing_df$Sale_Price)
#plotdist(housing_df$Sale_Price, histo = TRUE, demp = TRUE,pch = 19)
plot(fitdist(housing_df$Sale_Price, "norm"))
```



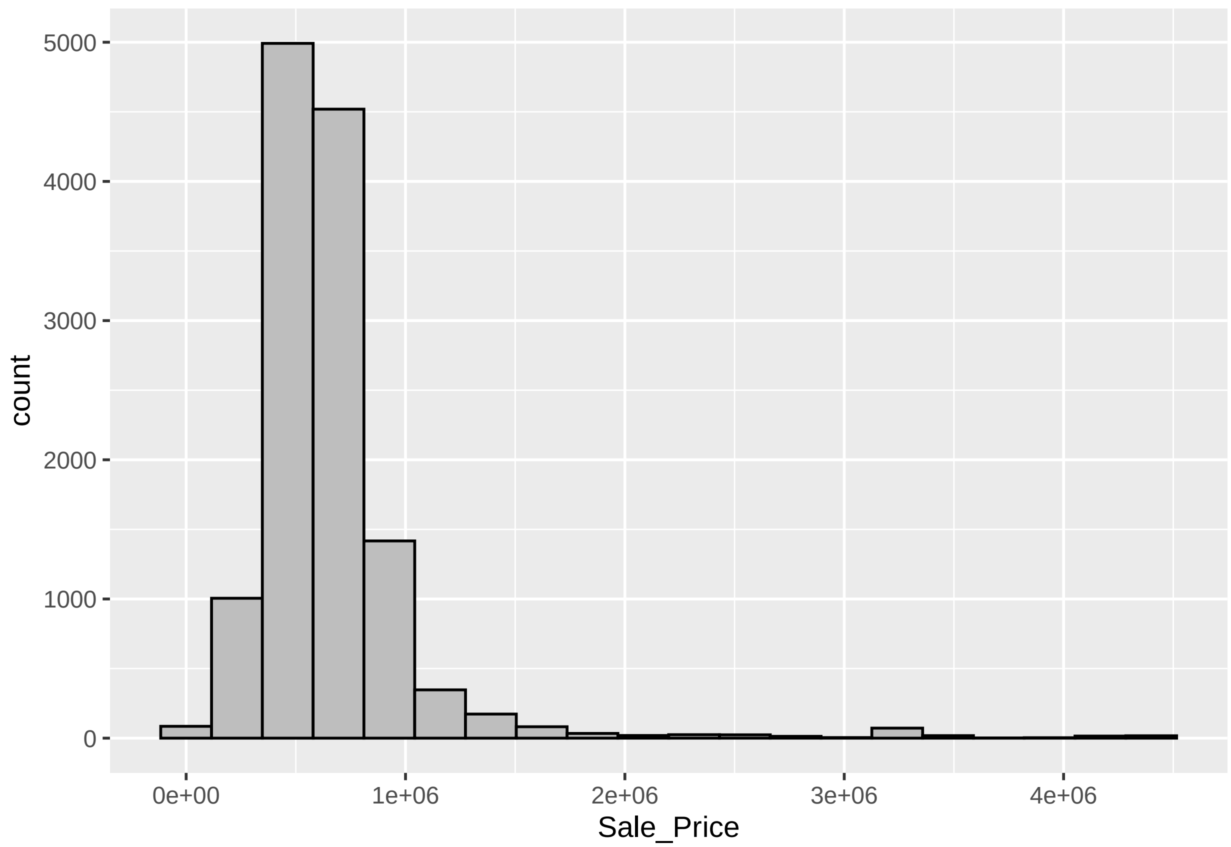
Empirical and theoretical dens.

Q−Q plot

Empirical and theoretical CDFs

P−P plot

```
#Identify if there are any outliers
library(ggplot2)
#ggplot(housing_df, aes(x=Sale_Price, y=prop_type))+ geom_point()
ggplot(housing_df, aes(x=Sale_Price))+ geom_histogram(fill="gray",bins=20, color="black")
```



```
#ggplot(housing_df, aes(x=Sale_Price))+ geom_boxplot()



#Create at least 2 new variables
State <- rep("California",12865)
Index <- c(1:12865)

housing <- data.frame(housing_df, State, Index)
colnames(housing)
```

```
##  [1] "Sale_Date"                "Sale_Price"
##  [3] "sale_reason"              "sale_instrument"
##  [5] "sale_warning"             "sitetype"
##  [7] "addr_full"                "zip5"
##  [9] "ctyname"                  "postalctyn"
## [11] "lon"                      "lat"
## [13] "building_grade"           "square_feet_total_living"
## [15] "bedrooms"                 "bath_full_count"
## [17] "bath_half_count"          "bath_3qtr_count"
## [19] "year_built"               "year_renovated"
## [21] "current_zoning"           "sq_ft_lot"
## [23] "prop_type"                "present_use"
## [25] "State"                    "Index"
```