EDA of Craigslist Housing Data – Part Two

1-2. Introduction

Craigslist is an online portal used for posting and finding local classified advertisements. This report cleans, organizes, and analyzes a messy dataset of Craigslist apartment postings in California. The aim of this report is to extract meaningful features from the initially unorganized data and analyze to draw conclusions about the apartments. The data was downloaded from the Craigslist website in a .txt file format. Data extraction and analysis is performed in R and Rstudio.

3. Discussion of *read_post* and *read_all_posts* user-defined function

My read_post function uses the readLines function from R to read a text file from a connection into a character vector. The function accepts one parameter 'directory' to specify the connection where the text file is located. I did not have to make any changes to read_post before writing read_all_posts. The function read_all_posts uses my first function read_post over multiple text files.

My read_all_posts function first takes the parameter 'directory' to get a list of file names in the directory. The recursive setting is set to TRUE so files from subdirectories will be included. Then, the lapply() function is used together with my function read_post (for reading one file) to read in all text files within the directory and its subdirectories and get a list as output. The str_split_fixed() function splits the output into three columns for title, text, and attributes. The data is then converted to a single, combined data frame where each row is an individual text posting, which makes it simple to compare the observations. The function also extracts the price from the title of each posting using the substr() function.

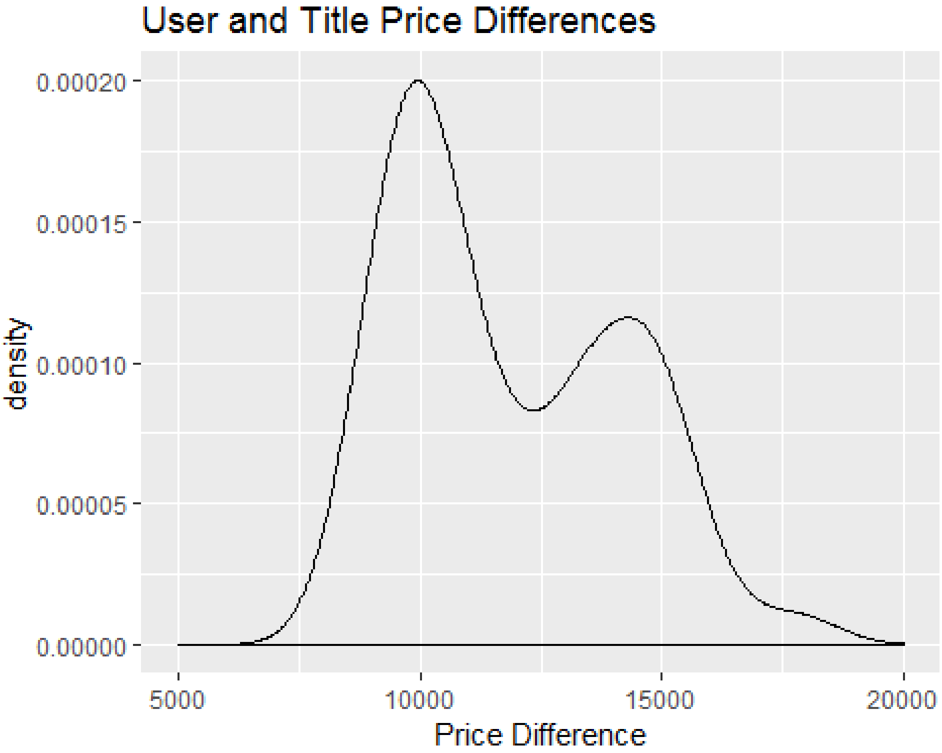
After reading in all the posts from the "messy" folder of Craigslist ads, additional string splitting operations were performed to further split the data into columns for price, latitude, longitude, bedrooms, bathrooms, and sqft in size. The column types are converted to numeric vectors for more convenient future analysis. This split data frame, called 'result', was combined with the original to create the final data frame. To answer questions 4-8, I extracted further details from the text body and added them as additional columns to my data frame. My choice of columns is suitable for comparing relationships between various features.

4. Do all of the titles have prices? How do these prices compare to the user-specified prices (the price attribute)?

Almost all titles have prices listed. There are 174 postings without title prices and 45,671 postings with title prices. Almost all the title listed prices are equal to the user-specified prices. There are 92 postings which don't have the title price equal to the price attribute included at the bottom of each posting. These are likely errors due to typos or non-apartment postings (such as ads for storage units).

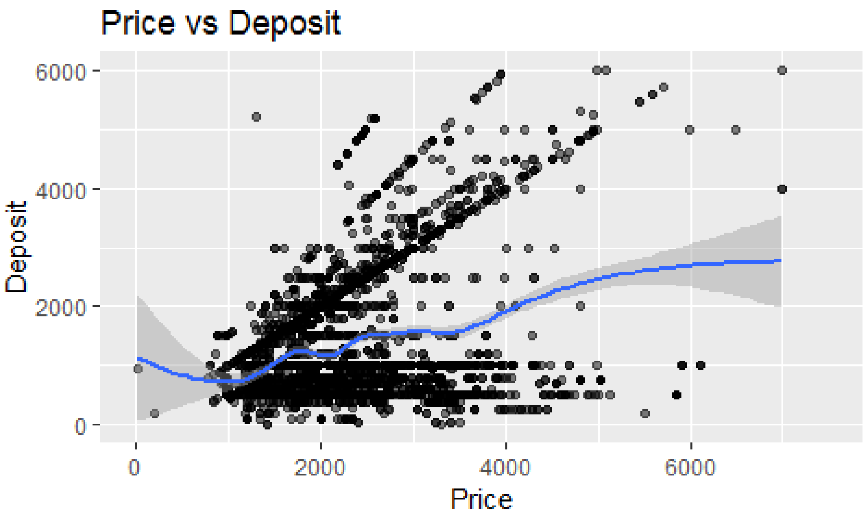
The density plot below shows the distribution in price differences between the two columns (user price and title price). Many differences in price are around $10,000, indicating the user price may have an extra '0' added in error. For instance, a $1,000 posting may have a $10,000 price in error, making the price difference 9,000. 3 outliers not shown in the density plot due to space limitation have a price difference greater than $20,000. By examining each of these posts individually, I found typos such as a range of prices entered as a single value. If one-bedroom and two-bedroom apartments had different prices, these may have been mistakenly combined into one price in the posting.

In summary, most if not all differences in user and title price are likely due to error in input or error in converting a range of prices to a single value. The title price is more likely to give an accurate price, because there are less missing values and less chance for input error.

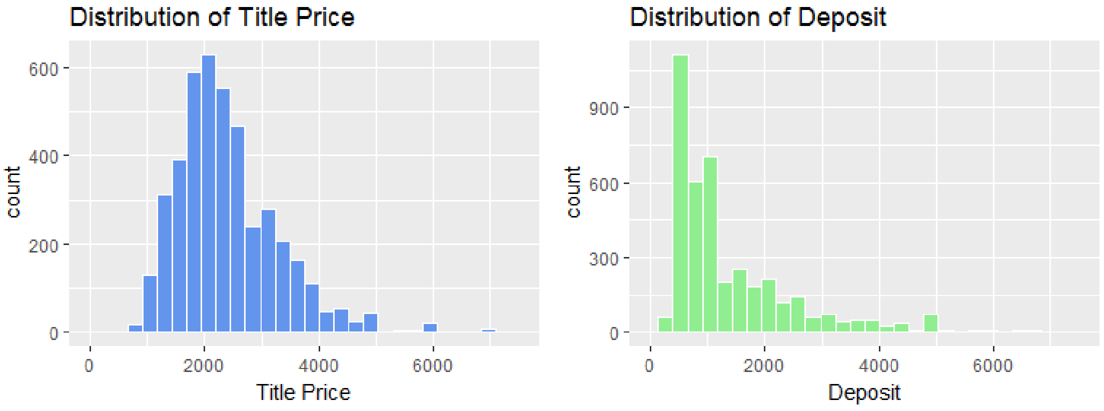


5. Is there a relationship between rental price and deposit amount?

To extract deposit amounts from the text, I used a regular expression that finds any character in between 'deposit:' and a space. A positive lookbehind `(?<=deposit: )` is used to detect the presence of 'deposit:' before my search query. A positive lookahead `(?= )` is used to detect a space after my search query. Finally, a regex `[^[:digit:]. ]` was used to extract only the digits and '.' from the string.

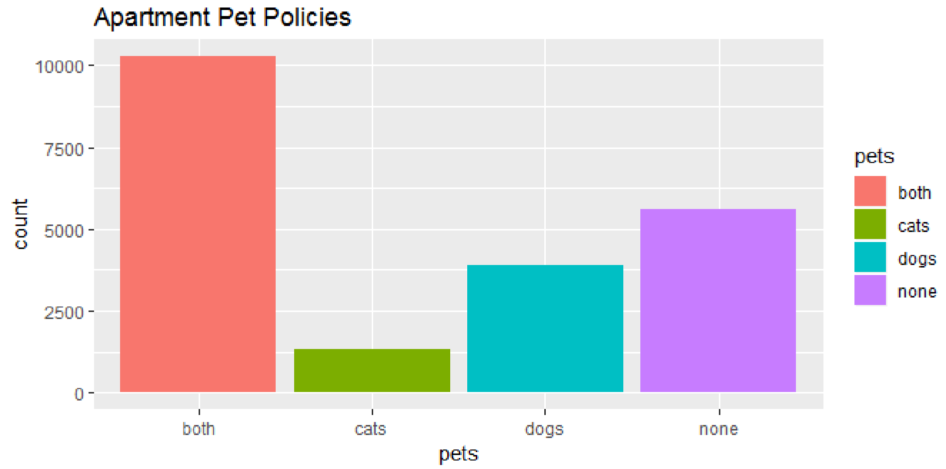


Yes, there is a definite positive relationship between rental price and deposit amount. The scatterplot of price versus deposits show a clear upward trend. As the price increases, the deposit is likely to increase slightly. The Pearson's correlation coefficient of 0.0248 supports this observed trend.



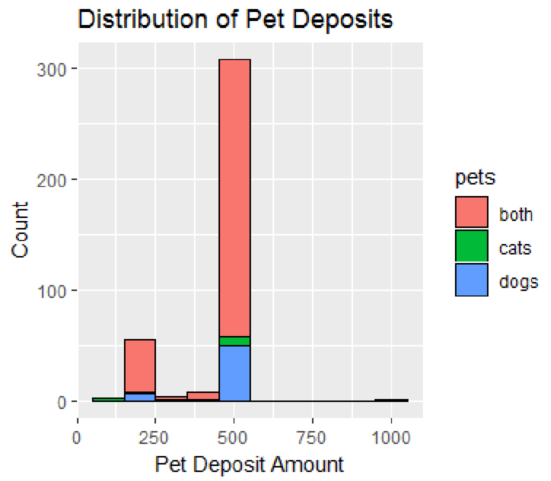
The distributions of title price and deposit show that title price has more variation. Deposits appear to be more right skewed, with the majority of values lower than $1,500. The data suggests that while price and deposit do have a positive correlation, deposit is still likely to be lower, and many postings set deposit as a fixed amount. For instance, almost 20% of postings (of the subset with information regarding deposits) have a $500 deposit.

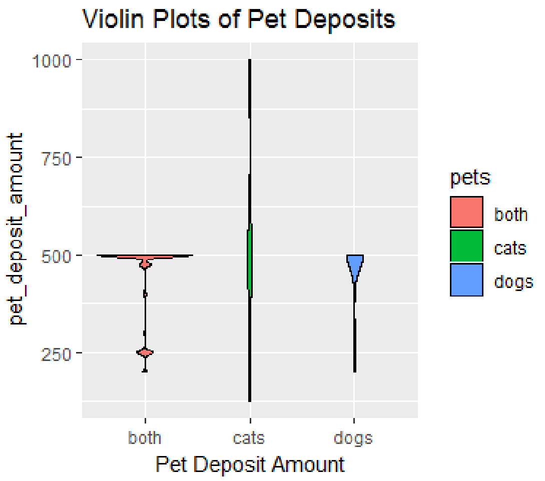
6. Pets



The bar chart above shows the distribution of the types of pet policies. “Both” is the most common policy, followed by ‘none’ and ‘dogs’. Many postings did not include pet information, which is represented by ‘NA’ in the pet column of the data frame.

There are apartments which allow birds and fish in addition to dogs and cats in their pet policy. By subsetting the data, I discover 299 postings that allow fish and 169 postings that allow birds.

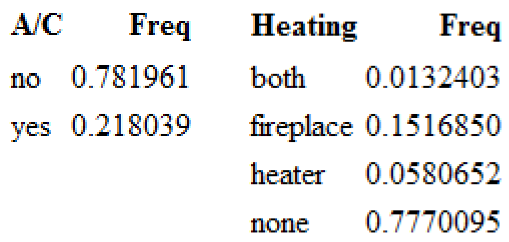




The graphic shows a distribution of pet deposits. Most deposit amounts are $500 and a small proportion around $250, with only a few outside of these values. Almost 75% of postings with pet deposits list the amount of $500. Apartments with “both” pet policy tend to have higher deposits. The graphic suggests most pet deposits are the same standard amount regardless of apartment price. The violin plots of pet distributions by pet type show that postings with “both” or “dogs” policies tend to have deposits close to $500. Postings with “cats” policies have more variation in pet deposit amount. Upon further examination, this is due to two high end outliers charging $1,000 cat deposits.

7. Heating and A/C

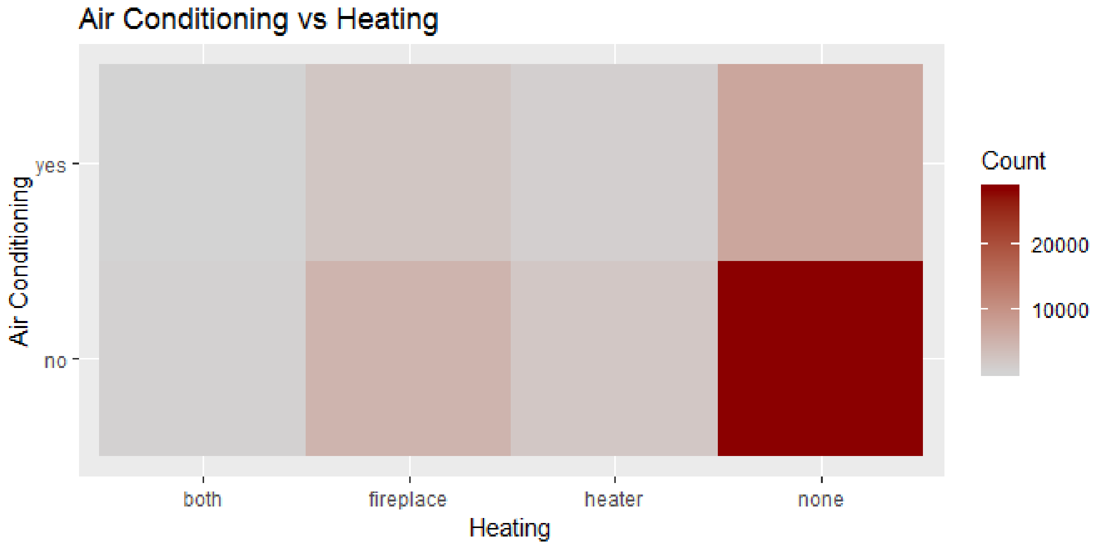
After extracting the heating and A/C features from the text, I calculated the proportion of apartments that have these features.



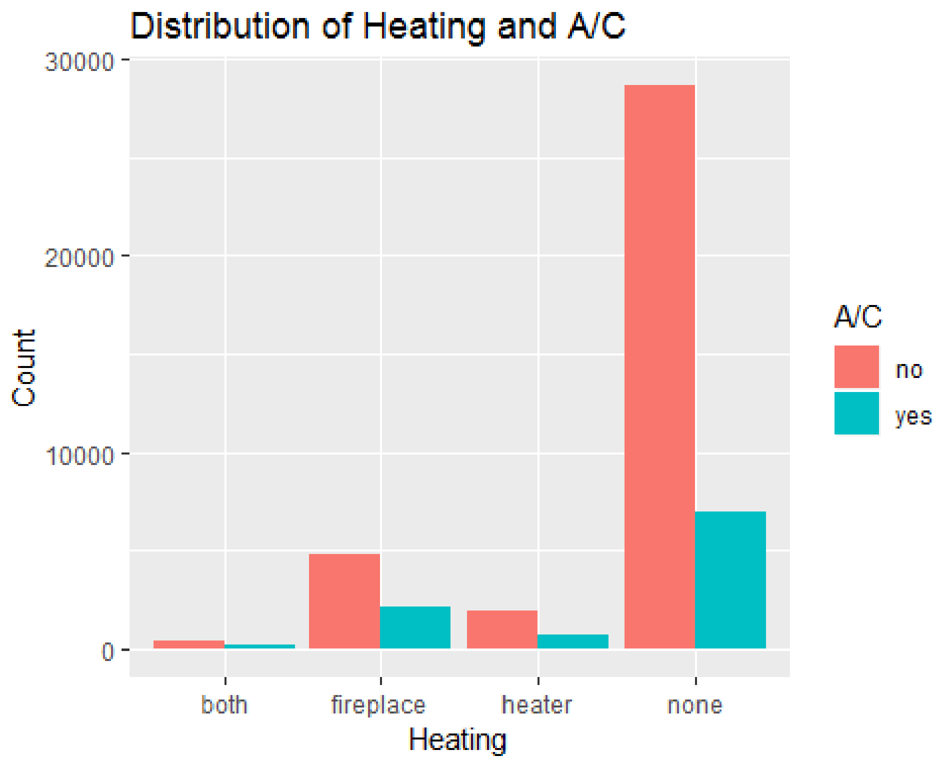
A/C	Freq
no	0.781961
yes	0.218039

Heating	Freq
both	0.0132403
fireplace	0.1516850
heater	0.0580652
none	0.7770095

The tables show a slightly higher proportion of apartments have some form of heating with either a heater, fireplace, or both. About 22.3% of all apartments have heating, compared to about 21.8% of apartments having air conditioning. Thus, heating is slightly more common than air conditioning.



A heatmap can show the covariate relationship between A/C and Heating. Since there are two categorical variables, the geom_tile function can visualize their relationship. The graphic shows there are few apartments with both A/C and heating. If an apartment has A/C, it is likely not to have heating as indicated by the darker color at the intersection of AC: "yes" and Heating: "none". If an apartment has heating, it is likely not to have A/C, as the AC:"no" tiles have darker colors at every column of Heating.



The bar chart shows most apartments have neither A/C nor heating. It is also likely many postings do not mention A/C or heating in the text but may implicitly provide these amenities without specification that is detectable by regular expression search. The higher “no A/C” orange bars corresponding to the “fireplace” and “heater” Heating categories show apartments with heating typically don’t have A/C.

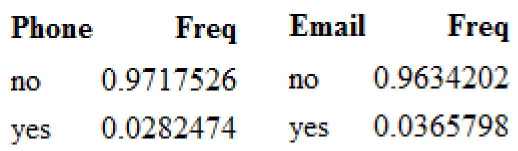
A Chi-Square Independence Test run on Heating and A/C yield a p-value of 2.2e^-16, which is less than the significance level at 5%. Thus, the null hypothesis stating Heating and A/C are independent is rejected. The test concludes there is a statistically significant relationship between Heating and A/C.

8. Email Addresses and Phone Numbers

To find out whether most postings hid their contact information, I computed the count of postings that show either phone or email by searching the data frame using regular expressions and the grepl function.

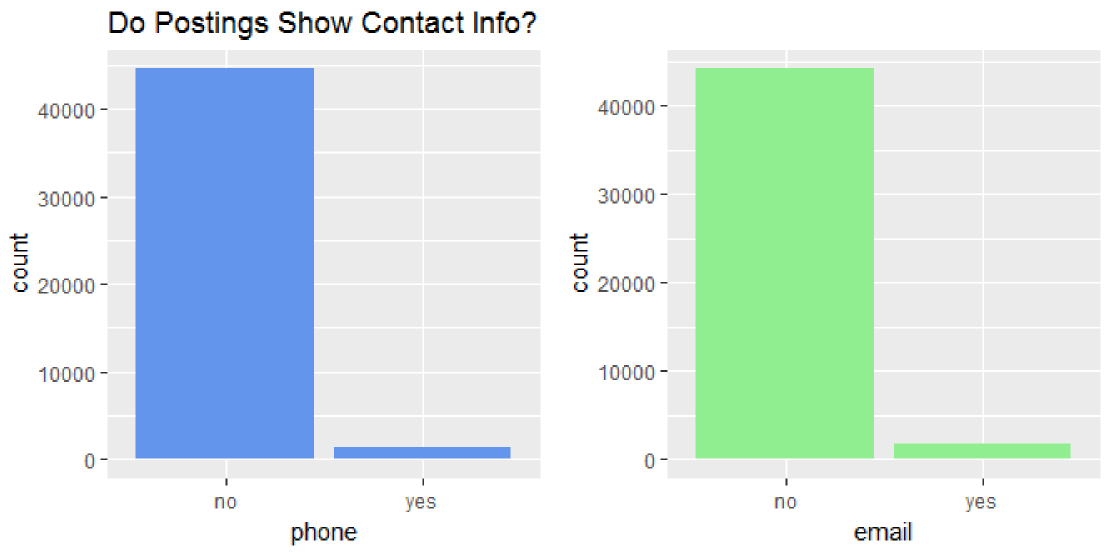
The regex pattern “\\(?\\d{3}\\)?[.-]? *\\d{3}[.-]? *[.-]?\\d{4}” can detect phone numbers. The first group, which corresponds to the area code, searches for digits that appear exactly three times enclosed in optional parentheses. This is followed by an optional dash, then it searches for digits that appear 3 times, followed by another optional dash, then finally ending with digits that appear 4 times. This pattern can detect phone numbers containing dashes, spaces, or neither between groups of digits. Since the area code is optional, it can also find phone numbers without an area code listed. The backslashes are necessary to use as escape characters.

The pattern “@” to search for emails is more straightforward. Since email addresses will contain an “@” symbol, the pattern will find all postings containing “@”.



Phone	Freq	Email	Freq
no	0.9717526	no	0.9634202
yes	0.0282474	yes	0.0365798

The proportion tables show a very small percentage of postings choose to show their phone number (at about 2.82%) or email (at about 3.66%).



The bar chart supports the findings from the table. Most people chose to hide their contact information from web scrapers.

Sources

Craigslist, www.craigslist.com/

H. Wickham. ggplot2: Elegant Graphics for Data Analysis.

H. Wickham. stringr: Simple, Consistent Wrappers for Common String Operations

Hadley Wickham, Romain Francois, Lionel Henry and Kirill Miller (2018). dplyr: A Grammar of Data Manipulation.

Piazza , www.piazza.com/.

Stack Overflow, stackoverflow.com/.

R Code

```
## ----setup, include=FALSE----------------------------------------------

knitr::opts_chunk$set(echo = FALSE, warning = FALSE, message = FALSE)

require(tidyverse)

require(ggpubr)

require(knitr)

require(kableExtra)



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

# 1.

# define a function to read one text post from a directory

# The parameter of the function is directory, the directory where the text file is stored

read_post <- function(directory) {

  readLines(directory)

}


# test the function

cl <- read_post("messy/losangeles/_ant_apa_d_1bd-1ba-tennis-court_6738904358.txt")


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


# 2.

# Define a function to read all text posts from a directory into a single data frame, using my previous read_post() function and parameter being directory


read_all_posts <- function(directory) {

  filenames = list.files(directory, pattern = "*.txt", full.names = TRUE, recursive = TRUE) # get the list of text files from the directory
  posts = lapply(X = filenames, FUN = read_post) # a list of all the posts read from the directory

  split = str_split_fixed(posts, "QR|Date Posted:", 3) # split posts into 3 columns for title, text, and attributes

  data_frame = data.frame(split) # convert to data frame

  price = substr(split[,1], start = 1, stop = 8)

  title_price = substr(price, start = 5, stop = 8)

  data_frame$title_price = as.numeric(gsub("\\D+", "", title_price)) # extract price from the title and define it as a new column in the data frame

  data_frame # return the final data frame

}


all <- read_all_posts("messy") # using the function to read all tiles from the "messy" folder and its subfolders into a single data frame, all


names(all) <- c("title", "text", "attribs", "title_price") # rename the columns


result <- str_split_fixed(all$attribs, "Price:|Latitude:|Longitude:|Bedrooms:|Bathrooms:|Sqft:", 7) # split the attribs column into 7 pieces


result <- as.data.frame(result) # convert result to data frame


names(result) <- c("date_posted", "price", "latitude", "longitude", "bdrms", "bthrms", "sqft") # rename the data frame


# converting the 7 columns to proper character or numeric format

result$date_posted <- as.character(result$date_posted)

result$price <- as.numeric(gsub("\\D+", "", result$price))

result$latitude <- as.numeric(gsub("\\D+", "", result$latitude))

result$longitude <- as.numeric(gsub("\\D+", "", result$longitude))
```

```
result$bdrms <- as.numeric(gsub("\\D+", "", result$bdrms))

result$bthrms <- as.numeric(gsub("\\D+", "", result$bthrms))

result$sqft <- as.numeric(gsub("\\D+", "", result$sqft))

result$date_posted <- strptime(result$date_posted, " %B %d, %Y at %H:%M") # convert to date-time format


# bind the two data frames into a final data frame

df_result <- cbind(all, result)


all <- df_result


all <- all[-match("attribs", names(all))]  # remove the old attribs column


all$date_posted <- as.Date(all$date_posted) # convert the date_posted column to a more convenient format



## ---- include = FALSE----------------------------------------------------

# 4.

# rental price from the title is stored in the "title_price" column

sum(is.na(all$title_price)) # 174 missing values from title_price


prices <- all %>%

  select(price, title_price)


price_match <- all$price == all$title_price

sum(price_match, na.rm = TRUE) # number of TRUE values where price matches the title price

nrow(all) - sum(price_match, na.rm = TRUE) # number of FALSE values where price and title price don't match
```

```
# calculate the difference between user and title price for non-matching rows


nonmatches <- all[which(all$price != all$title_price),]

nonmatches$price_diff <- nonmatches$price - nonmatches$title_price


nonmatches <- nonmatches %>%

  select(price, title_price, price_diff)


summary(nonmatches$price_diff) # how much do the prices differ - a summary of price differences


price_diff_plot <- ggplot(nonmatches, aes(x = price_diff)) + geom_density() + xlim(5000, 20000) + labs(title = "User and Title Price Differences", x = "Price Difference")


all$price_diff <- all$price - all$title_price

which(abs(all$price_diff) > 20000)

all[8984,] # one bdrm and two bdrm different prices mistakenly combined into one price



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

price_diff_plot


## ---- include = FALSE----------------------------------------------------

# 5.

# Extract deposit amount from text of the posting

deposit1 <- str_extract(all[,"text"], regex("(?<=deposit: )[^ ]*(?= )", ignore_case = TRUE))

deposit2 <- gsub("[^[:digit:]. ]", "", deposit1) # deposit amount
```

```
all$deposit <- as.numeric(deposit2) # add deposits as new column of data frame


subset <- subset(all,
                 !(is.na(all$deposit))) # filter out rows with NAs


subset2 <- subset %>%
  filter(deposit > 11) %>% # filter by deposits greater than 11 (smaller deposits are likely to be mistakes)
  arrange(deposit)


# scatterplot of price vs deposit
price_v_deposit <- subset2 %>%
  ggplot(aes(x = title_price, y = deposit)) +
  geom_point(alpha = 0.5) +
  geom_smooth() +
  ylim(0, 6000) +
  xlim(0, 7500) +
  labs(title = "Price vs Deposit", x = "Price", y = "Deposit")


# histograms for price and deposit
price_distr <- subset2 %>%
  ggplot(aes(x = title_price)) +
  geom_histogram(fill = "cornflowerblue", color = "white") +
  xlim(0,7500) +
  labs(title = "Distribution of Title Price", x = "Title Price")


dep_distr <- subset2 %>%
  ggplot(aes(x = deposit)) +
  geom_histogram(fill = "lightgreen", color = "white") +
  xlim(0, 7500) +
  labs(title = "Distribution of Deposit", x = "Deposit")
cor(x = subset2$title_price, y = subset2$deposit)



## ---- fig.width = 5, fig.height = 3-------------------------------------
price_v_deposit


## ---- fig.width = 8, fig.height = 3-------------------------------------
ggpubr::ggarrange(price_distr, dep_distr, ncol = 2)


## ---- include = FALSE---------------------------------------------------
# 6.
# Using the grepl function to detect pets inside the text of the dataframe
vector1 <- ifelse(grepl("no pets|no pet", all$text, ignore.case = TRUE),"none", NA)
vector2 <- ifelse(grepl("cats", all$text, ignore.case = TRUE), "cats", NA)
vector3 <- ifelse(grepl("dogs", all$text, ignore.case = TRUE), "dogs", NA)
vector4 <- ifelse(grepl("pet friendly|pet-friendly|cats and dogs|dogs and cats", all$text, 
ignore.case = TRUE), "both", NA)


# combine all vectors by NAs, replace the remaining NA with "both" pets policy, add the 
final vector as a column to the all data frame


vector1[is.na(vector1)] <- vector2[is.na(vector1)]
vector3[is.na(vector3)] <-  vector1[is.na(vector3)]
vector4[is.na(vector4)] <- vector3[is.na(vector4)]
table(vector4) # shows the counts for both, cats, dogs, and none variables
all$pets <- vector4
```

```
fish <- all %>%
  filter(str_detect(text, " fish"))


birds <- all %>%
  filter(str_detect(text, " birds"))


# in addition to dogs and cats, some apartments accept birds and fish as pets


# extract pet deposit amounts from text
pet_deposit <- all %>%
  filter(str_detect(text, "pet deposit"))


pet_deposit_amount <- str_extract(pet_deposit$text, "([^.][^.][^.][^.][^.][^.][^.][^.] pet
deposit)")


pet_deposit_amount <- gsub("[^[:digit:]. ]", "", pet_deposit_amount) # deposit amount


pet_deposit$pet_deposit_amount <- as.numeric(pet_deposit_amount) # convert to numeric


# Removing NAs from the data
pet_deposit <- pet_deposit[!is.na(pet_deposit$pet_deposit_amount),]
pet_deposit <- pet_deposit[!is.na(pet_deposit$pets),]


# Removing errors due to low amount or "none" pet policy
pet_deposit <- pet_deposit %>%
  filter(pet_deposit_amount > 10 & pets != "none")
```

```
# Histogram of pet deposit distribution

histplot <- ggplot(pet_deposit, aes(x = pet_deposit_amount)) + geom_histogram(aes(fill = pets), position = "stack", binwidth = 100, color = "black") + labs(title="Distribution of Pet Deposits", x="Pet Deposit Amount", y = "Count")


# Violinplot of pet deposit distribution

pet_violinplot <- ggplot(pet_deposit, aes(x = pets, y = pet_deposit_amount)) + geom_violin(aes(fill = pets), color = "black") + labs(title = "Violin Plots of Pet Deposits", x = "Pet Deposit Amount")


# Get proportions of pet_deposit_amounts

prop.table(table(pet_deposit$pet_deposit_amount))


# Investigating outliers

which(pet_deposit$pet_deposit_amount > 750)

pet_deposit[366,] # cats with 1000 pet deposit

pet_deposit[367,]


## ---- fig.width = 7, fig.height = 3.5------------------------------------

# types of pets graphic

pets_subset <- subset(all, (!is.na(all[,"pets"])))


ggplot(pets_subset, aes(x = pets)) + geom_bar(aes(fill = pets)) + labs(title = "Apartment Pet Policies")


## ---- fig.width = 8, fig.height = 3.5------------------------------------

ggpubr::ggarrange(histplot, pet_violinplot, ncol = 2)


## ---- include = FALSE----------------------------------------------------

# 7: Heating
# -------------------------

heating1 <- ifelse(grepl("heater", all$text, ignore.case = TRUE),"heater", NA)

heating2 <- ifelse(grepl("fireplace|fire place|wood-burning stove|wood burning stove",
all$text, ignore.case = TRUE), "fireplace", NA)

heating3 <- ifelse(grepl("heater.*fireplace|fireplace.*heater|heater.*wood-burning
stove|wood-burning stove.*heater|heater.*wood burning stove|wood burning
stove.*heater", all$text, ignore.case = TRUE), "both", NA)


# combine all vectors by NAs


heating1[is.na(heating1)] <- heating2[is.na(heating1)]

heating3[is.na(heating3)] <-  heating1[is.na(heating3)]

heating3[is.na(heating3)] <- "none"


all$heating <- heating3


table(all$heating)


# AC

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

ac <- ifelse(grepl("air conditioning|air-conditioning| ac | a/c |a/c", all$text, ignore.case =
TRUE),"yes", "no")

table(ac)


all$ac <- ac


# Percentage of postings with AC

prop.table(table(all$ac))


# Percentage of postings with heating
```

```
prop.table(table(all$heating))


# from the proportion of tables, we see a slightly greater percentage of apartments have
heating



## ---- fig.width = 3, fig.height = 3-------------------------------------------
kable(prop.table(table(all$ac)), col.names = c("A/C", "Freq"))
kable(prop.table(table(all$heating)), col.names = c("Heating", "Freq"))


## ---- fig.width = 7, fig.height = 3.5-----------------------------------------
# Graphics that display the covariate relationship between ac and heating
# geom_tile graphic:
all %>%
  count(heating, ac) %>%
  ggplot(aes(x = heating, y = ac)) +
  geom_tile(aes(fill = n)) +
  scale_fill_continuous(low = "light grey", high = "dark red", name = "Count") +
  labs(x = "Heating", y = "Air Conditioning", title = "Air Conditioning vs Heating")



## -----------------------------------------------------------------------------
# bar plot:
all %>%
  ggplot() + geom_bar(aes(x = heating, fill = ac), position = "dodge") +
  scale_fill_discrete(name = "A/C") +
  labs(title = "Distribution of Heating and A/C", x = "Heating", y = "Count")
```

```
## ---- include = F---------------------------------------------------------
# Chi-Square Test of Independence between Heating and AC categorical variables

# H0: AC and Heating are independent
# HA: AC and Heating are not independent

# Creating a 2x2 contigency table for the Chi-Square Test:
ac_yes <- all %>%
  filter(ac == "yes")
ac_yes_heating_yes <- ac_yes %>%
  filter(heating == "fireplace" | heating == "heater" | heating == "both")
# 3032 obs with both ac and heating
ac_yes_heating_no <- ac_yes %>%
  filter(heating == "none")
# 6964 obs with ac and no heating
ac_no <- all %>%
  filter(ac == "no")
ac_no_heating_yes <- ac_no %>%
  filter(heating == "fireplace" | heating == "heater" | heating == "both")
# 7191 obs with heating and no ac
ac_no_heating_no <- ac_no %>%
  filter(heating == "none")
# 28658 obs with no heating and no ac

# Creating the final matrix
chisqmatrix <- matrix(data = c(3032, 7191, 6964, 28658), nrow = 2, ncol = 2, byrow = T)
dimnames(chisqmatrix) <- list(Heating = c("Y", "N"), AC = c("Y", "N"))
```

```
# Running the Chi-Square Test

chi1 <- chisq.test(chisqmatrix, correct = F)

chi1


# Since the p value is less than the significance level, we reject the null. Therefore, AC and
Heating must have some relationship.


## ---- include = FALSE--------------------------------------------------

# Writing a regex that will detect phone numbers or email addresses, then count the
number of postings that contain this contact info


phone_pattern <- "\\(?\\d{3}\\)?[.-]? *\\d{3}[.-]? *[.-]?\\d{4}"

email_pattern <- "@"


phone <- ifelse(grepl(phone_pattern, all$text, ignore.case = TRUE),"yes", "no")

email <- ifelse(grepl(email_pattern, all$text, ignore.case = TRUE),"yes", "no")

table(phone)

table(email)

all$phone <- phone

all$email <- email


phoneplot <- ggplot(all, aes(x = phone)) + geom_bar(fill = "cornflowerblue") +
labs(title="Do Postings Show Contact Info?")

emailplot <- ggplot(all, aes(x = email)) + geom_bar(fill = "light green") + labs(title="")



## ---- fig.width = 3, fig.height = 3-------------------------------------

# Proportion tables of postings that show/hide their phone or email

kable(prop.table(table(all$phone)), col.names = c("Phone", "Freq"))
kable(prop.table(table(all$email)), col.names = c("Email", "Freq"))


## ---- fig.width = 7, fig.height = 3.5------------------------------------
ggpubr::ggarrange(phoneplot, emailplot, widths = c(1,1), heights = c(1,1))


# as the bar plots show, very few percentage of postings show phone or email
```