A Data Visualization Report on The London 2017 Housing Prices

By

Boluwatife Olajuwon

S2219503

April 2023

London Housing Data

Boluwatife Olajuwon

2023-04-25

Introduction

The UK housing prices paid dataset contains the sales of property in England and Wales. The different columns in the dataset are explained below. The transaction identifier is a unique reference number, that is automatically generated when a sale is published and will change any time a sale is recorded.

The price on the transfer deed is stated under the Price column. The transfer date represents the date when the sale was completed.

There are different Property Types in the dataset, D represents a Detached house, S a Semi-Detached, T is a Terraced house, F is for Flats or Maisonettes and O is for other, which represents property types not covered by existing values.

Old and New column points to the age of the property, with Y representing a newly built property and N is an established residential building.

Duration represents the tenure, either Freehold or Leasehold.

The Town/City represents the city where the property was sold, the district represents the exact location of the property in the City and County.

Research Questions

This report is focused on the City of London and the year 2017. There are a couple of questions that this report will answer.

1. What is the average price of housing in London in 2017?
2. Which districts are the most expensive and least expensive in London, in the year 2017?
3. What type of property is the most expensive and most sought after in the year 2017, in London?

The first code chunk contains different lines of code, loading different R libraries needed to bring this report to life.

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.
3.2 ──
```

```
## ✓ ggplot2 3.4.0     ✓ purrr   1.0.1
## ✓ tibble  3.1.8     ✓ dplyr   1.1.0
## ✓ tidyr   1.3.0     ✓ stringr 1.5.0
## ✓ readr   2.1.3     ✓ forcats 1.0.0
## ── Conflicts ─────────────────────────────────────── tidyverse_conflict
s() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()

library(ggplot2)
library(dplyr)
library(lubridate)

##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union

library(tmap)
library(sf)

## Linking to GEOS 3.9.3, GDAL 3.5.2, PROJ 8.2.1; sf_use_s2() is TRUE
```

The UK price paid records is read into RStudio using the read.csv code. The dataset is quite heavy and robust with 22489348 rows and 11 columns. All the columns are of the character class.The Price column is the only one with integers. The Price variable and the District variable will be focused on in this report.

Characteristics of the Variable of Interest

The variable of interest in this report is the Price of the property and the District where the property is located.

Price of the Property

The price column has a minimum of 1 and a max price of 98900000 in pounds. A mean of 178244 pounds and a median price of 130000 pounds. The first and third quartile are 75000 and 210000 pounds respectively.

```
#Read Document
house_data <- read.csv("C:/Users/TifeN/OneDrive/R Studio/price_paid_records.c
sv")
# Show the summary of the dataset
summary(house_data)

##  Transaction.unique.identifier     Price          Date.of.Transfer
##  Length:22489348               Min.   :       1   Length:22489348
```

```
##  Class :character              1st Qu.:   75000   Class :character
##  Mode  :character              Median :  130000   Mode  :character
##                                Mean   :  178244
##                                3rd Qu.:  210000
##                                Max.   :98900000
##  Property.Type        Old.New            Duration          Town.City
##  Length:22489348    Length:22489348    Length:22489348    Length:22489348
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##    District            County          PPDCategory.Type
##  Length:22489348    Length:22489348    Length:22489348
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##  Record.Status...monthly.file.only
##  Length:22489348
##  Class :character
##  Mode  :character
##
##
##
```

```
# show the columns and rows of the dataset
dim(house_data)
```

```
## [1] 22489348       11
```

```
# check for any null values in the dataset
any(is.na(house_data))
```

```
## [1] FALSE
```

```
# list the columns in the dataset
names(house_data)
```

```
##  [1] "Transaction.unique.identifier"     "Price"
##  [3] "Date.of.Transfer"                  "Property.Type"
##  [5] "Old.New"                           "Duration"
##  [7] "Town.City"                         "District"
##  [9] "County"                            "PPDCategory.Type"
## [11] "Record.Status...monthly.file.only"
```

London 2017

Due to how large the original dataset is, it has been subsetted into a smaller dataset, focused on the city of London and the year 2017. To subset the original dataset, the code subset is used to focus on the city of the London from the main dataset, named house_data. To call on the year 2017 from the new dataset named london_data, the format of the Date of Transfer column must be set, so R knows which number is the year, the month and the day, as.Date is the code function that makes this happen. Since, we are focusing on the year 2017, and we have a new dataset named london_data, it makes it easier to get a new dataset.To create the dataset we want, we filter the year 2017 from the london_data and name it the london_2017_data. The next lines of code function display the dimension and summary of the new dataset, as well as if the new dataset has any null values. The new dataset has only 25060 rows and 11 columns, which is easier to work with than house_data. It has no null values.

Characteristics of the Variable of Interest

The new dataset, london_2017_data still has the same variable of interest.

Price

In the year 2017, a property sold for 98446300 pounds, the highest price in the column, with the minimum price being 100 pounds. The mean price is 915983 pounds and a median of 510000 pounds. The first and third quartile are 360000 and 800000 pounds respectively.

District

The Wandsworth district has the most property sales in the column. The City of Westminster has the highest average price in the column, which also has the highest total sales price.

```
# subsetting the main data to only London and year 2017
london_data <- subset (house_data, Town.City == "LONDON")
london_year_data <- subset (london_data, Date.of.Transfer == 2017)
london_data$Date.of.Transfer <- as.Date(london_data$Date.of.Transfer, format
= "%Y-%m-%d")
london_2017_data <- london_data %>%
  filter(year(Date.of.Transfer) == 2017)
#Check the columns in the new data
glimpse(london_2017_data)

## Rows: 25,060
## Columns: 11
## $ Transaction.unique.identifier     <chr> "{5376B385-F02A-34C1-E053-6B04A8
C09F…
## $ Price                             <int> 540000, 815000, 1100000, 2000000
, 98…
```

```
## $ Date.of.Transfer                   <date> 2017-05-24, 2017-04-28, 2017-04
-11,…
## $ Property.Type                      <chr> "T", "T", "T", "S", "T", "T", "T
", "…
## $ Old.New                            <chr> "N", "N", "N", "N", "N", "N", "N
", "…
## $ Duration                           <chr> "F", "F", "F", "F", "F", "F", "F
", "…
## $ Town.City                          <chr> "LONDON", "LONDON", "LONDON", "L
ONDO…
## $ District                           <chr> "MERTON", "MERTON", "MERTON", "R
ICHM…
## $ County                             <chr> "GREATER LONDON", "GREATER LONDO
N", …
## $ PPDCategory.Type                   <chr> "A", "A", "A", "A", "A", "A", "A
", "…
## $ Record.Status...monthly.file.only <chr> "A", "A", "A", "A", "A", "A", "A
", "…
```

```
#summary of the data
summary(london_2017_data)
```

```
##  Transaction.unique.identifier     Price          Date.of.Transfer
##  Length:25060                  Min.   :     100   Min.   :2017-01-01
##  Class :character              1st Qu.:  360000   1st Qu.:2017-02-09
##  Mode  :character              Median :  510000   Median :2017-03-17
##                                Mean   :  915983   Mean   :2017-03-20
##                                3rd Qu.:  800000   3rd Qu.:2017-04-28
##                                Max.   :98446300   Max.   :2017-06-28
##  Property.Type        Old.New            Duration          Town.City
##  Length:25060       Length:25060       Length:25060       Length:25060
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##    District            County          PPDCategory.Type
##  Length:25060       Length:25060       Length:25060
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##  Record.Status...monthly.file.only
##  Length:25060
##  Class :character
##  Mode  :character
##
##
##
```

```
#dimension of the data
dim(london_2017_data)

## [1] 25060    11

#check for null values
any(is.na(london_2017_data))

## [1] FALSE
```

To get a clearer picture of the price, the column is cut into groups and a new dataset, london_2017_grouped is created. The price column is broken into different price ranges.

```
# Define the breaks for the price range
breaks <- c(0, 100000, 200000, 300000, 400000, 500000, Inf)
# Use the cut() function to group the rows by price range
london_2017_grouped <- london_2017_data %>%
  group_by(price_range = cut(Price, breaks = breaks, include.lowest = TRUE))
```

The trend of different datasets being created is already obvious in this report, as we need to do this to get a clearer picture of the london_2017_data.

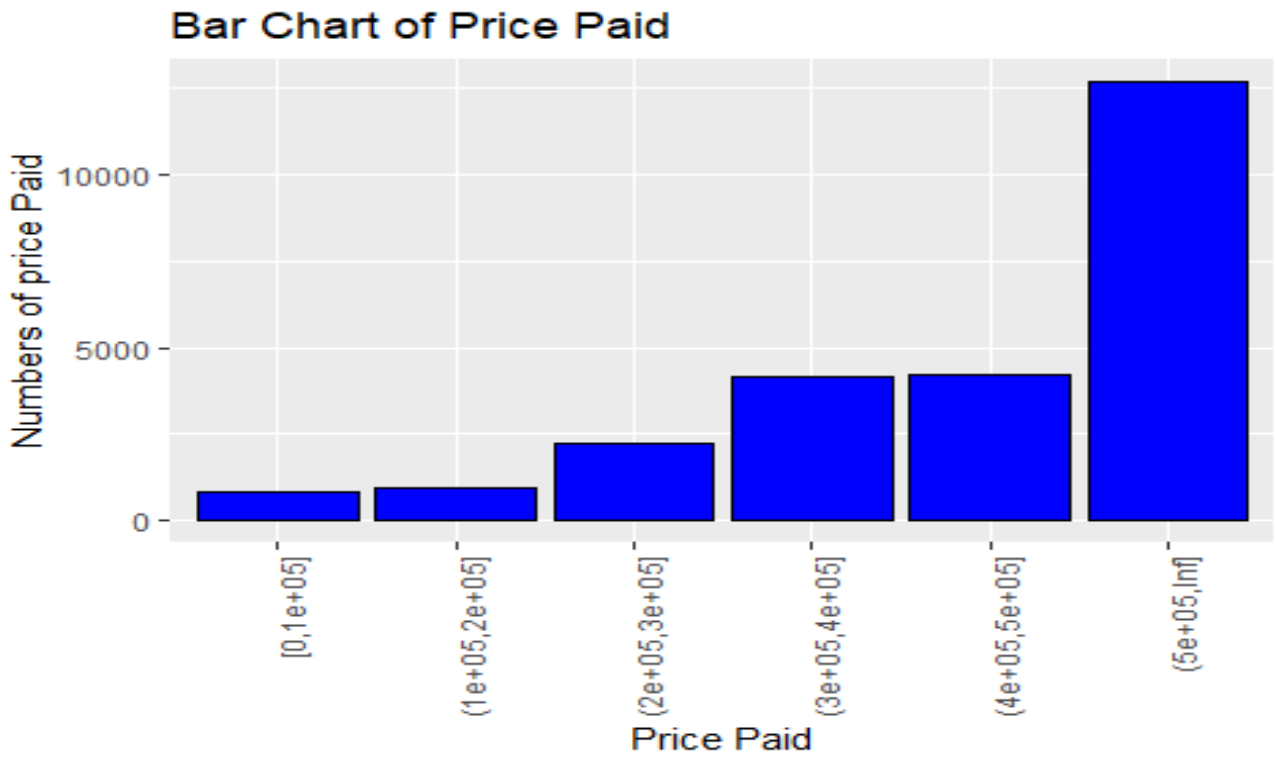
A new dataset that shows the average price by each date of transfer is created.

```
# group the data by transaction date and calculate the average price paid
avg_price_by_date <- london_2017_data %>%
  group_by(Date.of.Transfer) %>%
  summarize(avg_price = mean(Price))
```

The Plots

The next set of code chunks show the different plots for exploring the data, london_2017_data. The first plot is a bar chart of the grouped price paid.The chart shows that most property sold between the price range of 50 000 pounds and the highest price in the dataset, 98446300 pounds.

```
# Barchart of Price Paid in the data
ggplot(london_2017_grouped, aes(x = price_range)) +
  geom_bar(fill = "blue", color = "black") +
  ggtitle("Bar Chart of Price Paid") +
  xlab("Price Paid") +
  ylab("Numbers of price Paid")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

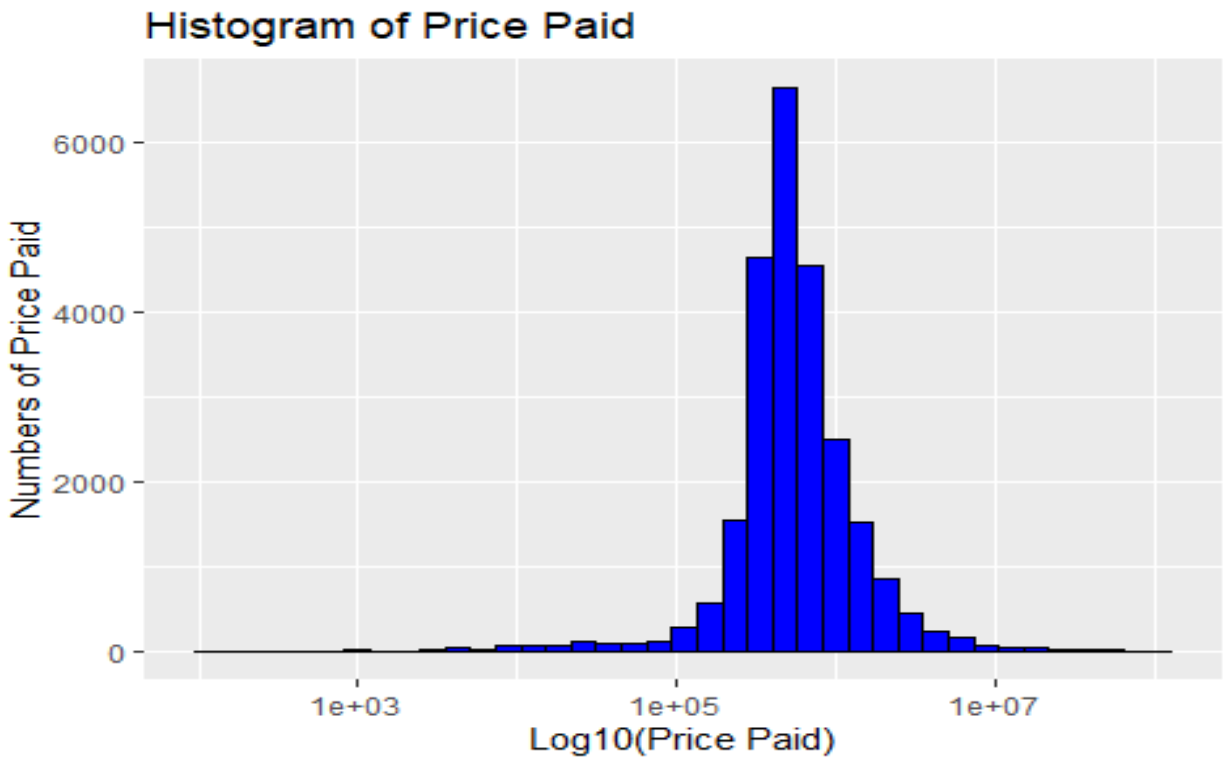



```
theme(plot.width = 10, plot.height = 6)

## List of 2
##  $ plot.width : num 10
##  $ plot.height: num 6
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

Histogram

The second chat is a histogram of the price paid from the dataset, london_2017_data, with the x axis in a logarithmic scale to fit the wide price range. The chart is negatively skewed, however, the positive side has the highest prices in the distribution.

```
# Histogram plot with price paid on a logarithmic scale
ggplot(london_2017_data, aes(x = Price)) +
  geom_histogram(bins = 39, fill = "blue", color = "black") +
  scale_x_log10()+
  ggtitle("Histogram of Price Paid") +
  xlab("Log10(Price Paid)") + ylab("Numbers of Price Paid")
```




```
theme(plot.width = 10, plot.height = 6)
```

```
## List of 2
##  $ plot.width : num 10
##  $ plot.height: num 6
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

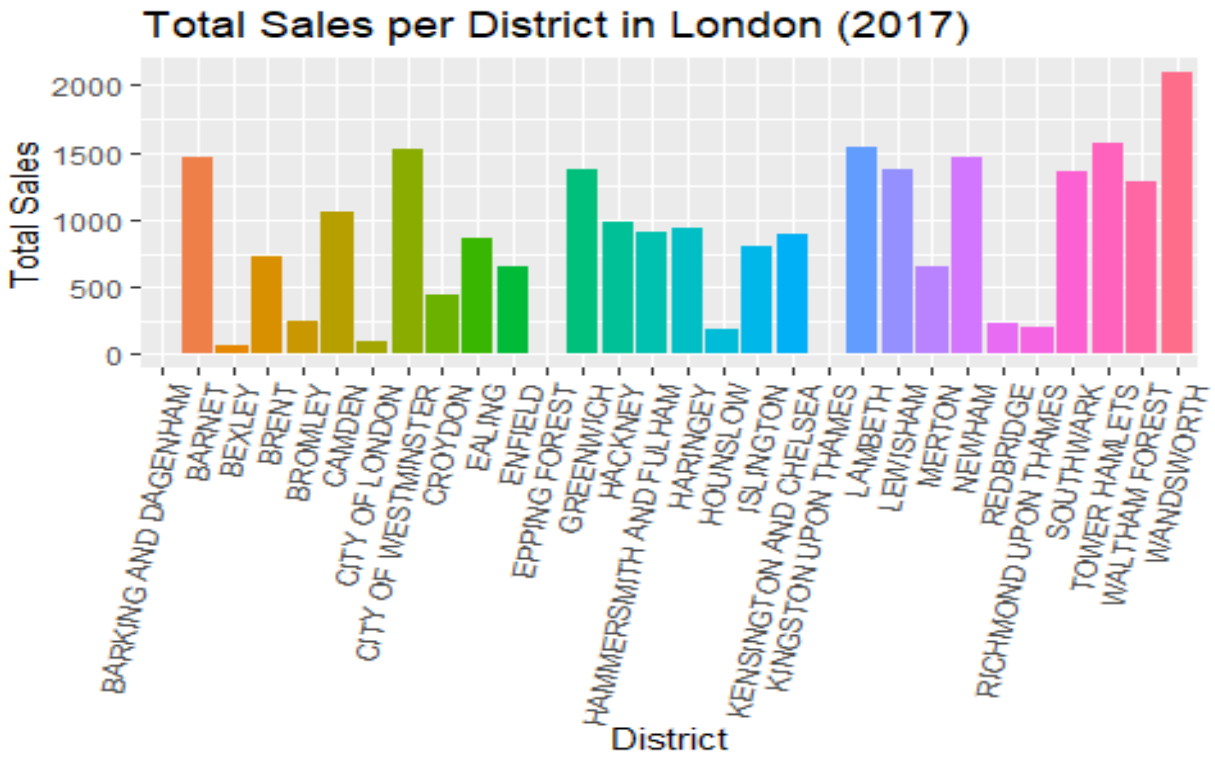
Bar Chart of District

How many property was sold in each district? This question is answered with a bar chart of the district, which shows that the Wandsworth district has the highest number of property sales. Three districts have the lowest property sales, Barking and Dagenham, Epping Forest and Kingston upon Thames districts.

```
# Bar Chart of District
# First, group the data by district and calculate the total number of sales
district_sales <- london_2017_data %>%
  group_by(District) %>%
  summarize(Total_Sales = n()) %>%
  arrange(desc(Total_Sales))

# Plot the data as a bar chart
ggplot(district_sales, aes(x = District, y = Total_Sales, fill = District)) +
  geom_bar(stat = "identity") +
  theme(axis.text.x = element_text(angle = 80, hjust = 1)) +
```

```
  xlab("District") +
  ylab("Total Sales") +
  ggtitle("Total Sales per District in London (2017)")+
  guides(fill = FALSE)
```

```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none
" instead as
## of ggplot2 3.3.4.
```

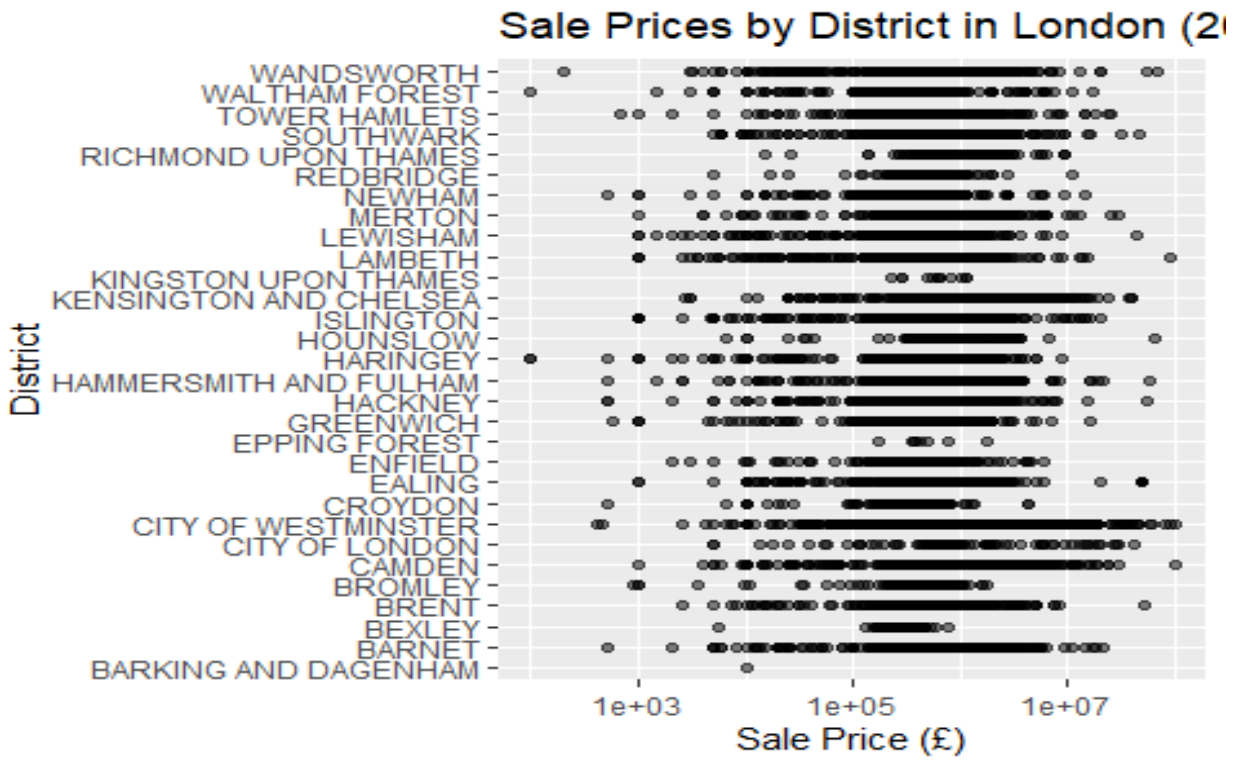



```
theme(plot.width = 10, plot.height = 6)
```

```
## List of 2
##  $ plot.width : num 10
##  $ plot.height: num 6
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

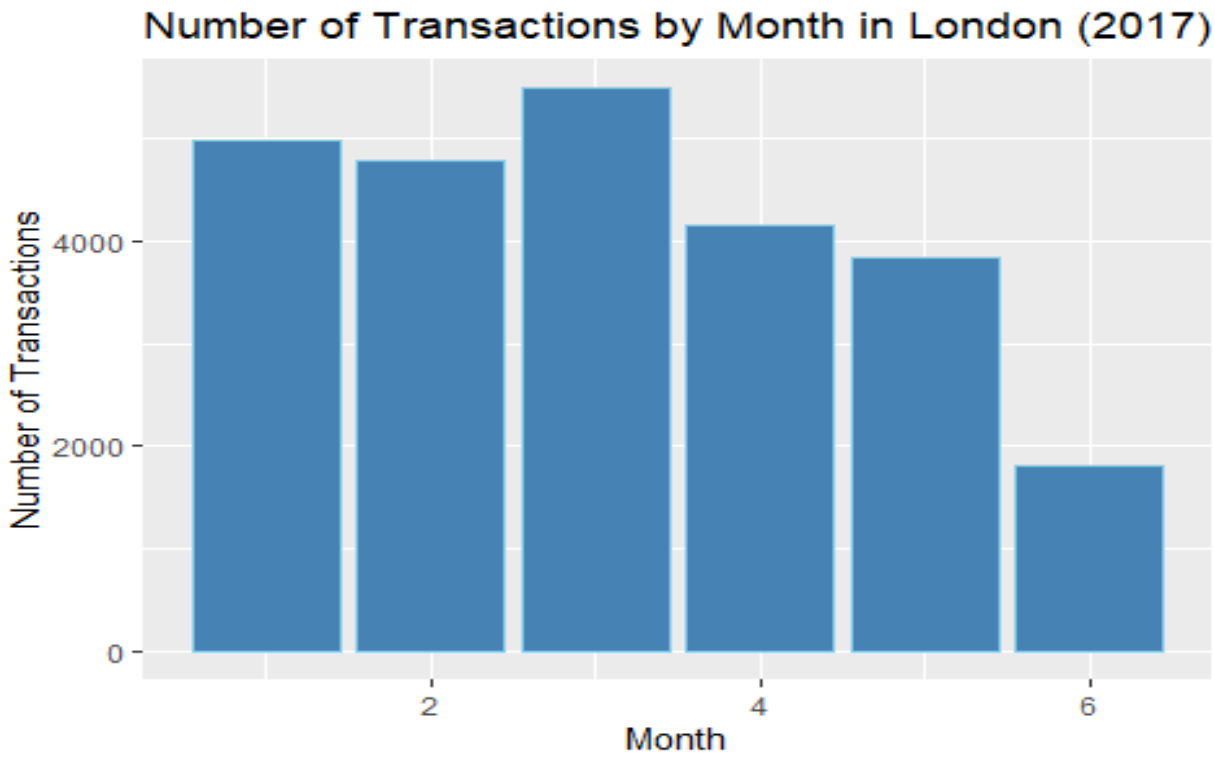
A scatter plot gives a clearer picture of the sale price per district. It shows that Barking and Dagenham has the lowest sales price in the district.

```
# Scatter Plot of Sale Price per District
ggplot(london_2017_data, aes(y = District, x = Price)) +
  geom_point(alpha = 0.5) +
  scale_x_log10()+
  labs(title = "Sale Prices by District in London (2017)",
       x = "Sale Price (£)", y = "District")
```



A bar chart of the months in the year 2017, shows that March and January have the highest number of transactions. To be precise, most sales happened in the winter months of January and February and the first month of spring, March.

```
#Bar Chart of transactions per month
london_2017_data %>%
  mutate(month = month(Date.of.Transfer)) %>%
  count(month) %>%
  ggplot(aes(x = month, y = n)) +
  geom_bar(stat = "identity", fill = "steelblue", color = "skyblue") +
  labs(title = "Number of Transactions by Month in London (2017)",
       x = "Month", y = "Number of Transactions")
```

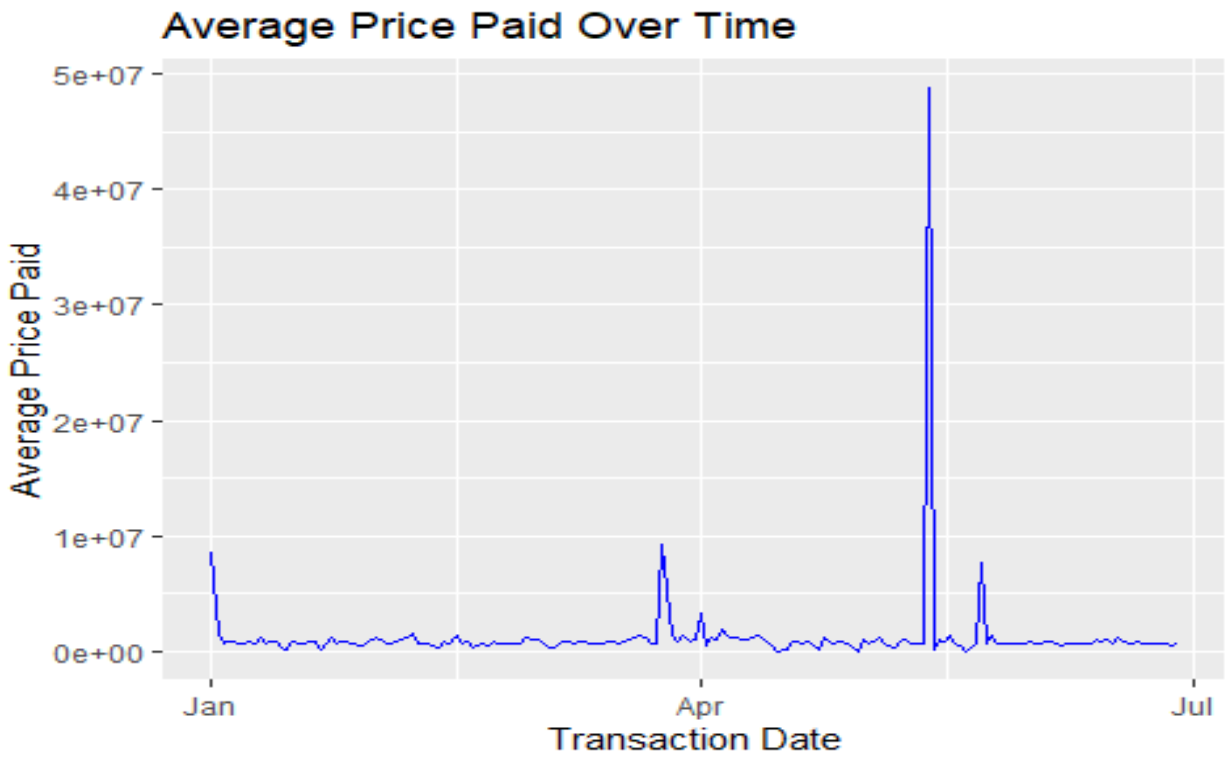



```
theme(plot.width = 10, plot.height = 6)
```

```
## List of 2
##  $ plot.width : num 10
##  $ plot.height: num 6
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

A line graph shows a spike of the average price paid between the months of May and June.

```
# create the line graph
ggplot(avg_price_by_date, aes(x = Date.of.Transfer, y = avg_price)) +
  geom_line(color = "blue") +
  ggtitle("Average Price Paid Over Time") +
  xlab("Transaction Date") +
  ylab("Average Price Paid")
```




```
theme(plot.width = 10, plot.height = 6)
```

```
## List of 2
##  $ plot.width : num 10
##  $ plot.height: num 6
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

Box Plot and Violin Plot

A violin plot is just like an advanced box plot, that does not just focus on the median, quartiles and range of the data. It provides a representation of the probability density of the data.

A box plot simply focuses on the statistics of the data and the outliers of the dataset.

Both charts are useful for comparing the variable between different groups.

The property type is plotted against the price. This compares the outliers and the descriptive statistics of the different types of property in the dataset. The property type O, which are the property types not covered by existing values has the most outliers, this is visible in both box and violin plots. The density curve of flats/maisonettes property type is wider than the other property types, which means it occurs more frequently in the dataset. The detached property type has the least distribution.

However, the outliers in this first set of plots make it quite impossible to fully visualize the density of the variables.

```
# create the violinplot
ggplot(london_2017_grouped, aes(x = Property.Type, y = Price)) +
  geom_violin(fill = "skyblue", alpha = 0.5) +
  labs(title = "Distribution of Property Prices by Property Type",
       x = "Property Type",
       y = "Price (in pounds)")
```


```
#create the boxplot
ggplot(london_2017_grouped, aes(x = Property.Type, y = Price)) +
  geom_boxplot(fill = "skyblue", alpha = 0.5) +
  labs(title = "Distribution of Property Prices by Property Type",
       x = "Property Type",
       y = "Price (in pounds)")
```


```
theme(plot.width = 10, plot.height = 6)
```

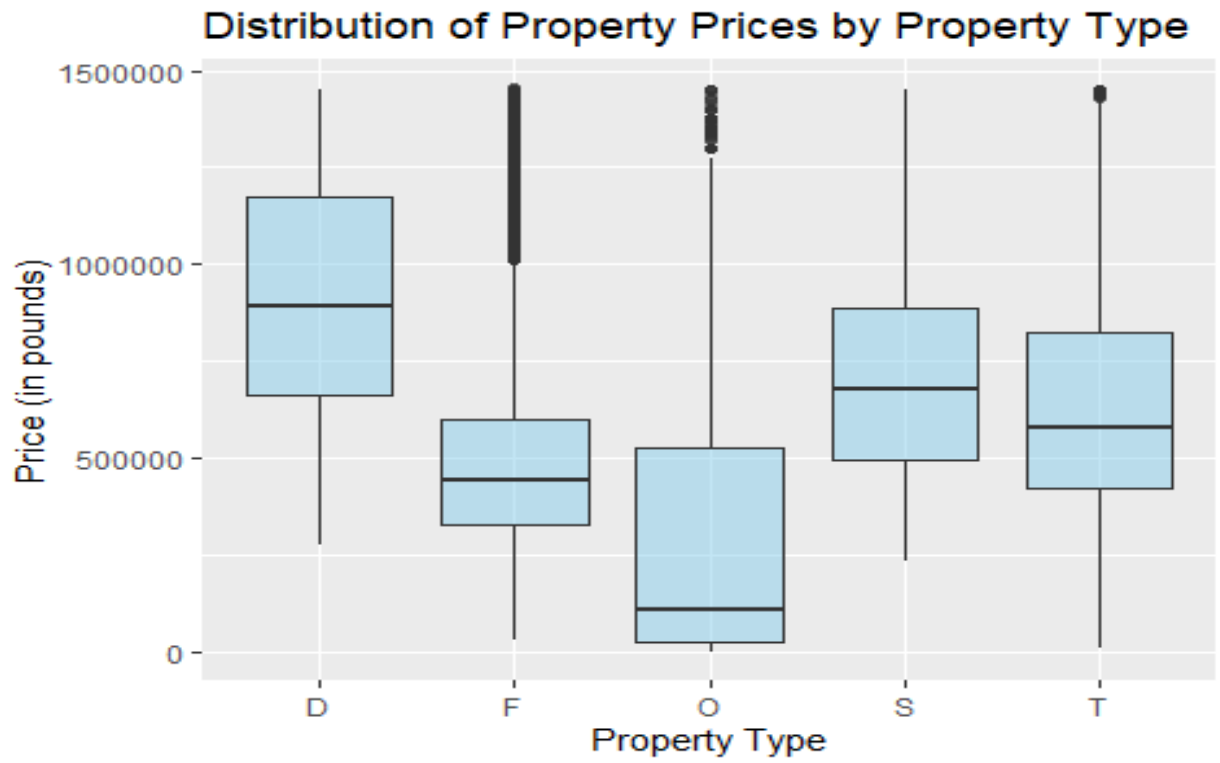
```
## List of 2
##  $ plot.width : num 10
##  $ plot.height: num 6
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

To get a clearer picture of the violin and box plots, the outliers must be removed. This is done by removing the upper and lower quartile. To achieve this, a new dataset is subsetted from the london_2017_data.

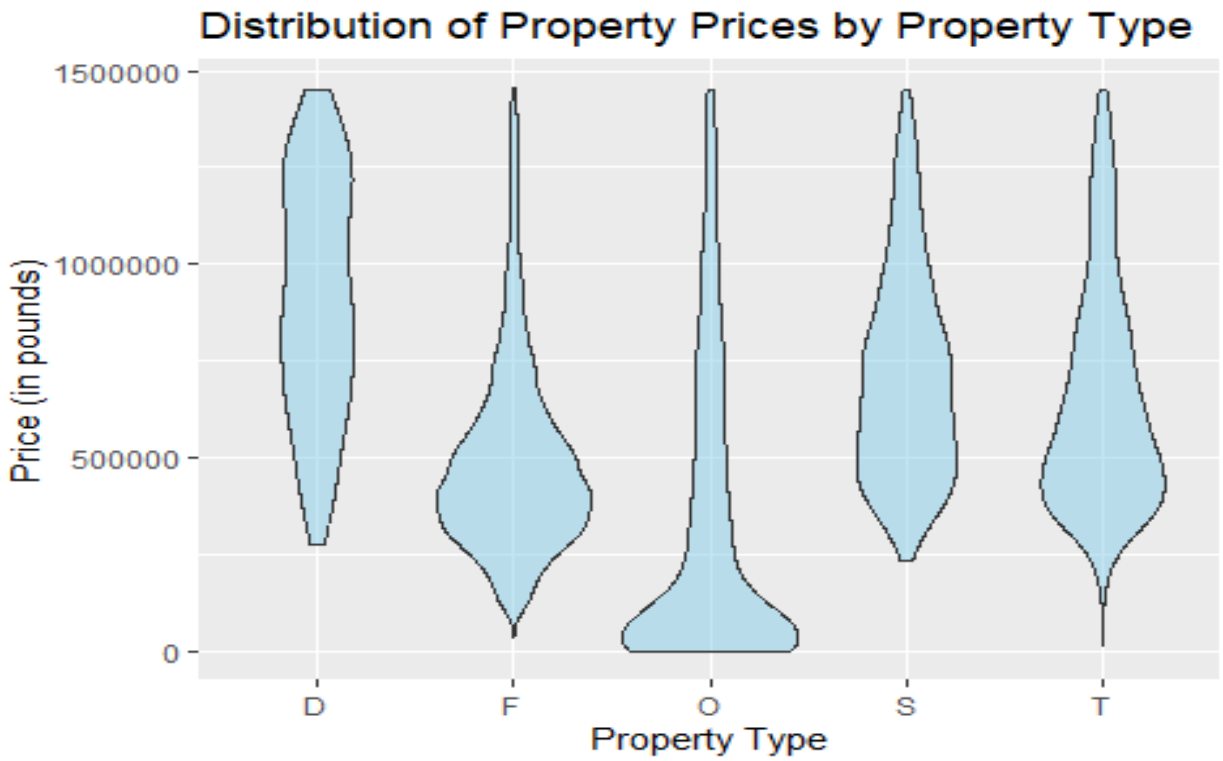
The new plots show us that detached property types command the highest price, based on the density plot made visible by the violin plot. The median price is also the highest among all property types. All property types are positively skewed, with the exception of the semi-detached property type, which is evenly distributed. The detached property type has a distribution of prices, evenly distributed over the range of 500000 and 1000000 pounds.There is a small distribution between 1000000 and the maximum price of 1400000 pounds thereabouts. The Flat property types is mainly distributed below the 500000 pounds sales price. The terraced property type has the highest distribution below the 500000 pounds sales price. Of all the property types, the detached property type has the smoothest distribution and density.

The first plot in this code chunk shows the box plot, the second plot shows the violin plot and the last plot shows a box plot embedded in a violin plot.

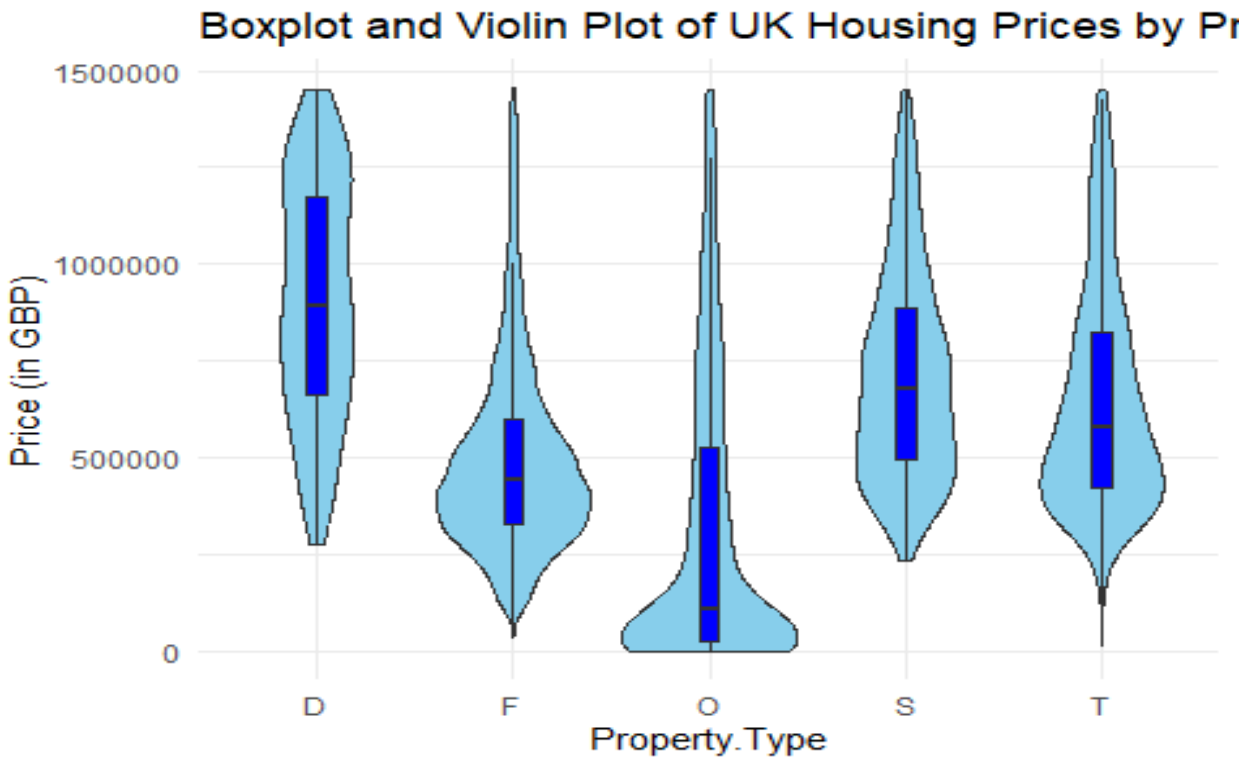
```r
#Remove the outliers
Q1 <- quantile(london_2017_data$Price, 0.25, na.rm = TRUE)
Q3 <- quantile(london_2017_data$Price, 0.75, na.rm = TRUE)
IQR <- Q3 - Q1
lower <- Q1 - 1.5 * IQR
upper <- Q3 + 1.5 * IQR
london_2017 <- subset(london_2017_data, london_2017_data$Price > lower & lond
on_2017_data$Price < upper)

#Create new boxplot without outliers
ggplot(london_2017, aes(x = Property.Type, y = Price)) +
  geom_boxplot(fill = "skyblue", alpha = 0.5) +
  labs(title = "Distribution of Property Prices by Property Type",
       x = "Property Type",
       y = "Price (in pounds)")
```




```r
#create new violinplot without outliers
ggplot(london_2017, aes(x = Property.Type, y = Price)) +
  geom_violin(fill = "skyblue", alpha = 0.5) +
  labs(title = "Distribution of Property Prices by Property Type",
       x = "Property Type",
       y = "Price (in pounds)")
```




```r
#create boxplot embedded in violin plot
ggplot(london_2017, aes(x=Property.Type, y=Price)) +
  geom_violin(fill = "skyblue") +
  geom_boxplot(width=0.1, fill="Blue", outlier.shape = NA) +
  theme_minimal() +
  xlab("Property.Type") +
  ylab("Price (in GBP)") +
  ggtitle("Boxplot and Violin Plot of UK Housing Prices by Property Type")
```



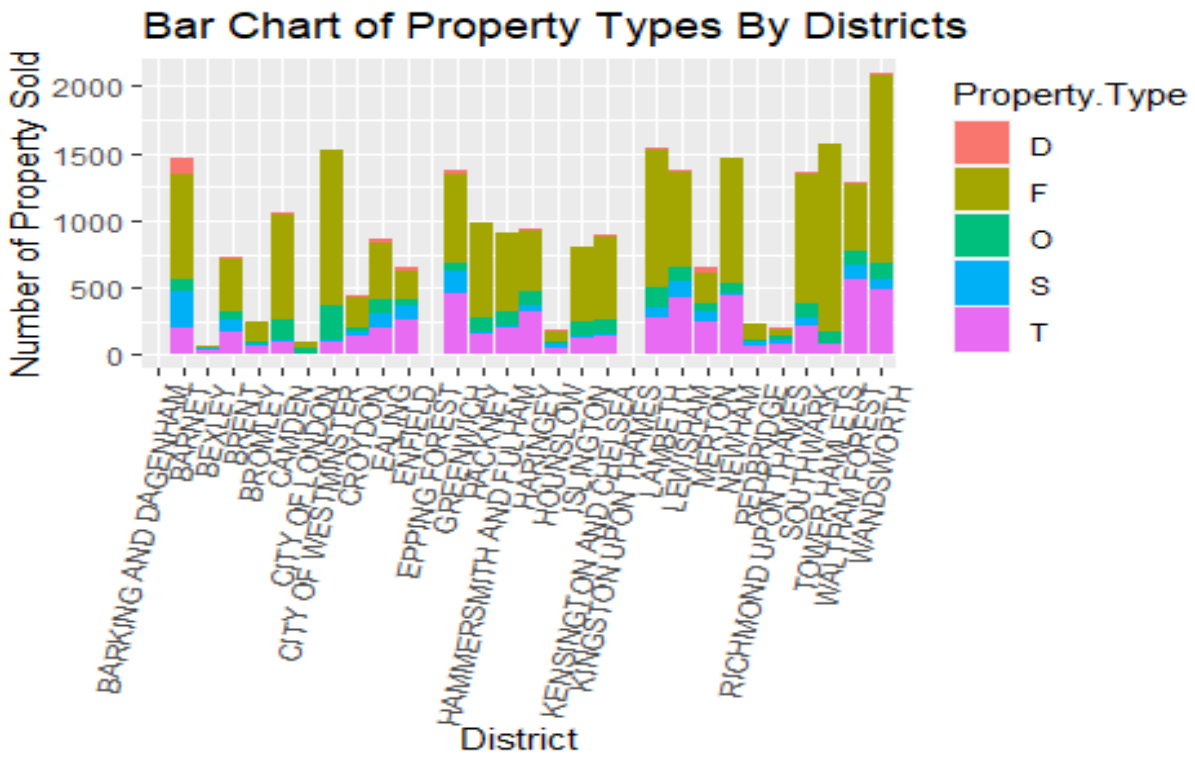

```
theme(plot.width = 10, plot.height = 6)
```

```
## List of 2
##  $ plot.width : num 10
##  $ plot.height: num 6
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

Bar Chart of Districts Showing Property Type

The bar chart created by this chunk code is a count of the property and property sales per district with a focus on the property type, using the fill function, the property types are well represented for each district. The flat property type is well represented for each district, as it is predominant on the chart. The terraced property type comes after . In the Tower Hamlets district, most of the property sold are flats, but in Barnet, the detached property type sold more than in any other district, while Waltham Forest recorded the highest sales of terraced property.

```
ggplot(london_2017_data, aes(x = District, fill = Property.Type)) +
  geom_bar()+
  theme(axis.text.x = element_text(angle =80, vjust = 1, hjust=1))+
  ggtitle("Bar Chart of Property Types By Districts")+
  ylab("Number of Property Sold")
```




```
theme(plot.width = 10, plot.height = 6)

## List of 2
##  $ plot.width : num 10
##  $ plot.height: num 6
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

Line Graph of Property Average Price

This graph shows how the price of the different property types changes over the different months in the year 2017. The price of the property type other, steadily rises from January to May, then sees a sharp decline from May to June. The detached property type rises and drops over the year 2017, the best time to buy is February and to turn a profit, you sell in April. The flat property type, maintains close to the same price over the year 2017.The same goes for the terraced and semi-detached property type.

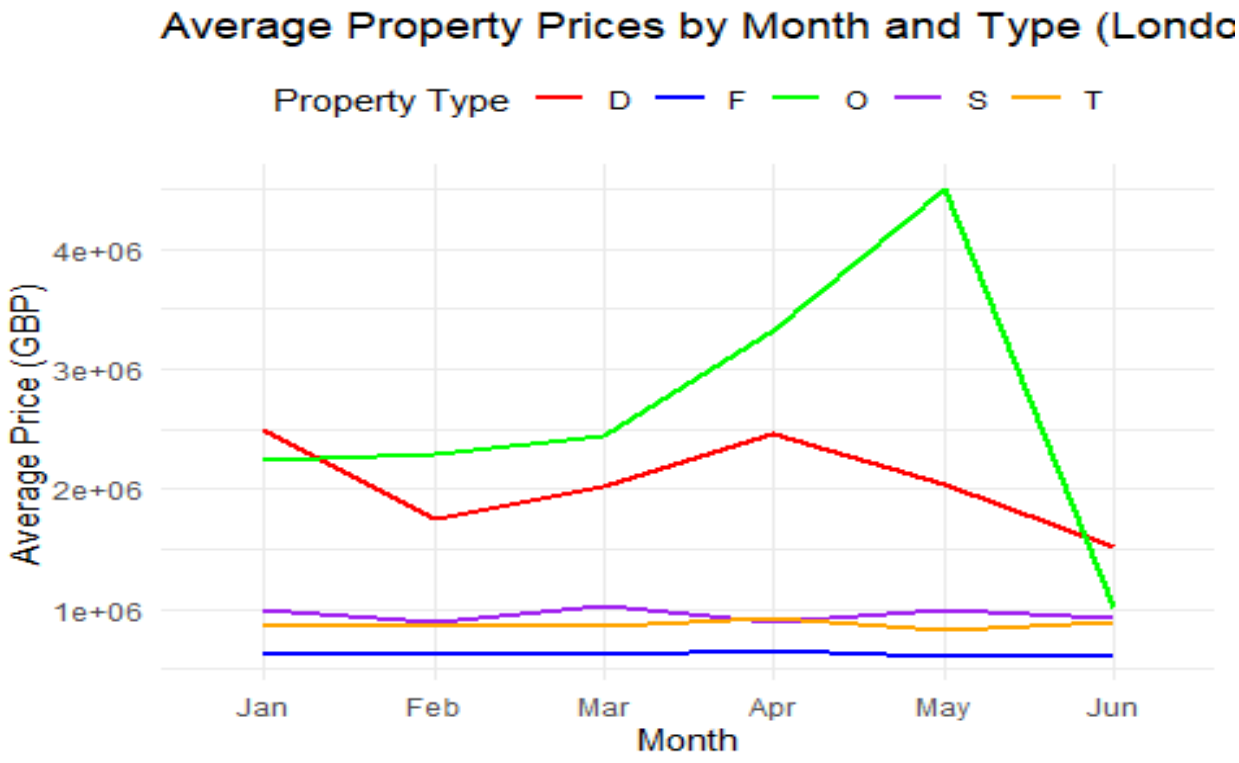
```
# Convert Date of transfer to a date format
london_2017_data$Date.of.Transfer <- as.Date(london_2017_data$Date.of.Transfe
r, format = "%Y/%m/%d")

# Create a new column for month
london_2017_data$Month <- month(london_2017_data$Date.of.Transfer, label = TR
UE)
#Average Price by Month
price_by_month <- london_2017_data %>%
```

```
  group_by(Property.Type, Month ) %>%
  summarise(Avg_Price = mean(Price))
```

```
## `summarise()` has grouped output by 'Property.Type'. You can override usin
g the
## `.groups` argument.
```

```
#line graph of Property average price by month
ggplot(price_by_month, aes(x = Month, y = Avg_Price, group = Property.Type, c
olor = Property.Type)) +
  geom_line(size = 1) +
  scale_color_manual(values = c("red", "blue", "green", "purple", "orange"))
+
  theme_minimal() +
  labs(title = "Average Property Prices by Month and Type (London 2017)",
       x = "Month",
       y = "Average Price (GBP)",
       color = "Property Type") +
  theme(legend.position = "top", legend.direction = "horizontal")
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
```




```
theme(plot.width = 10, plot.height = 6)
```

```
## List of 2
##  $ plot.width : num 10
##  $ plot.height: num 6
```

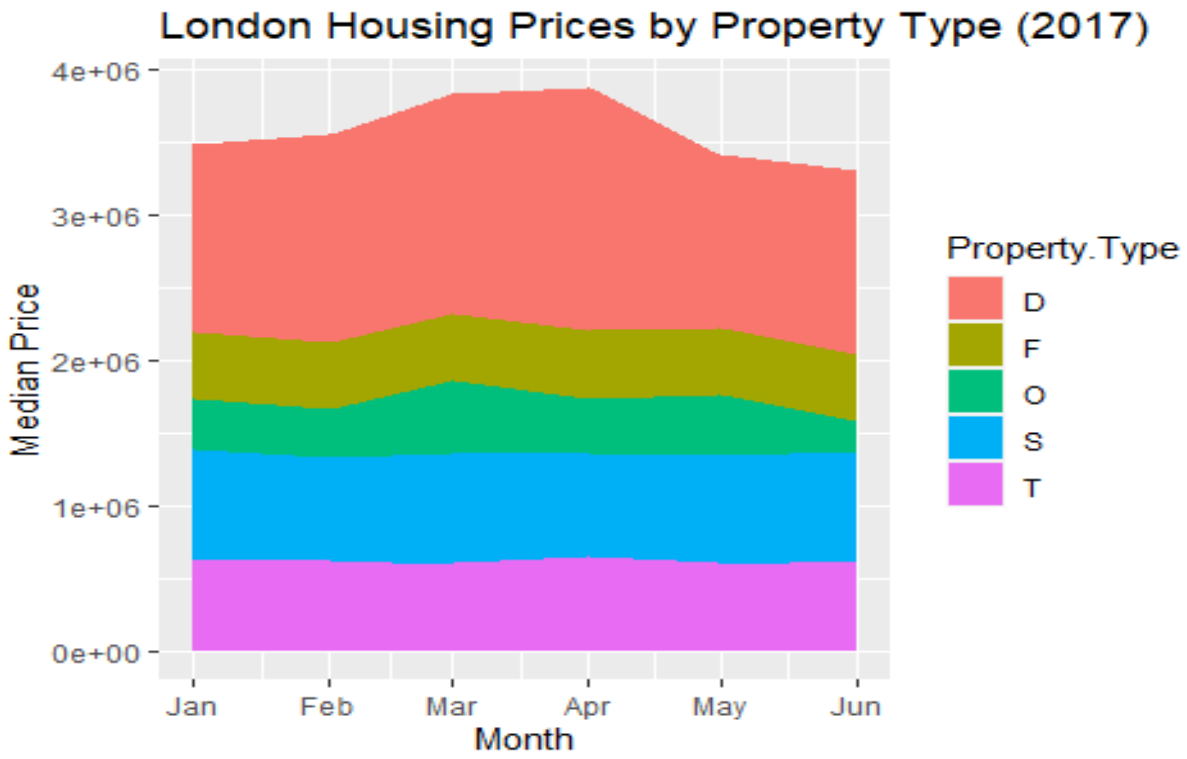
```
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

Area Chart

In this code chunk, the area chart of the median price is plotted by months, with a focus on the property types. Only six months are viable in the year 2017. The median price is considered in the new dataset, london_monthly, which is piped from the original london_2017_data. The wide range in the price column is why we need the median price of the sales price.

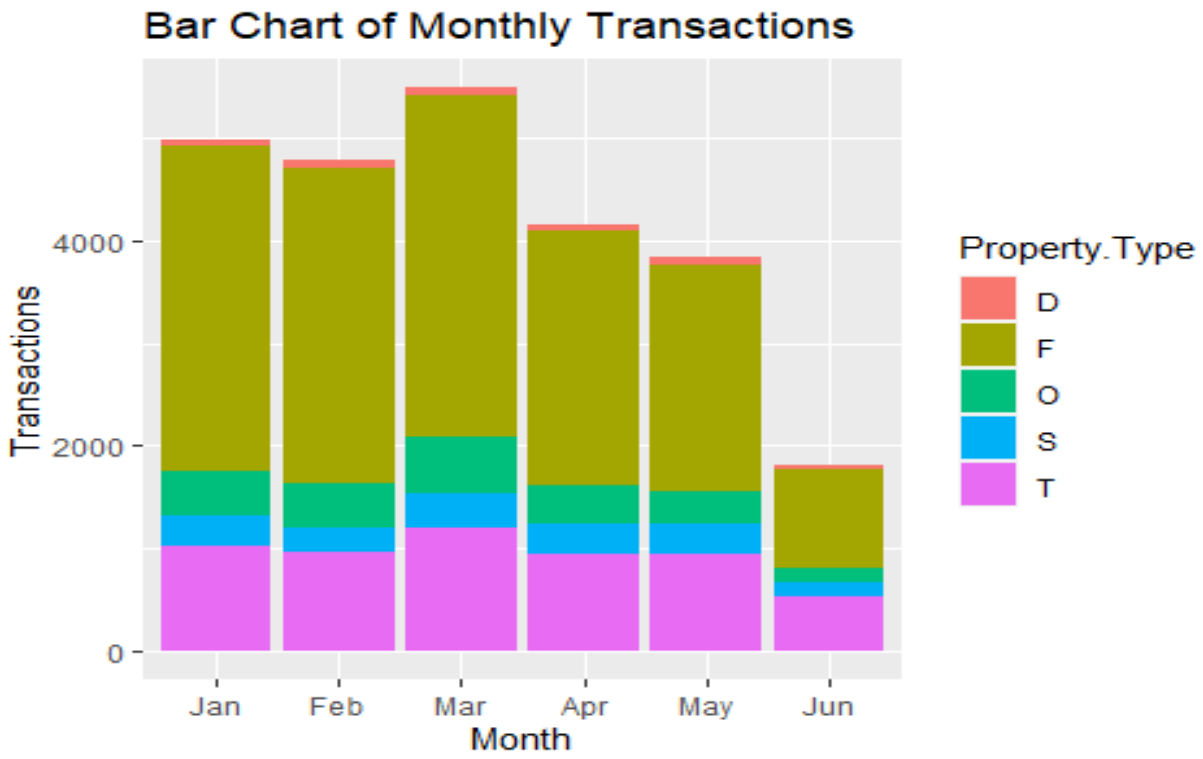
Even though the detached property type has the lowest count of all the property types, it has the largest area in regards to the median price. This means it sold for much more compared to all other property types.

```
# Summarize the data by month
london_monthly <- london_2017_data %>%
  group_by(month = floor_date(Date.of.Transfer, "month"), Property.Type) %>%
  summarise(median_price = median(Price))

## `summarise()` has grouped output by 'month'. You can override using the
## `.groups` argument.

# Plot area chart
ggplot(london_monthly, aes(x = month, y = median_price, fill = Property.Type)
) +
  geom_area() +
  labs(x = "Month", y = "Median Price", title = "London Housing Prices by Pro
perty Type (2017)")
```



As stated earlier, the first three months of the year 2017 sold the most properties. This chart shows that the flat property type has the highest sales for all months, followed by the terraced property type.

```
# Convert Date of transfer to a date format
london_2017_data$Date.of.Transfer <- as.Date(london_2017_data$Date.of.Transfe
r, format = "%Y/%m/%d")

# Create a new column for month
london_2017_data$Month <- month(london_2017_data$Date.of.Transfer, label = TR
UE)

# Count number of transactions per month and property type
monthly_count <- aggregate(london_2017_data$Price, by = list(london_2017_data
$Month, london_2017_data$Property.Type), length)
colnames(monthly_count) <- c("Month", "Property.Type", "Transactions")

# Create bar chart
ggplot(london_2017_data, aes(x = Month, fill = Property.Type))+
  geom_bar()+
  ggtitle("Bar Chart of Monthly Transactions")+
  ylab("Transactions")
```

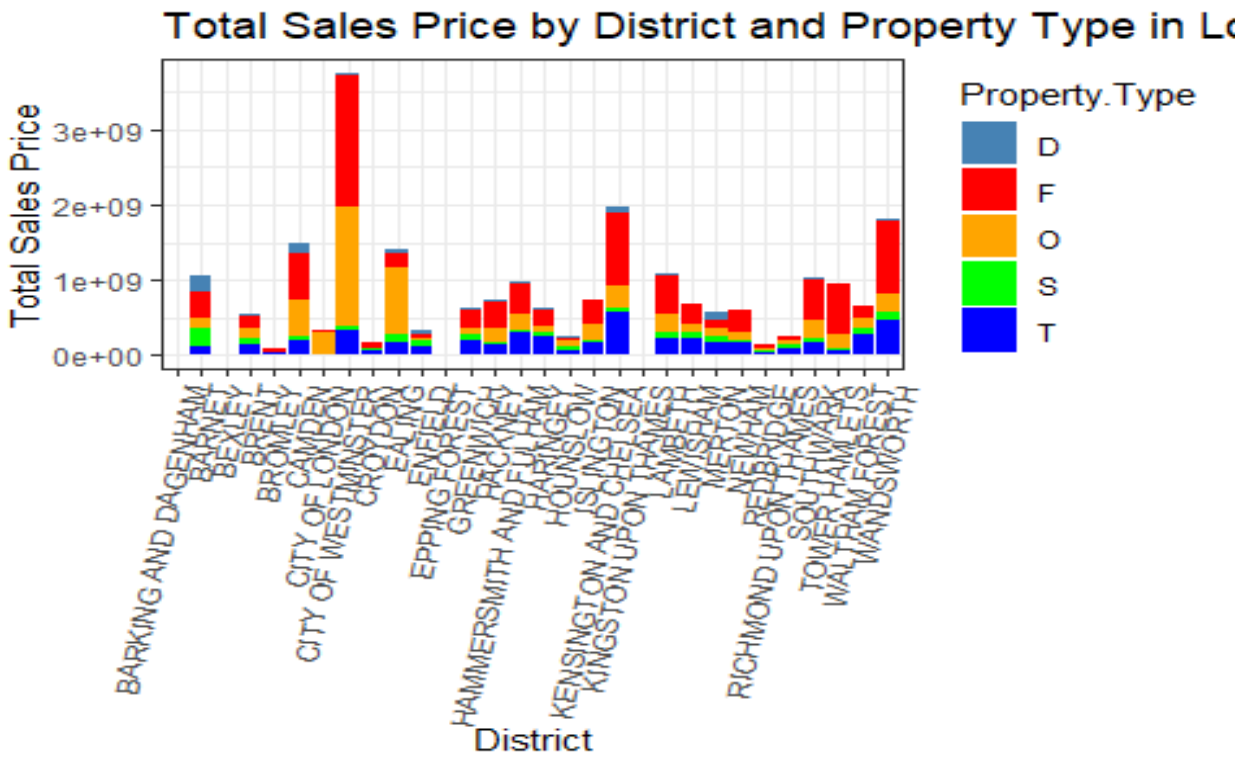


Sales Price by District

The chart in this code chunk shows the cummulative sum of property sales price for each district, with the property type filled. The City of Westminster has the most total sales price compared to all other districts, with Flats selling the most total sales price. Kingston Upon Thames, Epping Forest and Bexley districts have the lowest total sales price in all the districts.

```
# Create a new data frame with total sales price by district and property typ
e
sales_by_district <- london_2017_data %>%
  group_by(District, Property.Type) %>%
  summarize(Total_Price = sum(Price)) %>%
  ungroup()

## `summarise()` has grouped output by 'District'. You can override using the
## `.groups` argument.

# Create a stacked bar chart
ggplot(sales_by_district, aes(x = District, y = Total_Price, fill = Property.
Type)) +
  geom_col(position = "stack") +
  theme_bw() +
  labs(title = "Total Sales Price by District and Property Type in London (20
17)",
       x = "District", y = "Total Sales Price") +
  scale_fill_manual(values = c("steelblue", "red", "orange", "green", "blue",
```

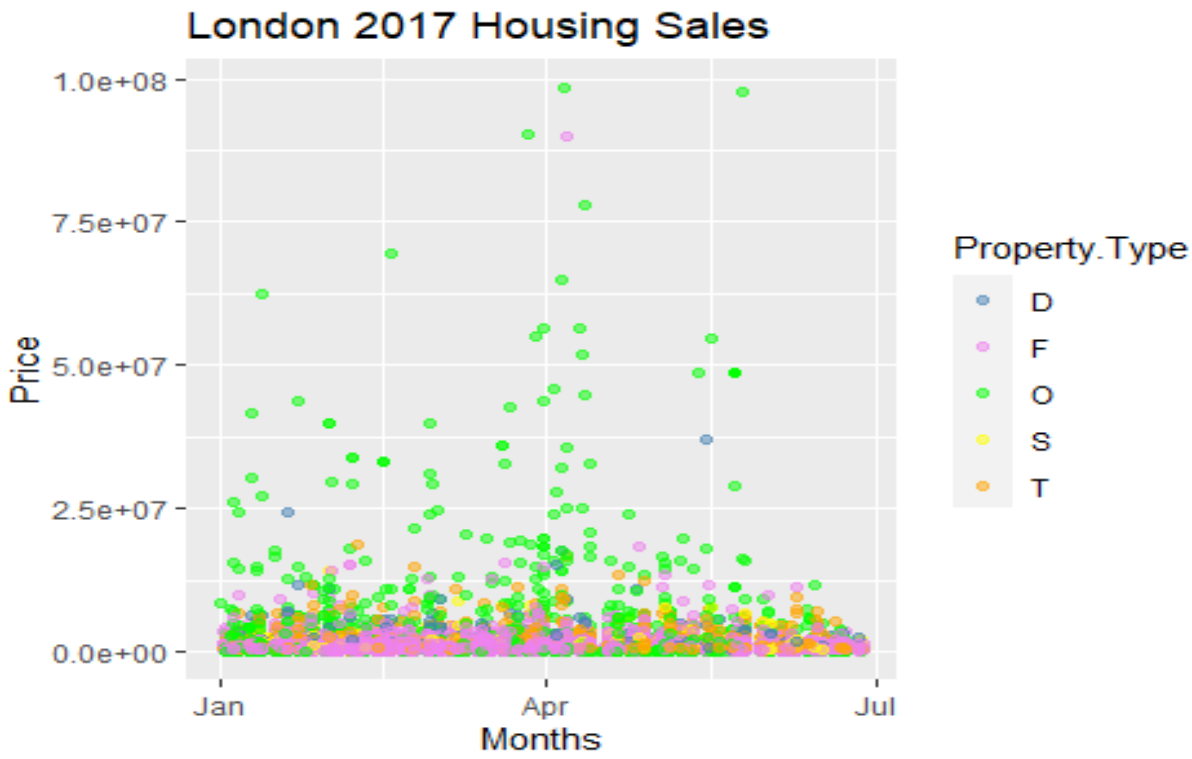
```
"yellow")) +
  theme(axis.text.x = element_text(angle = 80, hjust = 1))
```



**Scatter Plot of Date of Transfer Against Price

The scatter plot in this chunk displays the month of transfer of the different property types against the price of the property. This chart shows that the most expensive properties were transferred at a later date, however, a very large proportion of the cheap properties are clustered in all the months. Cheap properties will always sell, no matter the time of the year. Only one flat is seen in the upper echelon of the plot, with the transfer month coming just after April. The most expensive properties in the dataset is the other type od property.

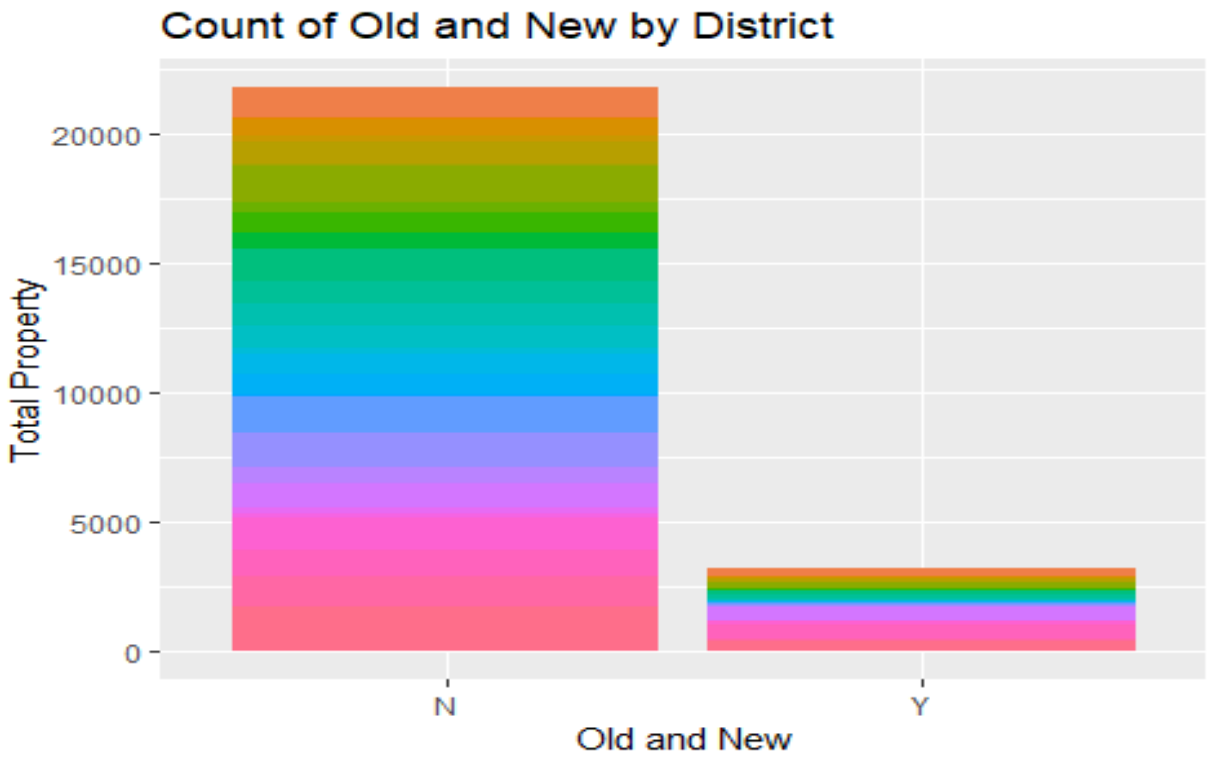
```
ggplot(london_2017_data, aes(x = Date.of.Transfer, y = Price, color = Propert
y.Type)) +
  geom_point(alpha = 0.5) +
  scale_color_manual(values = c("steelblue", "violet", "green", "yellow", "or
ange")) +
  labs(title = "London 2017 Housing Sales", x = "Months", y = "Price")
```



Old and New Properties

In this chart, the Old and New properties is displayed by the district, N represents an established building, while Y is for a newly built building. The legend is set to false, as the chart fails to display when the RMarkdown file is knitted. The next chunk, shows the legend alongside the plot. Truly, it is not a discernible plot, as the districts all but morph into one another.

```
#Bar Chart of Old and New Properties
ggplot(london_2017_data, aes(x = Old.New, fill = District))+
  geom_bar()+
  labs(title = "Count of Old and New by District", x = "Old and New", y = "To
tal Property")+
  guides(fill = FALSE)
```




```
#Bar Chart of Old and New Properties
ggplot(london_2017_data, aes(x = Old.New, fill = District))+
  geom_bar()+
  labs(title = "Count of Old and New by District", x = "Old and New", y = "To
tal Property")
```

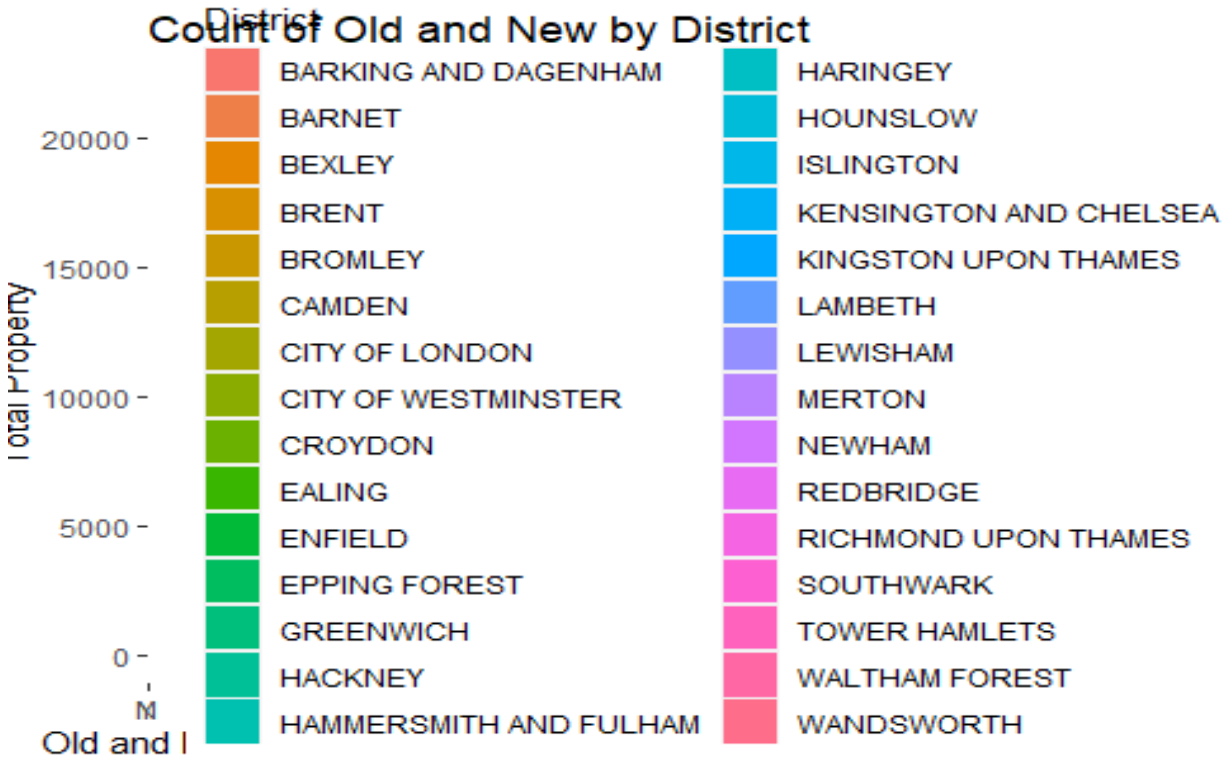
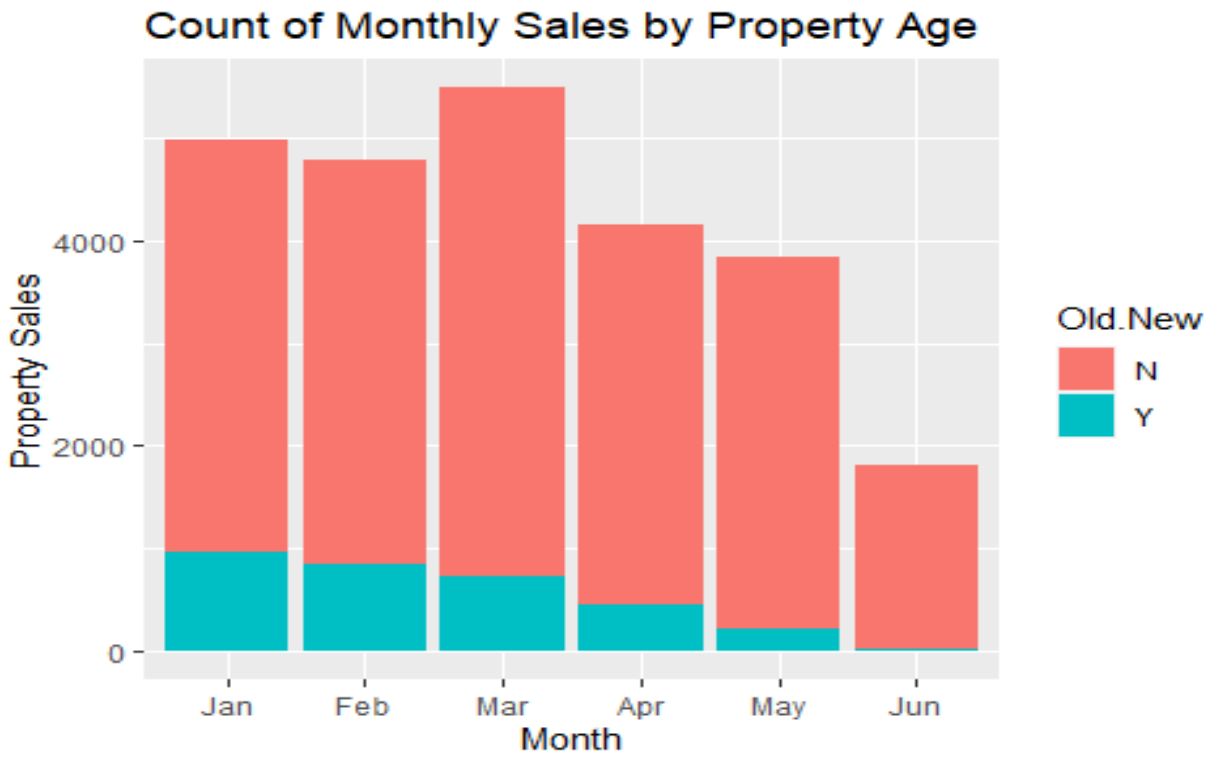


Chart of Monthly Sales by Property Age

The chart in this code chunk displays the monthly sales based on the age of the property, either old or new. Most of the properties in the dataset are old and they cover most of the monthly sales. January sold more new properties compared to the other months, with june having the lowest sales of new properties.

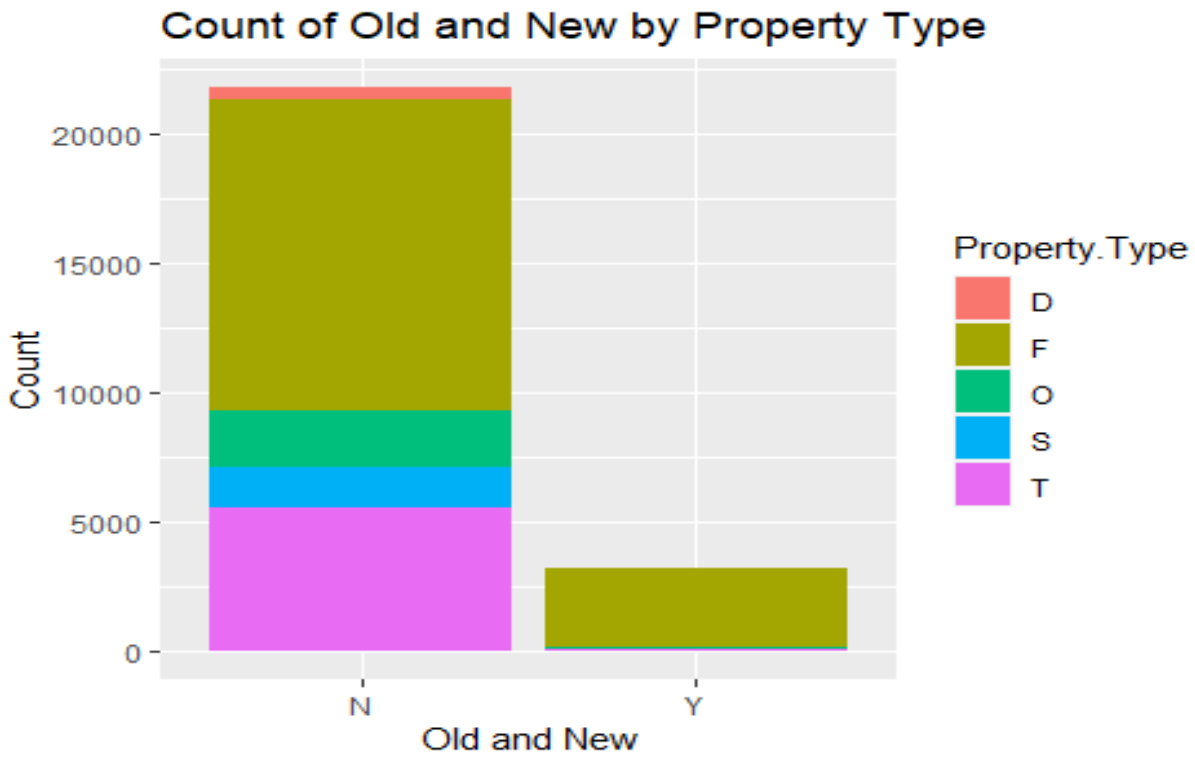
```
#Bar Chart of Monthly Sales by Property Age
ggplot(london_2017_data, aes(x = Month, fill = Old.New))+
  geom_bar()+
  labs(title = "Count of Monthly Sales by Property Age", x = "Month", y = "Pr
operty Sales")
```



Old and New Chart Focused on Property Type

The code in this chunk is focused on the number of property types, noting how many are old and new in the dataset.The Flat property type has the most count in both the new and old column of the dataset.There are more old terraced apartments than semi-detached, other and detached property types.

```
ggplot(london_2017_data, aes(x = Old.New, fill = Property.Type))+
  geom_bar()+
  labs(title = "Count of Old and New by Property Type", x = "Old and New", y
= "Count")
```



London Map

The code in this chunk runs a shapefile of the London wards. The resulting plot is a map of London.

```
# read in the shapefile of London wards
london_wards <- st_read("london_wards_2011_wgs84.shp")
```

```
## Reading layer `london_wards_2011_wgs84' from data source
##   `C:\Users\TifeN\OneDrive\R Studio\london_wards_2011_wgs84.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 649 features and 0 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -0.5103751 ymin: 51.28676 xmax: 0.3340156 ymax: 51.69
187
## CRS:           NA
```

```
# plot the map using tmap
tm_shape(london_wards) +
  tm_polygons() +
  tm_borders(lwd = 0.2)+
  tm_layout(frame = FALSE)
```

```
## Warning: Currect projection of shape london_wards unknown. Long-lat (WGS84
) is
## assumed.
```

```
## Warning: One tm layer group has duplicated layer types, which are omitted.
To
## draw multiple layers of the same type, use multiple layer groups (i.e. spe
cify
## tm_shape prior to each of them).
```

Conclusion

This report provides an insight into the housing situation in London in the year 2017. The number of properties sold per district and the type of property sold. The age of the properties in the dataset is also prominent in the data visualization techniques. There are more flats in the dataset than any other property type. The detached property type has the highest average price of any other property type. The month of March has the most property sales. The first three months also has the highest sales compared to other months in the dataset.The Wandsworth district have the highest number of transactions compared to other districts.The City of Westminster has the highest total sales price among the district. Barking and Dagenham district recorded just one sale in the year 2017.