1, Data Import and Brief Description

For my final project, I used the Melbourne Housing Dataset from Kaggle (https://www.kaggle.com/datasets/dansbecker/melbourne-housing-snapshot). This dataset was created by Tony Pino in 2017, who scraped the data from Domain.com.au. It includes various features such as Address, Type of real estate, Suburb, Method of Sale, Number of Rooms, Price, Real Estate Agent, Date of Sale, and Distance from the Central Business District (CBD).

The detailed values in each column are below:

- Rooms: Number of rooms
- Price: Price in dollars
- Method: S - property sold; SP - property sold prior; PI - property passed in; PN - sold prior not disclosed; SN - sold not disclosed; NB - no bid; VB - vendor bid; W - withdrawn prior to auction; SA - sold after auction; SS - sold after auction price not disclosed. N/A - price or highest bid not available.
- Type: br - bedroom(s); h - house,cottage,villa, semi,terrace; u - unit, duplex; t - townhouse; dev site - development site; o res - other residential.
- SellerG: Real Estate Agent
- Date: Date sold
- Distance: Distance from CBD
- Regionname: General Region (West, North West, North, North east ...etc)
- Propertycount: Number of properties that exist in the suburb.
- Bedroom2 : Scraped # of Bedrooms (from different source)
- Bathroom: Number of Bathrooms
- Car: Number of carspots
- Landsize: Land Size
- BuildingArea: Building Size
- CouncilArea: Governing council for the area

I chose this dataset because Melbourne, Australia, is one of the most popular and rapidly growing cities in the world. As such, the demand for housing is high, and purchasing a property is a significant financial decision that typically requires careful consideration. For newcomers to Melbourne, understanding the real estate market can be overwhelming and time-consuming.

To address these challenges, I decided to explore this dataset in order to identify the key factors that influence housing prices. The goal is to provide useful insights that could help potential buyers make more informed decisions.

2, Source Code for analysis

Used Library

In [1]:
```
library("dplyr")

#install.packages("lubridate")
library(lubridate)
```

```
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


Loading required package: timechange

Warning message in system("timedatectl", intern = TRUE):
“running command 'timedatectl' had status 1”

Attaching package: ‘lubridate’


The following objects are masked from ‘package:base’:

    date, intersect, setdiff, union
```

Load dataset

In [2]:
```r
df = read.csv("melb_data.csv")
head(df)
```

A data.frame: 6 × 21

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	⋯	Bathroom	Car	Landsize	Builc
	<chr>	<chr>	<int>	<chr>	<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>	⋯	<dbl>	<dbl>	<dbl>	
1	Abbotsford	85 Turner St	2	h	1480000	S	Biggin	3/12/2016	2.5	3067	⋯	1	1	202	
2	Abbotsford	25 Bloomburg St	2	h	1035000	S	Biggin	4/02/2016	2.5	3067	⋯	1	0	156	
3	Abbotsford	5 Charles St	3	h	1465000	SP	Biggin	4/03/2017	2.5	3067	⋯	2	0	134	
4	Abbotsford	40 Federation La	3	h	850000	PI	Biggin	4/03/2017	2.5	3067	⋯	2	1	94	
5	Abbotsford	55a Park St	4	h	1600000	VB	Nelson	4/06/2016	2.5	3067	⋯	1	2	120	
6	Abbotsford	129 Charles St	2	h	941000	S	Jellis	7/05/2016	2.5	3067	⋯	1	0	181	

Data Cleaning

- Check all column names
- Remove columns which are unnecessary
- Check and Fill missing values

In [3]:
```r
colnames(df)

# Address, Postcode, Bedroom2, Propertycount, SellerG, Regionname (delete)
# Type, Method, Date, CouncilArea (Categorical Variables)
df = df %>%
    select(-Address, -Postcode, -Bedroom2)

dim(df)
```

'Suburb' · 'Address' · 'Rooms' · 'Type' · 'Price' · 'Method' · 'SellerG' · 'Date' · 'Distance' · 'Postcode' · 'Bedroom2' · 'Bathroom' · 'Car' · 'Landsize' · 'BuildingArea' · 'YearBuilt' · 'CouncilArea' · 'Lattitude' · 'Longtitude' · 'Regionname' · 'Propertycount'

13580 · 18

In [4]:
```r
head(df)
colSums(is.na(df))
```

A data.frame: 6 × 18

	Suburb	Rooms	Type	Price	Method	SellerG	Date	Distance	Bathroom	Car	Landsize	BuildingArea	YearBuilt	Counci
	<chr>	<int>	<chr>	<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	
1	Abbotsford	2	h	1480000	S	Biggin	3/12/2016	2.5	1	1	202	NA	NA	
2	Abbotsford	2	h	1035000	S	Biggin	4/02/2016	2.5	1	0	156	79	1900	
3	Abbotsford	3	h	1465000	SP	Biggin	4/03/2017	2.5	2	0	134	150	1900	
4	Abbotsford	3	h	850000	PI	Biggin	4/03/2017	2.5	2	1	94	NA	NA	
5	Abbotsford	4	h	1600000	VB	Nelson	4/06/2016	2.5	1	2	120	142	2014	
6	Abbotsford	2	h	941000	S	Jellis	7/05/2016	2.5	1	0	181	NA	NA	

Suburb: 0 **Rooms:** 0 **Type:** 0 **Price:** 0 **Method:** 0 **SellerG:** 0 **Date:** 0 **Distance:** 0 **Bathroom:** 0 **Car:** 62 **Landsize:** 0 **BuildingArea:** 6450 **YearBuilt:** 5375 **CouncilArea:** 0 **Lattitude:** 0 **Longtitude:** 0 **Regionname:** 0 **Propertycount:** 0

In [5]:
```
# Fill missing values
df_RNull = df %>%
    mutate(
        Car = ifelse(is.na(Car), median(Car, na.rm=TRUE), Car),
        BuildingArea = ifelse(is.na(BuildingArea), median(BuildingArea, na.rm=TRUE), BuildingArea),
        YearBuilt = ifelse(is.na(YearBuilt), median(YearBuilt, na.rm=TRUE), YearBuilt))

# Delete BuildingArea and YearBuilt because there are so many missing values
df_RNull = df_RNull %>%
    select(-BuildingArea, -YearBuilt)

head(df_RNull)
```

A data.frame: 6 × 16

	Suburb	Rooms	Type	Price	Method	SellerG	Date	Distance	Bathroom	Car	Landsize	CouncilArea	Lattitude	Longtit
	<chr>	<int>	<chr>	<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>	<d
1	Abbotsford	2	h	1480000	S	Biggin	3/12/2016	2.5	1	1	202	Yarra	-37.7996	144.9
2	Abbotsford	2	h	1035000	S	Biggin	4/02/2016	2.5	1	0	156	Yarra	-37.8079	144.9
3	Abbotsford	3	h	1465000	SP	Biggin	4/03/2017	2.5	2	0	134	Yarra	-37.8093	144.9
4	Abbotsford	3	h	850000	PI	Biggin	4/03/2017	2.5	2	1	94	Yarra	-37.7969	144.9
5	Abbotsford	4	h	1600000	VB	Nelson	4/06/2016	2.5	1	2	120	Yarra	-37.8072	144.9
6	Abbotsford	2	h	941000	S	Jellis	7/05/2016	2.5	1	0	181	Yarra	-37.8041	144.9

In [6]:
```
unique(df_RNull$Type)
# h - house,cottage,villa, semi,terrace; u - unit, duplex; t - townhouse
```

'h' · 'u' · 't'

Data Preprocessing

- Remove Outliers for numerical variables
- Split Date to Year, Month, Day
- Apply frequency encoding for categorical variables

In [7]:
```
# Remove outliers
# Checking for outliers
detect_outliers = function(data) {
    Q1 = quantile(data, .25)
    Q3 = quantile(data, .75)
    IQR = Q3 - Q1
    return(data > Q3 + 1.5*IQR | data < Q1 - 1.5*IQR)
    }

# Function for removing outliers
remove_outliers = function(df, col_names) {
```

```
    for (name in col_names) {
        df = df[!detect_outliers(df[[name]]), ]
    }
    return(df)
}

# Extract only numeric variables
df_numeric = df_RNull %>%
    select(where(is.numeric))

num_colnames = colnames(df_numeric)

# Apply created functions
df_RNull = remove_outliers(df_RNull, num_colnames)

# Show the output
head(df_RNull)
```

A data.frame: 6 × 16

	Suburb	Rooms	Type	Price	Method	SellerG	Date	Distance	Bathroom	Car	Landsize	CouncilArea	Lattitude	Longtit
	<chr>	<int>	<chr>	<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>	<d
1	Abbotsford	2	h	1480000	S	Biggin	3/12/2016	2.5	1	1	202	Yarra	-37.7996	144.9
2	Abbotsford	2	h	1035000	S	Biggin	4/02/2016	2.5	1	0	156	Yarra	-37.8079	144.9
3	Abbotsford	3	h	1465000	SP	Biggin	4/03/2017	2.5	2	0	134	Yarra	-37.8093	144.9
4	Abbotsford	3	h	850000	PI	Biggin	4/03/2017	2.5	2	1	94	Yarra	-37.7969	144.9
5	Abbotsford	4	h	1600000	VB	Nelson	4/06/2016	2.5	1	2	120	Yarra	-37.8072	144.9
6	Abbotsford	2	h	941000	S	Jellis	7/05/2016	2.5	1	0	181	Yarra	-37.8041	144.9

In [8]:

```
# Split Date to Year, Month, Day
df_DSplit = df_RNull %>%
    mutate(Date = dmy(Date),
           Year = year(Date),
           Month = month(Date),
           Day = day(Date)
           )

df_DSplit = df_DSplit %>%
    select(-Date)
```

In [9]:

```
# Target Encoding for categorcal variables (Type, Method, SellerG, CouncilArea, Regionname)
# df_encoded_targetE = df_DSplit %>%
#     group_by(Method, Type, Suburb, CouncilArea, SellerG, Regionname) %>%
#     mutate(TEncoded_Method = mean(Price),
#            TEncoded_Type = mean(Price),
#            TEncoded_Suburb = mean(Price),
#            TEncoded_CouncilArea = mean(Price),
#            TEncoded_SellerG = mean(Price),
#            TEncoded_Regionname = mean(Price)) %>%
#     ungroup()

# df_TEncoded = df_encoded_targetE %>%
#     select(-Method, -Type, -Suburb, -CouncilArea, -SellerG, -Regionname)
```

In [10]:

```
# When applying target encoding, the results become NA because of multicollinearity
# So, I use the frequency encoding instead

# Count frequency
# Method
freq_Method <- df_DSplit %>%
    count(Method) %>%
    rename(freqE_Method = n)

# Type (h:6987, u:2641, t:1025)
freq_Type <- df_DSplit %>%
    count(Type) %>%
    rename(freqE_Type = n)

# Suburb
freq_Suburb <- df_DSplit %>%
    count(Suburb) %>%
    rename(freqE_Suburb = n)

# CouncilArea
```

```r
freq_CouncilArea <- df_DSplit %>%
    count(CouncilArea) %>%
    rename(freqE_CouncilArea = n)

# SellerG
freq_SellerG <- df_DSplit %>%
    count(SellerG) %>%
    rename(freqE_SellerG = n)

# Regionname
freq_Regionname <- df_DSplit %>%
    count(Regionname) %>%
    rename(freqE_Regionname = n)

# Combine dataframe
df_encoded_freqE <- df_DSplit %>%
    left_join(freq_Method, by = "Method") %>%
    left_join(freq_Type, by = "Type") %>%
    left_join(freq_Suburb, by = "Suburb") %>%
    left_join(freq_CouncilArea, by = "CouncilArea") %>%
    left_join(freq_SellerG, by = "SellerG") %>%
    left_join(freq_Regionname, by = "Regionname")

# Delete columns that are not necessary
df_processed = df_encoded_freqE %>%
    select(-Method, -Type, -Suburb, -CouncilArea, -SellerG, -Regionname)

# Show the result
head(df_processed)
```

A data.frame: 6 × 18

	Rooms	Price	Distance	Bathroom	Car	Landsize	Lattitude	Longtitude	Propertycount	Year	Month	Day	freqE_Method	fre
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<int>	
1	2	1480000	2.5	1	1	202	-37.7996	144.9984	4019	2016	12	3	7125	
2	2	1035000	2.5	1	0	156	-37.8079	144.9934	4019	2016	2	4	7125	
3	3	1465000	2.5	2	0	134	-37.8093	144.9944	4019	2017	3	4	1350	
4	3	850000	2.5	2	1	94	-37.7969	144.9969	4019	2017	3	4	1190	
5	4	1600000	2.5	1	2	120	-37.8072	144.9941	4019	2016	6	4	916	
6	2	941000	2.5	1	0	181	-37.8041	144.9953	4019	2016	5	7	7125	

In [11]:

```r
# Check the basic info
summary(df_processed)
dim(df_processed)
```

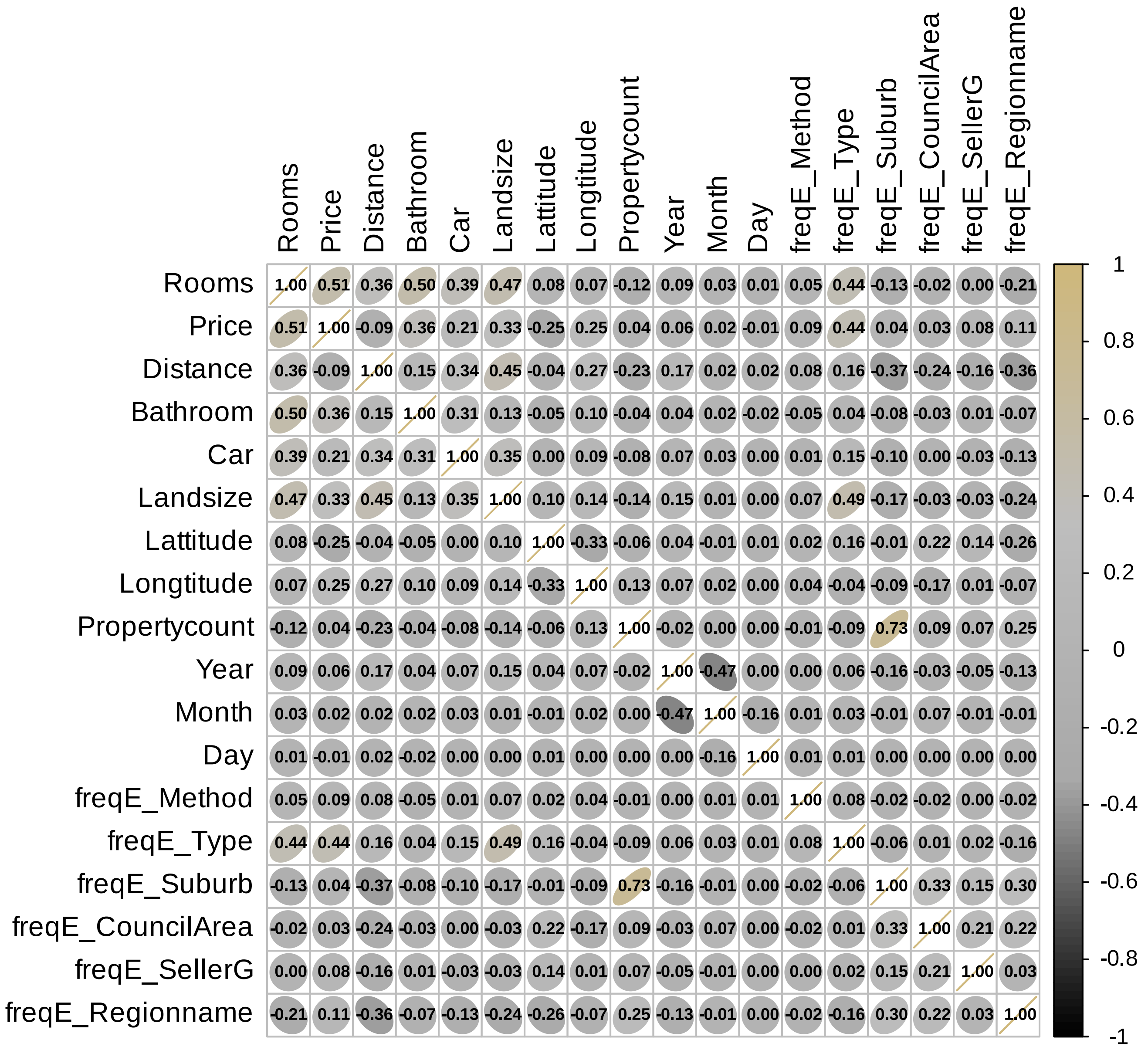
```
     Rooms           Price             Distance          Bathroom    
 Min.   :1.000   Min.   :  85000   Min.   : 0.000   Min.   :0.000  
 1st Qu.:2.000   1st Qu.: 650000   1st Qu.: 5.900   1st Qu.:1.000  
 Median :3.000   Median : 900000   Median : 8.800   Median :1.000  
 Mean   :2.758   Mean   : 985206   Mean   : 9.161   Mean   :1.426  
 3rd Qu.:3.000   3rd Qu.:1271000   3rd Qu.:12.400   3rd Qu.:2.000  
 Max.   :4.000   Max.   :2291000   Max.   :22.700   Max.   :3.000  
      Car           Landsize         Lattitude        Longtitude   
 Min.   :0.00   Min.   :   0.0   Min.   :-38.00   Min.   :144.7  
 1st Qu.:1.00   1st Qu.: 138.0   1st Qu.:-37.85   1st Qu.:144.9  
 Median :1.00   Median : 325.0   Median :-37.80   Median :145.0  
 Mean   :1.42   Mean   : 366.5   Mean   :-37.81   Mean   :145.0  
 3rd Qu.:2.00   3rd Qu.: 599.0   3rd Qu.:-37.76   3rd Qu.:145.1  
 Max.   :3.00   Max.   :1307.0   Max.   :-37.61   Max.   :145.2  
 Propertycount        Year          Month             Day         freqE_Method 
 Min.   :  389   Min.   :2016   Min.   : 1.000   Min.   : 1.00   Min.   :  72  
 1st Qu.: 4181   1st Qu.:2016   1st Qu.: 5.000   1st Qu.: 8.00   1st Qu.:1350  
 Median : 6388   Median :2016   Median : 7.000   Median :16.00   Median :7125  
 Mean   : 6953   Mean   :2016   Mean   : 7.041   Mean   :16.09   Mean   :5149  
 3rd Qu.: 9028   3rd Qu.:2017   3rd Qu.: 9.000   3rd Qu.:24.00   3rd Qu.:7125  
 Max.   :17496   Max.   :2017   Max.   :12.000   Max.   :30.00   Max.   :7125  
   freqE_Type    freqE_Suburb    freqE_CouncilArea freqE_SellerG
 Min.   :1025   Min.   :  1.00   Min.   :   1.0    Min.   :   1  
 1st Qu.:2641   1st Qu.: 49.00   1st Qu.: 403.0    1st Qu.: 134  
 Median :6987   Median : 86.00   Median : 582.0    Median : 427  
 Mean   :5336   Mean   : 96.41   Mean   : 636.9    Mean   : 544  
 3rd Qu.:6987   3rd Qu.:137.00   3rd Qu.: 898.0    3rd Qu.: 930  
 Max.   :6987   Max.   :230.00   Max.   :1051.0    Max.   :1350  
 freqE_Regionname
 Min.   :   6    
 1st Qu.:2481    
 Median :3117    
 Mean   :2927    
 3rd Qu.:3748    
 Max.   :3748    
```

10653 · 18

In [12]:
```
# Missing values check
which(is.na(df_processed))
```

In [13]:
```
# Correlation
install.packages("corrplot")
library(corrplot)
col4 = colorRampPalette(c("black", "darkgrey", "grey","#CFB87C"))
corrplot(cor(df_processed), method = "ellipse", col = col4(100), number.cex=0.6, pch.cex=1.5, addCoef.col = "black", tl.col =
```

```
Updating HTML index of packages in '.Library'

Making 'packages.html' ...
 done

corrplot 0.95 loaded
```

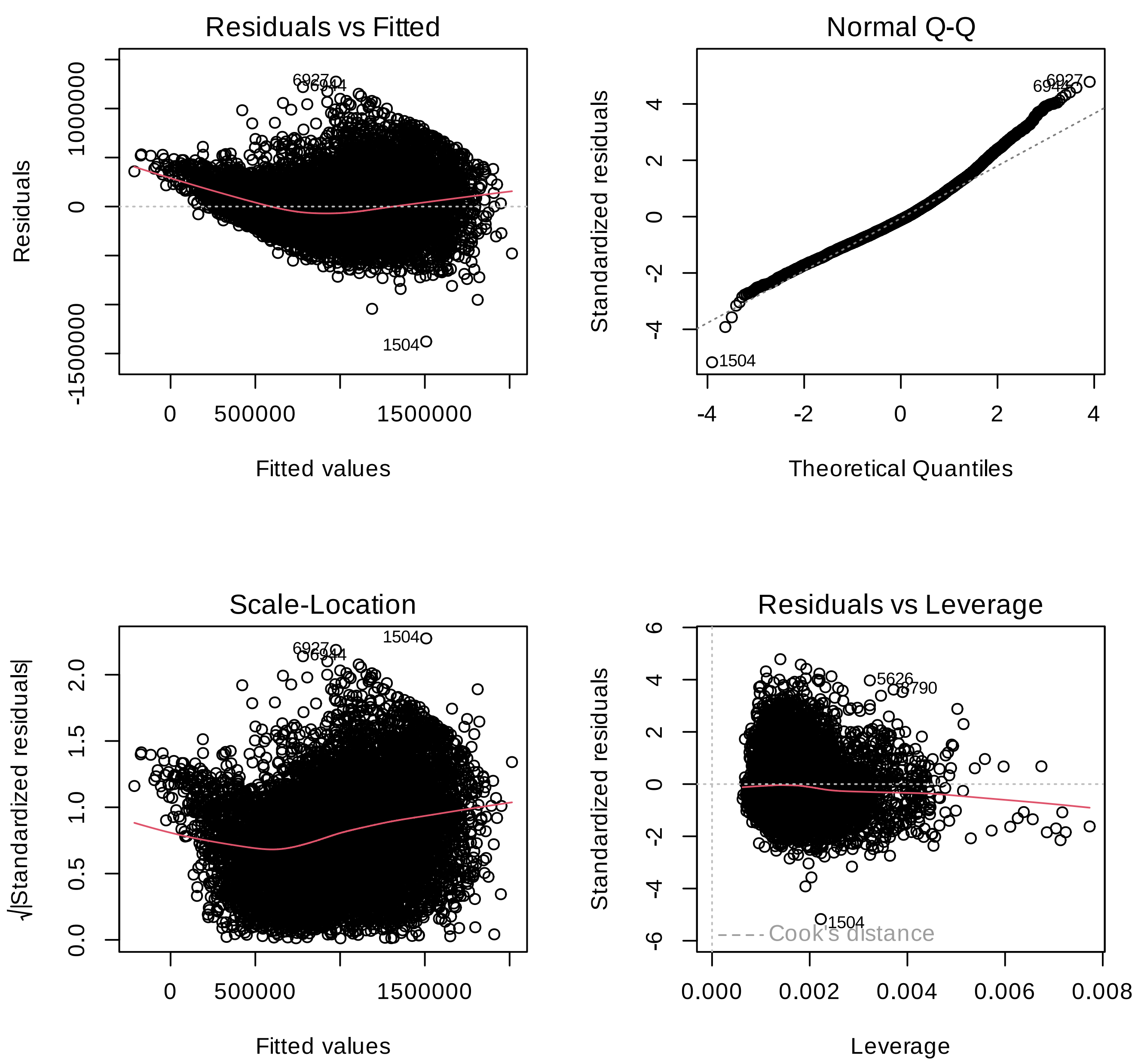


Data Analysis

In [14]:
```
# Multiple Linear Regression
lm_full = lm(Price ~., data=df_processed)
summary(lm_full)

# Day, freqE_Suburb, and freqE_CouncilArea are not significant as p-values are greater than 0.05
# From the full-F test, the p-value is smaller than 0.05, so we reject the null hypothesis that all coefficients are 0.
# Also, F stat is 1010 and quite large, so we can say that the model is very strongly significant as a whole

# So, Delete Day, freqE_Suburb, and freqE_CouncilArea variables
df_processed = df_processed %>%
    select(-Day, -freqE_Suburb, -freqE_CouncilArea)
```

```
Call:
lm(formula = Price ~ ., data = df_processed)

Residuals:
     Min       1Q   Median       3Q      Max 
-1376902  -181613   -29607   152367  1274507 

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)    
(Intercept)       -3.714e+08  1.356e+07 -27.385  < 2e-16 ***
Rooms              1.985e+05  4.609e+03  43.077  < 2e-16 ***
Distance          -4.400e+04  7.873e+02 -55.890  < 2e-16 ***
Bathroom           1.263e+05  5.457e+03  23.151  < 2e-16 ***
Car                2.720e+04  4.112e+03   6.615 3.90e-11 ***
Landsize           2.573e+02  1.251e+01  20.573  < 2e-16 ***
Lattitude         -1.721e+06  4.451e+04 -38.659  < 2e-16 ***
Longtitude         1.171e+06  3.483e+04  33.620  < 2e-16 ***
Propertycount     -4.670e+00  1.153e+00  -4.048 5.20e-05 ***
Year               6.771e+04  6.280e+03  10.782  < 2e-16 ***
Month              4.872e+03  1.185e+03   4.113 3.94e-05 ***
Day               -2.669e+01  3.096e+02  -0.086    0.931    
freqE_Method       1.097e+01  9.272e-01  11.833  < 2e-16 ***
freqE_Type         5.736e+01  1.381e+00  41.532  < 2e-16 ***
freqE_Suburb       9.158e+01  7.513e+01   1.219    0.223    
freqE_CouncilArea  1.653e+01  1.178e+01   1.403    0.161    
freqE_SellerG      3.911e+01  5.890e+00   6.640 3.28e-11 ***
freqE_Regionname   4.539e+01  3.464e+00  13.103  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 266600 on 10635 degrees of freedom
Multiple R-squared:  0.6314,	Adjusted R-squared:  0.6308 
F-statistic:  1072 on 17 and 10635 DF,  p-value: < 2.2e-16
```

In [15]:
```
# Checking the residuals
par(mfrow=c(2,2))
plot(lm_full)
```



Residuals vs Fitted:

- Looking at the fitted vs residuals plot, we can see that the variance is non-constant. Because there is no "rectangular" shape, and it looks like corn shaped, whcih suggess heteroscdasticity.

Normal QQ: Normal: Right skewed

- The upper tail does not follow the line. So, we can say that it is skewed

Therefore, I'll apply the log transformation to response variable(Price)

In [16]:
```
# Log transformation
df_processed$Price = log(df_processed$Price)
```

In [17]:
```
# Multi-Linearity (VIF)
source("vif_function.r")
vif(lm_full)

# All of values are below 5, which means there is no multicollinearity
```

Rooms: 2.07277984342271 **Distance:** 1.75638499018834 **Bathroom:** 1.48614860817829 **Car:** 1.32232378151706 **Landsize:** 1.80491713078198 **Lattitude:** 1.38560894892908 **Longtitude:** 1.37418596636122 **Propertycount:** 2.68370499611002 **Year:** 1.47736526878305 **Month:** 1.41095648719057 **Day:** 1.03651221496041 **freqE_Method:** 1.01854163751325 **freqE_Type:** 1.53685065225366 **freqE_Suburb:** 3.08356636297773 **freqE_CouncilArea:** 1.39895754457178 **freqE_SellerG:** 1.09957365608372 **freqE_Regionname:** 1.39864712662766

In [18]:
```
# Split to train and test datasets
set.seed(11111)
n = floor(0.8 * nrow(df_processed)) #find the number corresponding to 80% of the data
index = sample(seq_len(nrow(df)), size = n) #randomly sample indicies to be included in the training set

train = df_processed[index, ] #set the training set to be the randomly sampled rows of the dataframe
test = df_processed[-index, ] #set the testing set to be the remaining rows
cat("There are", dim(train)[1], "rows and",dim(train)[2],"columns in the training set. ")  #check the dimensions
cat("There are", dim(test)[1], "rows and",dim(test)[2],"columns in the testing set.")  #check the dimensions
```

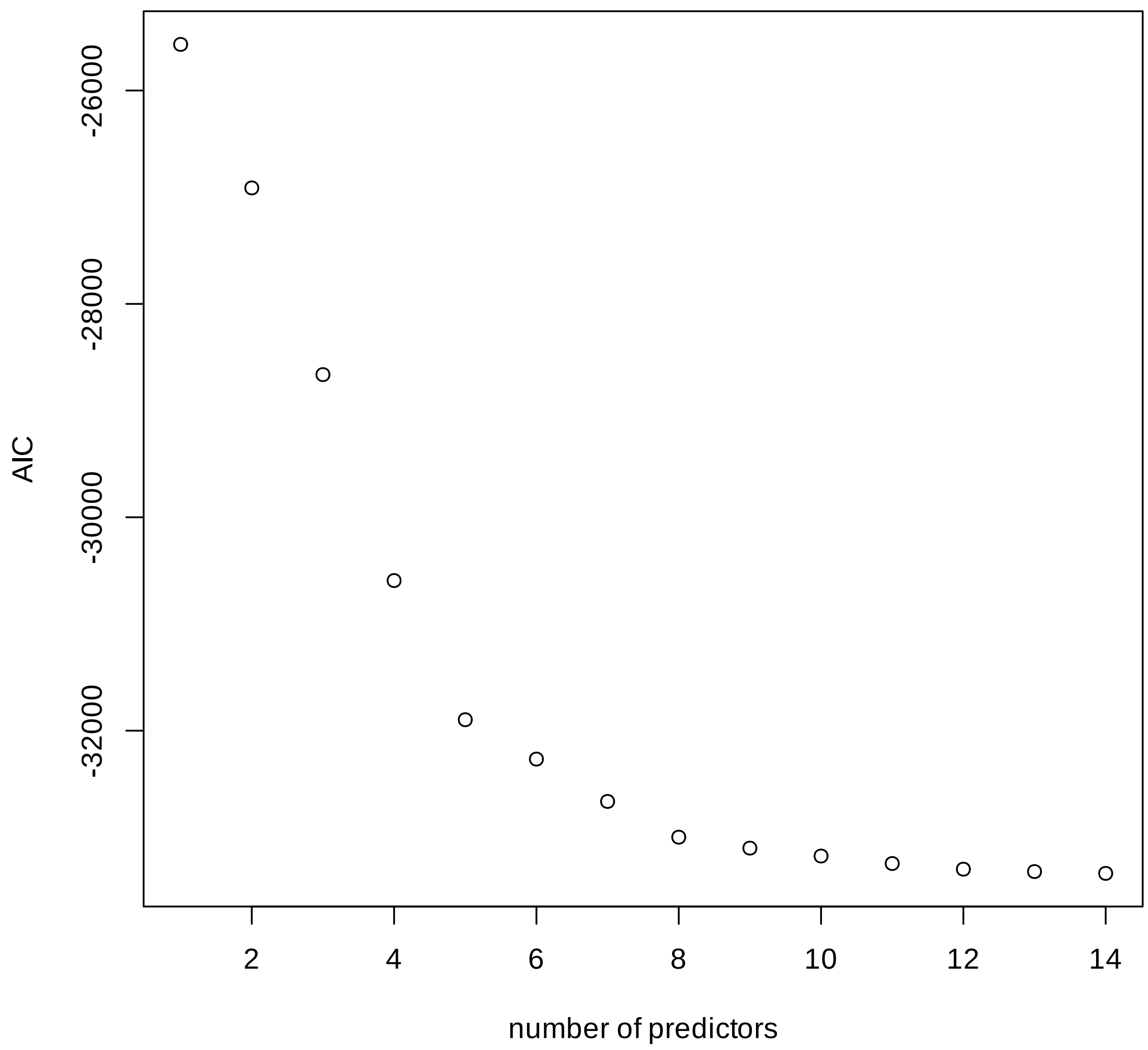
```
There are 8522 rows and 15 columns in the training set. There are 3986 rows and 15 columns in the testing set.
```

In [19]:
```
# Select predictors based on AIC, BIC, and R^2
install.packages("leaps")
library(leaps)

n = dim(df_processed)[1];
#reg1 = regsubsets(Price ~ ., data = df_processed, nvmax = 18)
reg1 = regsubsets(Price ~ ., data = train, nvmax = 15)
rs = summary(reg1)
rs$which

# AIC
AIC = 2*(2:15) + n*log(rs$rss/n)
plot(AIC ~ I(1:14), xlab = "number of predictors", ylab = "AIC")
```

```
Updating HTML index of packages in '.Library'

Making 'packages.html' ...
 done
```

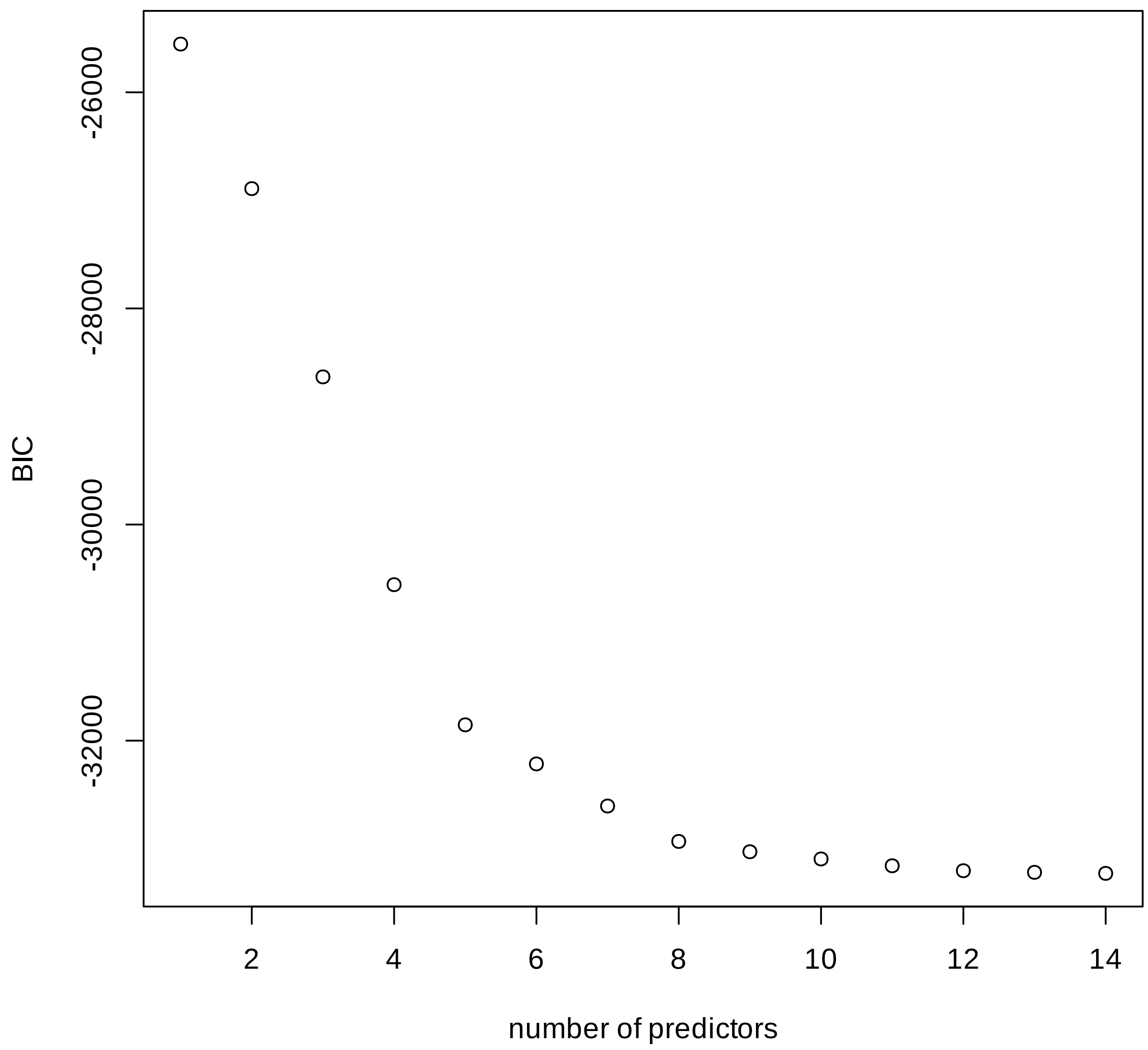
A matrix: 14 × 15 of type lgl

	(Intercept)	Rooms	Distance	Bathroom	Car	Landsize	Lattitude	Longtitude	Propertycount	Year	Month	freqE_Method	freqE_Ty
1	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAI
2	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAI
3	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FAI
4	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	TR
5	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TR
6	TRUE	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TR
7	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TR
8	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	TR
9	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	TR
10	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	TR
11	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	TR
12	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	TR
13	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TR
14	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TR

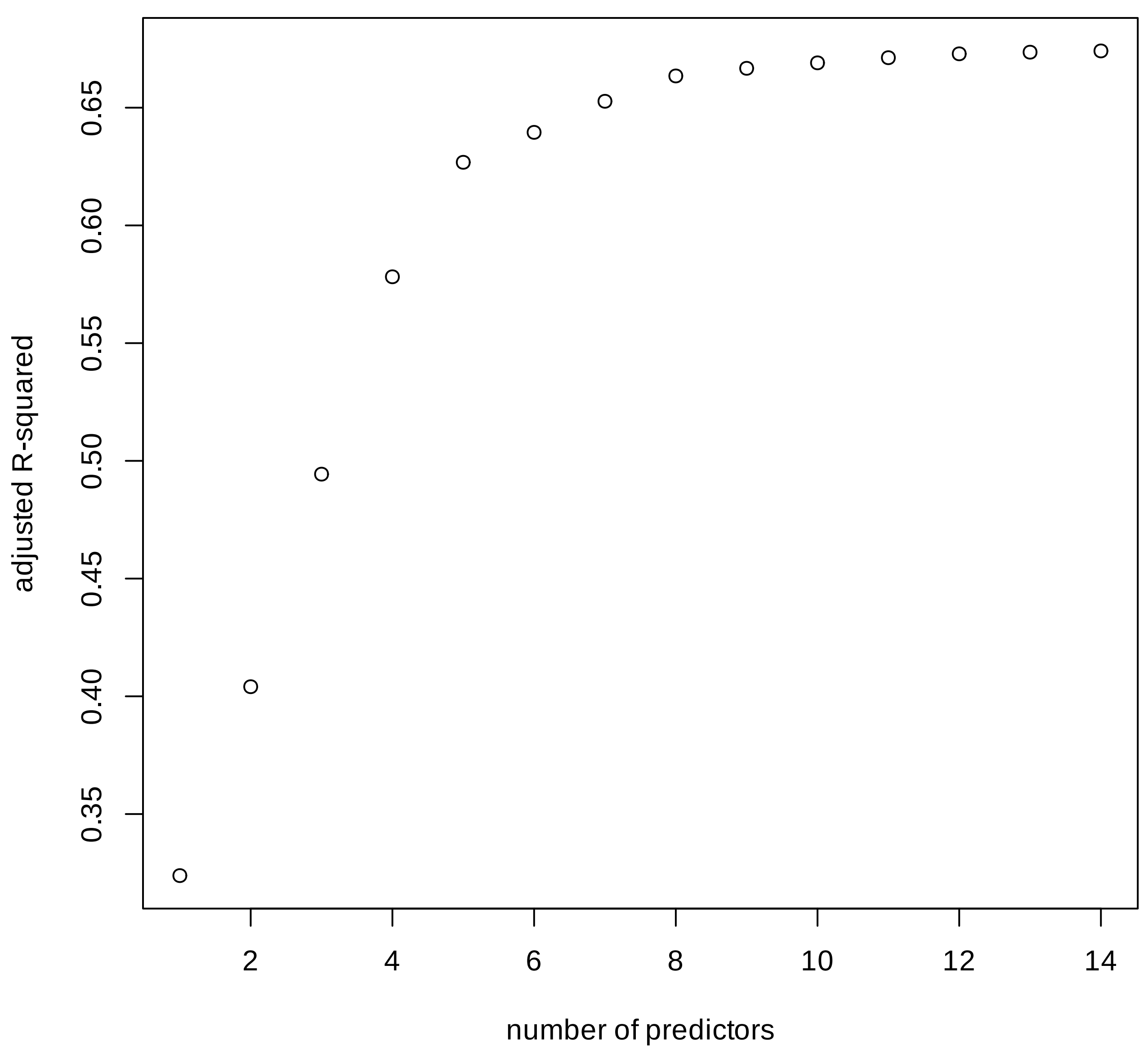


In [20]:
```
#Length(rs$adjr2)
```

In [21]:
```
# BIC
BIC = log(n)*(2:15) + n*log(rs$rss/n)
plot(BIC ~ I(1:14), xlab = "number of predictors", ylab = "BIC")
```



In [22]:
```
# R^2
plot(1:14, rs$adjr2, xlab = "number of predictors", ylab = "adjusted R-squared")

# We'll use all predictors from the results of AIC, BIC, and R^2
```



In [23]:
```
# Modeling (Generalized Linear)
glm_model = glm(Price ~ ., data=train, family=gaussian())
summary(glm_model)
```

```
Call:
glm(formula = Price ~ ., family = gaussian(), data = train)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.05202  -0.17422  -0.00123   0.17295   1.07429

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)      -3.757e+02  1.621e+01 -23.182  < 2e-16 ***
Rooms             2.505e-01  5.776e-03  43.367  < 2e-16 ***
Distance         -4.650e-02  9.677e-04 -48.048  < 2e-16 ***
Bathroom          1.089e-01  6.866e-03  15.860  < 2e-16 ***
Car               3.438e-02  5.180e-03   6.637 3.45e-11 ***
Landsize          2.350e-04  1.573e-05  14.939  < 2e-16 ***
Lattitude        -1.789e+00  5.391e-02 -33.180  < 2e-16 ***
Longtitude        1.182e+00  4.297e-02  27.504  < 2e-16 ***
Propertycount    -3.246e-06  9.420e-07  -3.446 0.000573 ***
Year              7.408e-02  7.586e-03   9.765  < 2e-16 ***
Month             5.708e-03  1.432e-03   3.986 6.78e-05 ***
freqE_Method      1.732e-05  1.163e-06  14.899  < 2e-16 ***
freqE_Type        6.046e-05  1.723e-06  35.082  < 2e-16 ***
freqE_SellerG     4.987e-05  7.276e-06   6.853 7.87e-12 ***
freqE_Regionname  2.735e-05  4.200e-06   6.513 7.90e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.06985868)

    Null deviance: 1428.9  on 6666  degrees of freedom
Residual deviance:  464.7  on 6652  degrees of freedom
  (1855 observations deleted due to missingness)
AIC: 1194.3

Number of Fisher Scoring iterations: 2
```

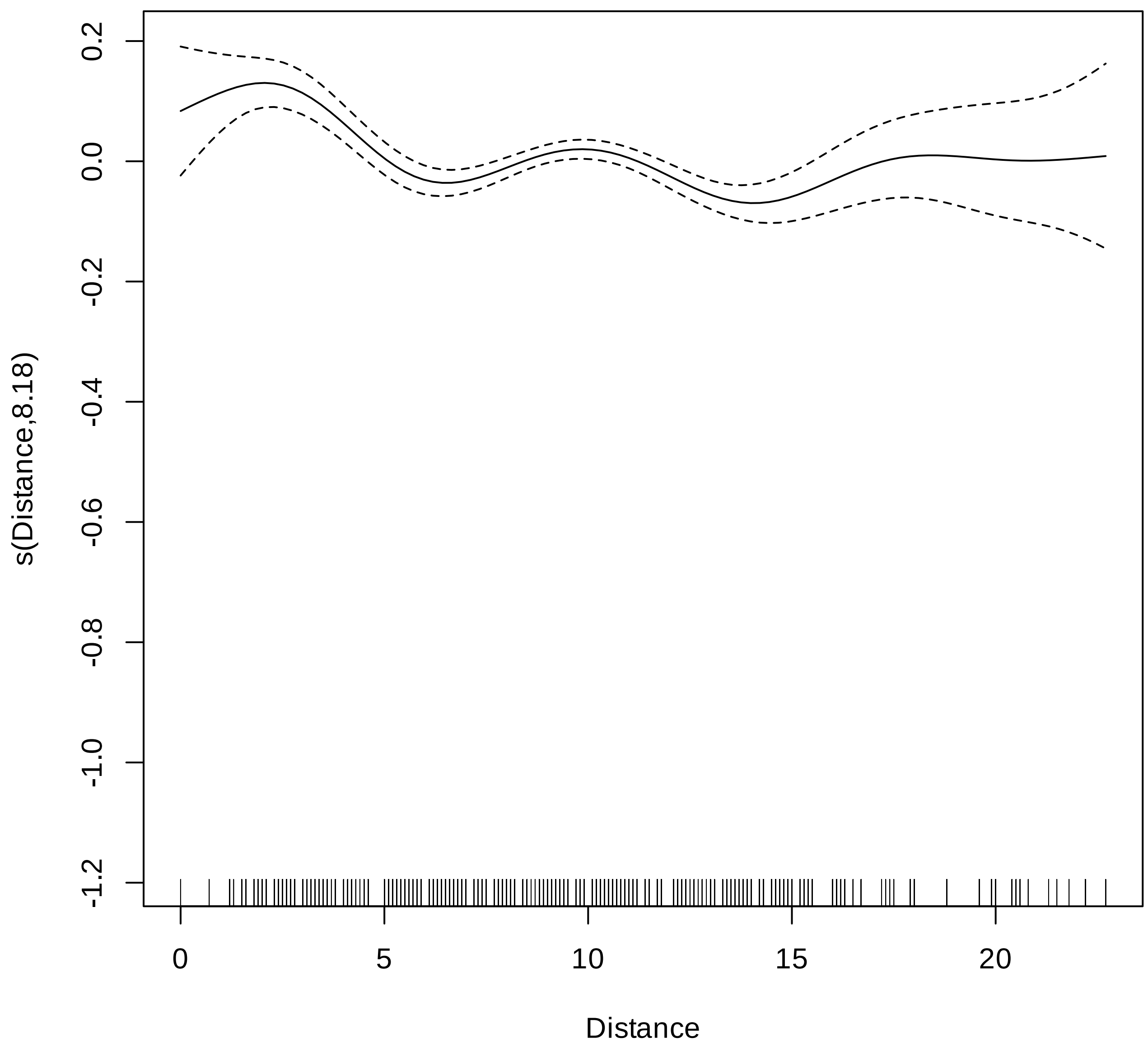
In [24]:
```
# Modeling (GAMs)
library(mgcv)
```

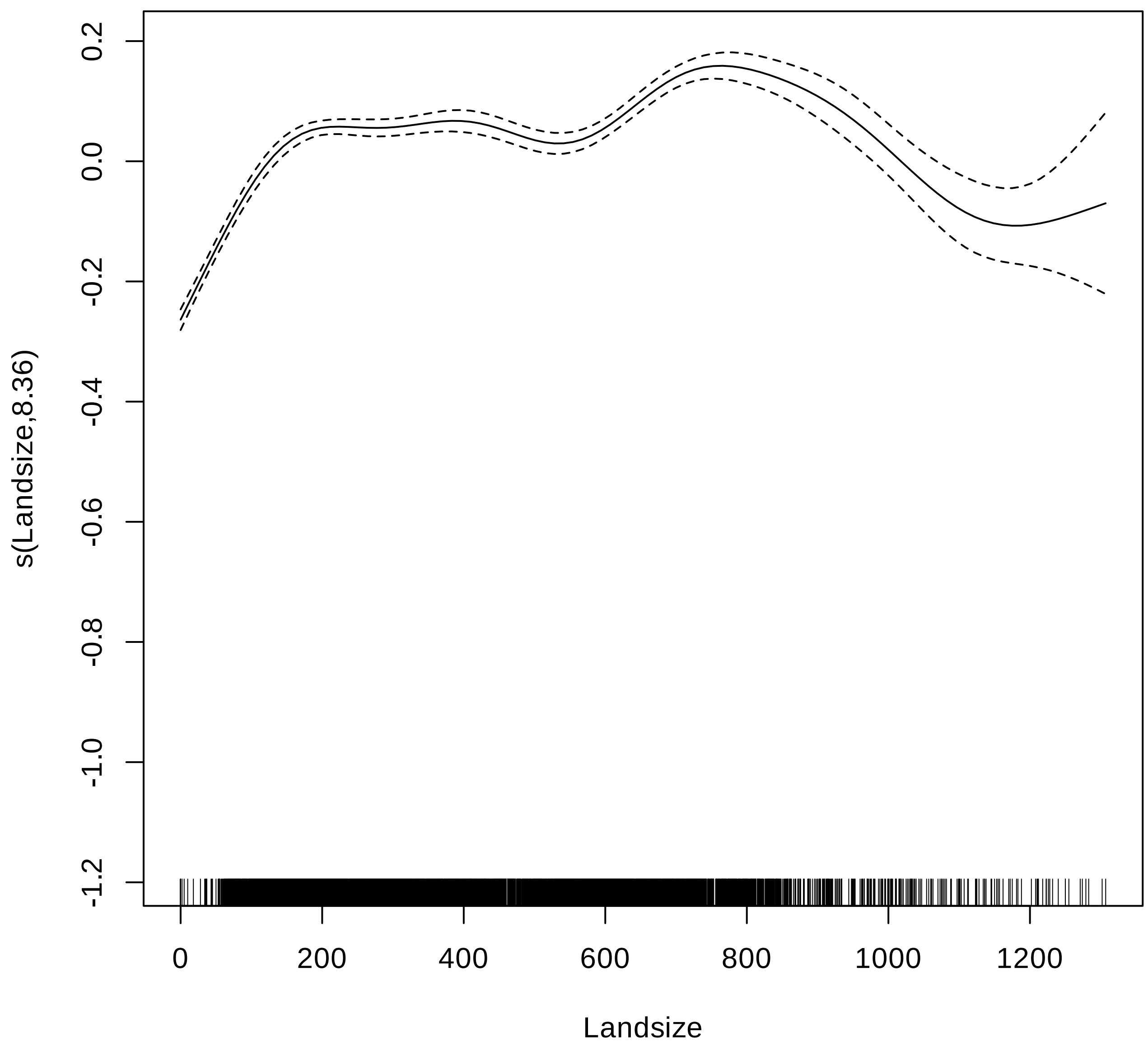
```
gamod = gam(Price ~ Rooms + s(Distance) + Bathroom + Car + s(Landsize) + s(Lattitude) + s(Longtitude) + s(Propertycount)
            + Year + s(Month) + freqE_Method + freqE_Type + freqE_SellerG + freqE_Regionname
           , data=train, family=gaussian())

summary(gamod)
```

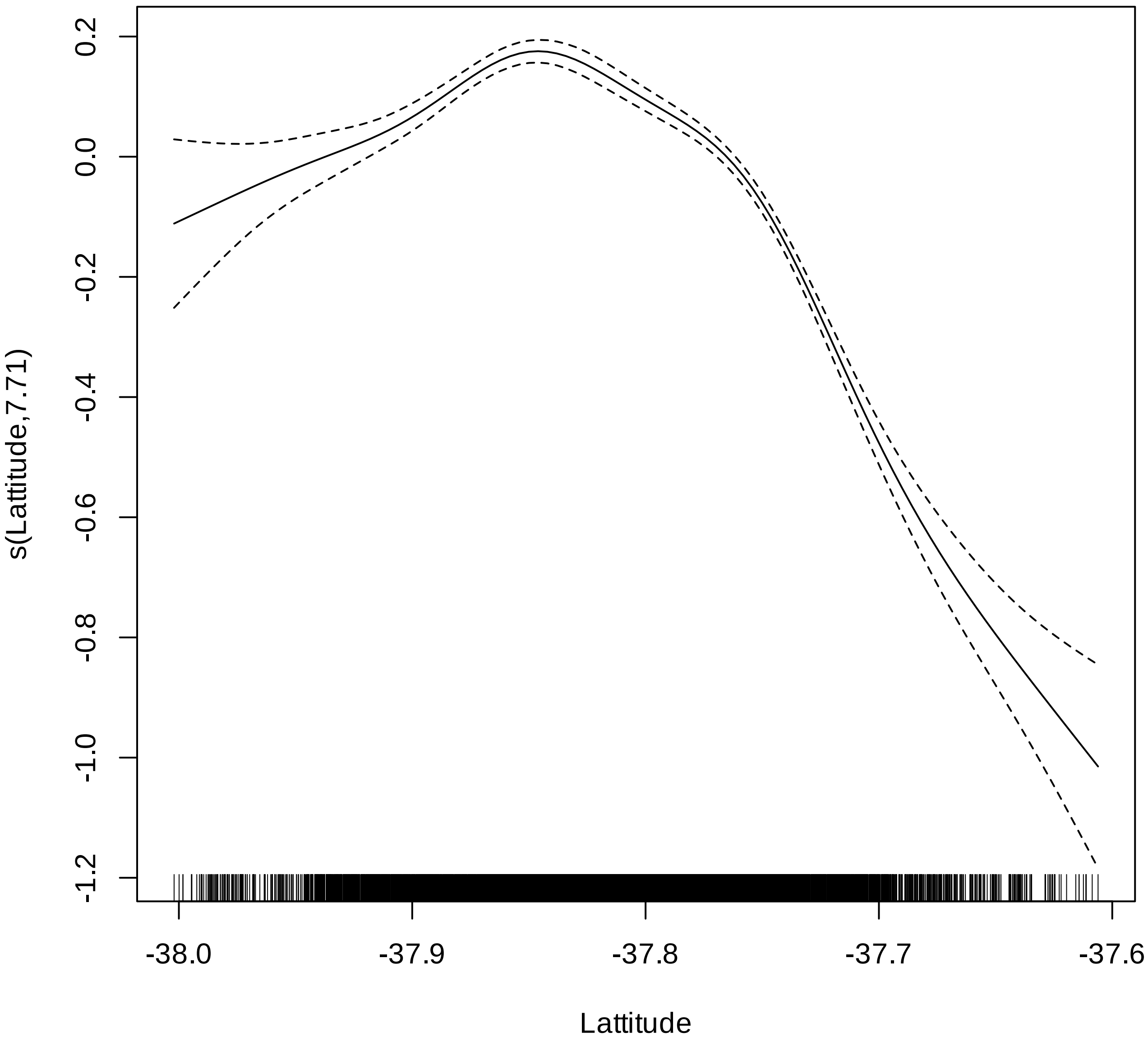
```
Loading required package: nlme


Attaching package: ‘nlme’


The following object is masked from ‘package:dplyr’:

    collapse


This is mgcv 1.8-41. For overview type 'help("mgcv-package")'.
```

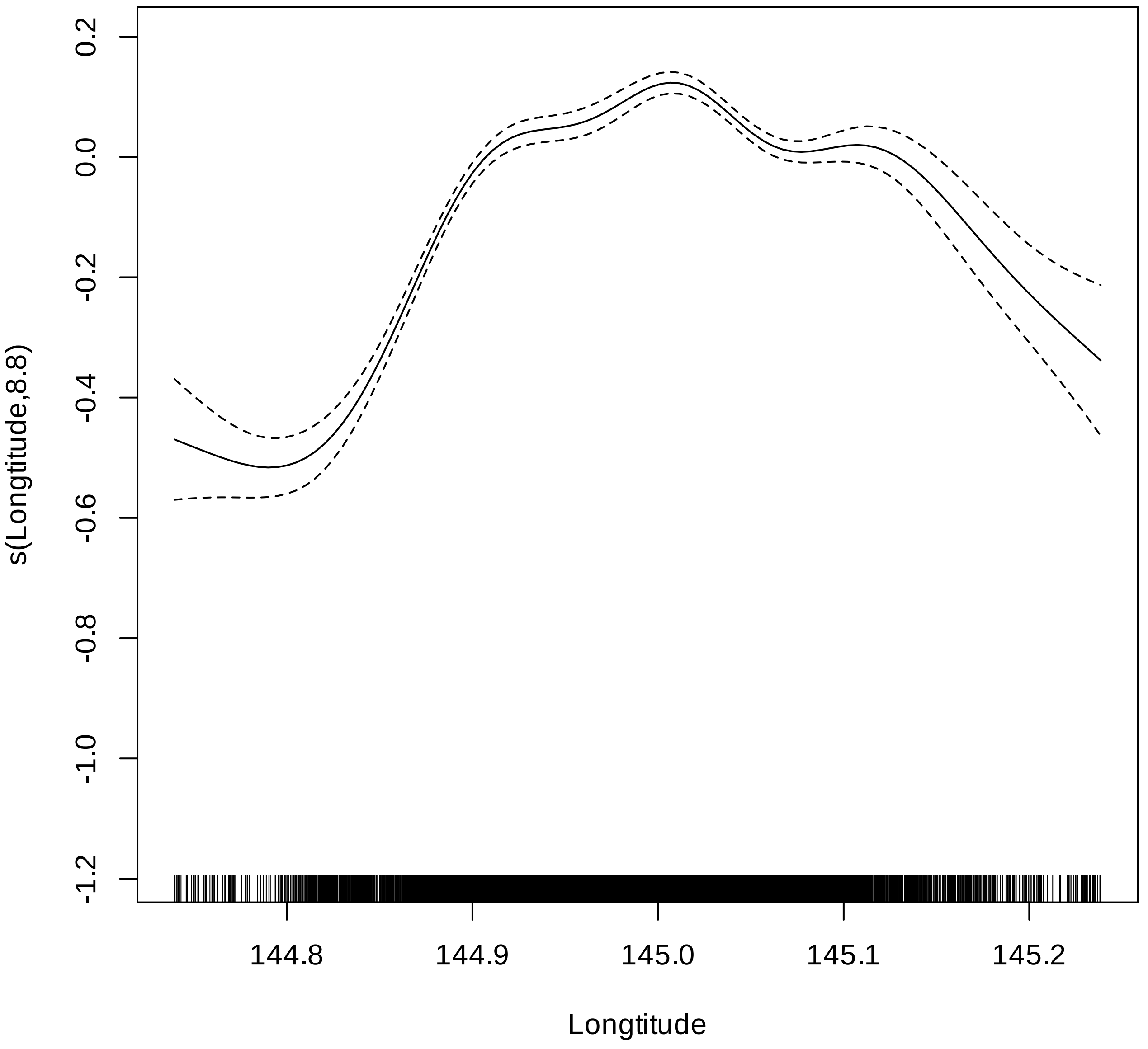
```
Family: gaussian
Link function: identity

Formula:
Price ~ Rooms + s(Distance) + Bathroom + Car + s(Landsize) +
    s(Lattitude) + s(Longtitude) + s(Propertycount) + Year +
    s(Month) + freqE_Method + freqE_Type + freqE_SellerG + freqE_Regionname

Parametric coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)      -1.876e+02  1.514e+01 -12.395  < 2e-16 ***
Rooms             2.240e-01  5.402e-03  41.461  < 2e-16 ***
Bathroom          9.638e-02  6.333e-03  15.220  < 2e-16 ***
Car               4.392e-02  4.815e-03   9.122  < 2e-16 ***
Year              9.921e-02  7.507e-03  13.217  < 2e-16 ***
freqE_Method      1.456e-05  1.061e-06  13.721  < 2e-16 ***
freqE_Type        4.907e-05  1.697e-06  28.910  < 2e-16 ***
freqE_SellerG     3.303e-05  6.909e-06   4.780 1.79e-06 ***
freqE_Regionname  3.472e-05  5.614e-06   6.185 6.60e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
                   edf Ref.df      F p-value
s(Distance)      8.182  8.789  14.60  <2e-16 ***
s(Landsize)      8.364  8.856 123.96  <2e-16 ***
s(Lattitude)     7.711  8.600 177.85  <2e-16 ***
s(Longtitude)    8.800  8.987 124.17  <2e-16 ***
s(Propertycount) 8.594  8.941  15.93  <2e-16 ***
s(Month)         1.149  1.283  28.54  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.733   Deviance explained = 73.5%
GCV = 0.057664  Scale est. = 0.057216   n = 6667
```

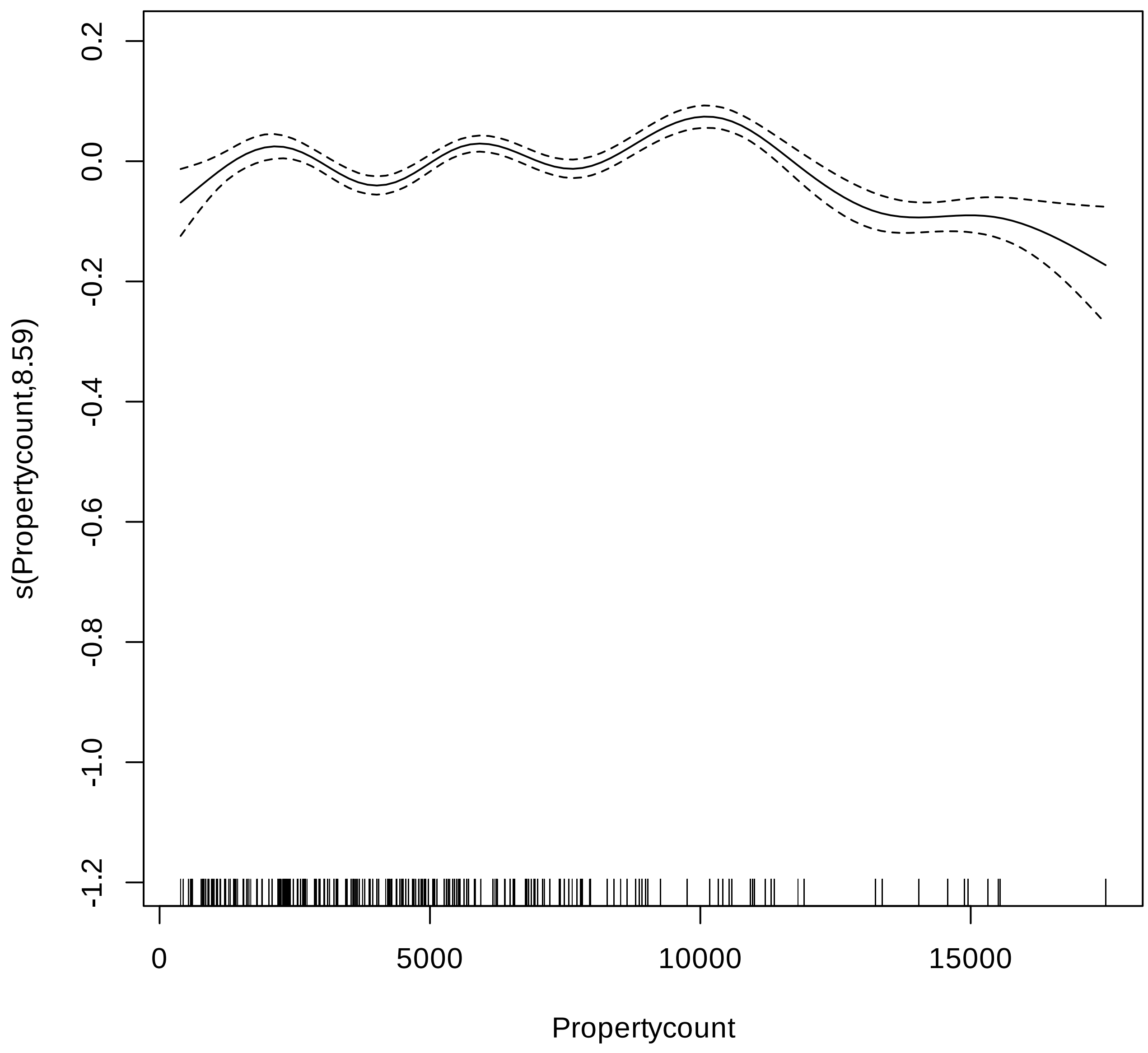
In [25]:
```
plot.gam(gamod)

# do not need s() for Month
```

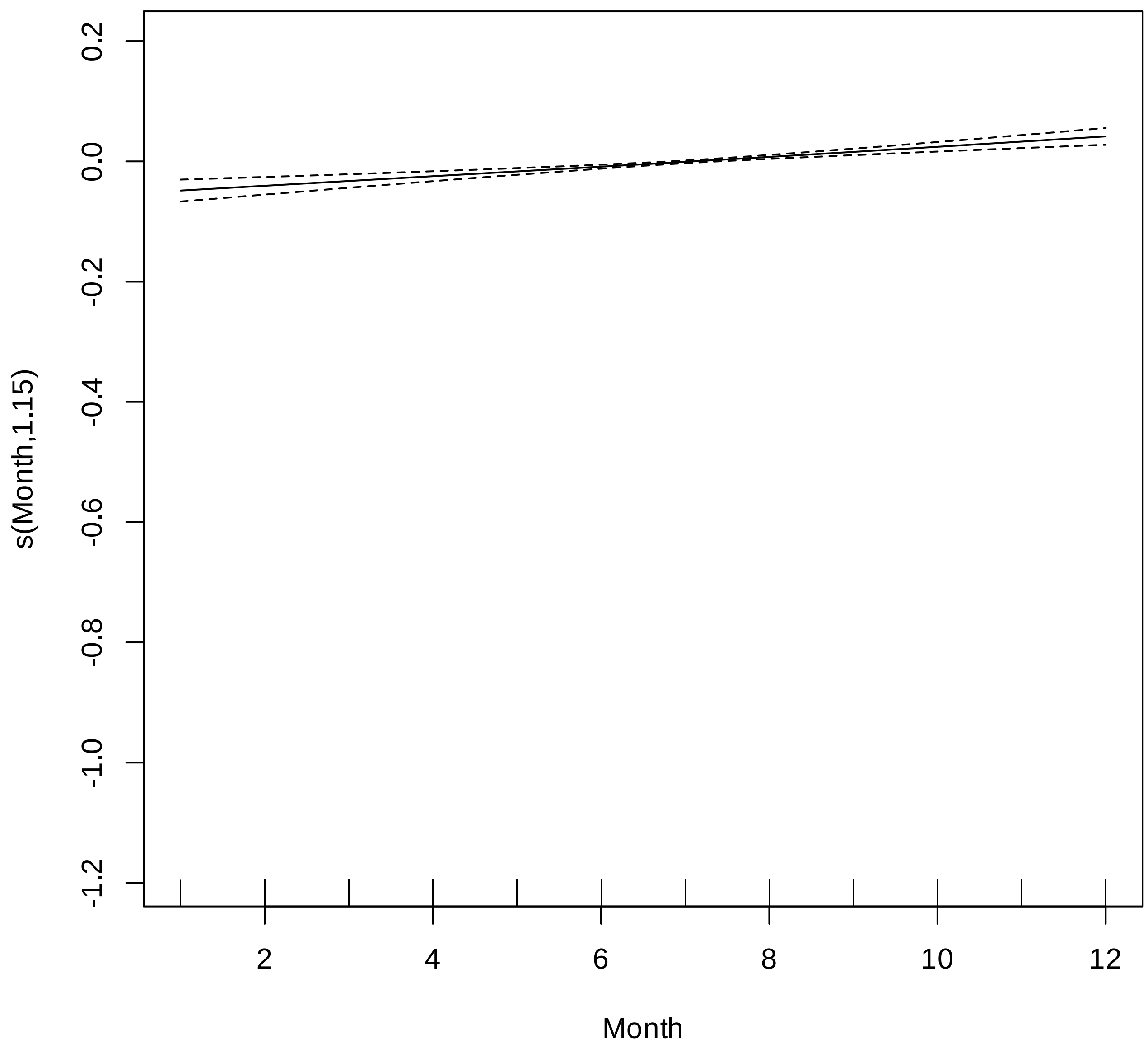

s(Distance,8.18)
Distance
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
-1.2
0
5
10
15
20


s(Landsize,8.36)
Landsize
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
-1.2
0
200
400
600
800
1000
1200


s(Lattitude,7.71)
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
-1.2
-38.0
-37.9
-37.8
-37.7
-37.6
Lattitude


s(Longtitude,8.8)
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
-1.2
144.8
144.9
145.0
145.1
145.2
Longtitude





In [26]:
```
gamod2 = gam(Price ~ Rooms + s(Distance) + Bathroom + Car + s(Landsize) + s(Lattitude) + s(Longtitude) + s(Propertycount)
            + Year + Month + freqE_Method + freqE_Type + freqE_SellerG + freqE_Regionname
           , data=train, family=gaussian())

summary(gamod2)
```

```
Family: gaussian
Link function: identity

Formula:
Price ~ Rooms + s(Distance) + Bathroom + Car + s(Landsize) +
    s(Lattitude) + s(Longtitude) + s(Propertycount) + Year +
    Month + freqE_Method + freqE_Type + freqE_SellerG + freqE_Regionname

Parametric coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)      -1.877e+02  1.514e+01 -12.396  < 2e-16 ***
Rooms             2.240e-01  5.402e-03  41.462  < 2e-16 ***
Bathroom          9.638e-02  6.333e-03  15.220  < 2e-16 ***
Car               4.392e-02  4.815e-03   9.121  < 2e-16 ***
Year              9.920e-02  7.505e-03  13.218  < 2e-16 ***
Month             8.194e-03  1.327e-03   6.173 7.11e-10 ***
freqE_Method      1.457e-05  1.061e-06  13.725  < 2e-16 ***
freqE_Type        4.907e-05  1.697e-06  28.912  < 2e-16 ***
freqE_SellerG     3.300e-05  6.909e-06   4.776 1.83e-06 ***
freqE_Regionname  3.473e-05  5.614e-06   6.186 6.53e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
                   edf Ref.df      F p-value
s(Distance)      8.185  8.791  14.58  <2e-16 ***
s(Landsize)      8.362  8.856 123.95  <2e-16 ***
s(Lattitude)     7.715  8.603 178.06  <2e-16 ***
s(Longtitude)    8.802  8.988 124.30  <2e-16 ***
s(Propertycount) 8.594  8.941  15.93  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.733   Deviance explained = 73.5%
GCV = 0.057664  Scale est. = 0.057217   n = 6667
```

In [27]:

```
# Goodness of fit for test data
##### MLR #####
lm_model = lm(Price ~., data=train)
print("##### MLR #####")

y_true = test$Price
pred_mlr = predict(lm_model, newdata=test, type="response")

mse = mean((pred_mlr - test$Price)^2)
rmse = sqrt(mse)

cat("RMSE on test set is: ", rmse, "\n")

# R^2
rss = sum((y_true - pred_mlr)^2)
tss = sum((y_true - mean(y_true))^2)

r_squared = 1 - rss / tss
cat("R-squared:", r_squared, "\n")

# adj R^2
n = length(y_true)
p = length(coef(lm_model)) - 1  # remove intercept

adj_r_squared = 1 - (1 - r_squared) * (n - 1) / (n - p - 1)
cat("Adjusted R-squared:", adj_r_squared)
```

```
[1] "##### MLR #####"
RMSE on test set is:  0.269899
R-squared: 0.6549829
Adjusted R-squared: 0.6537665
```

In [28]:

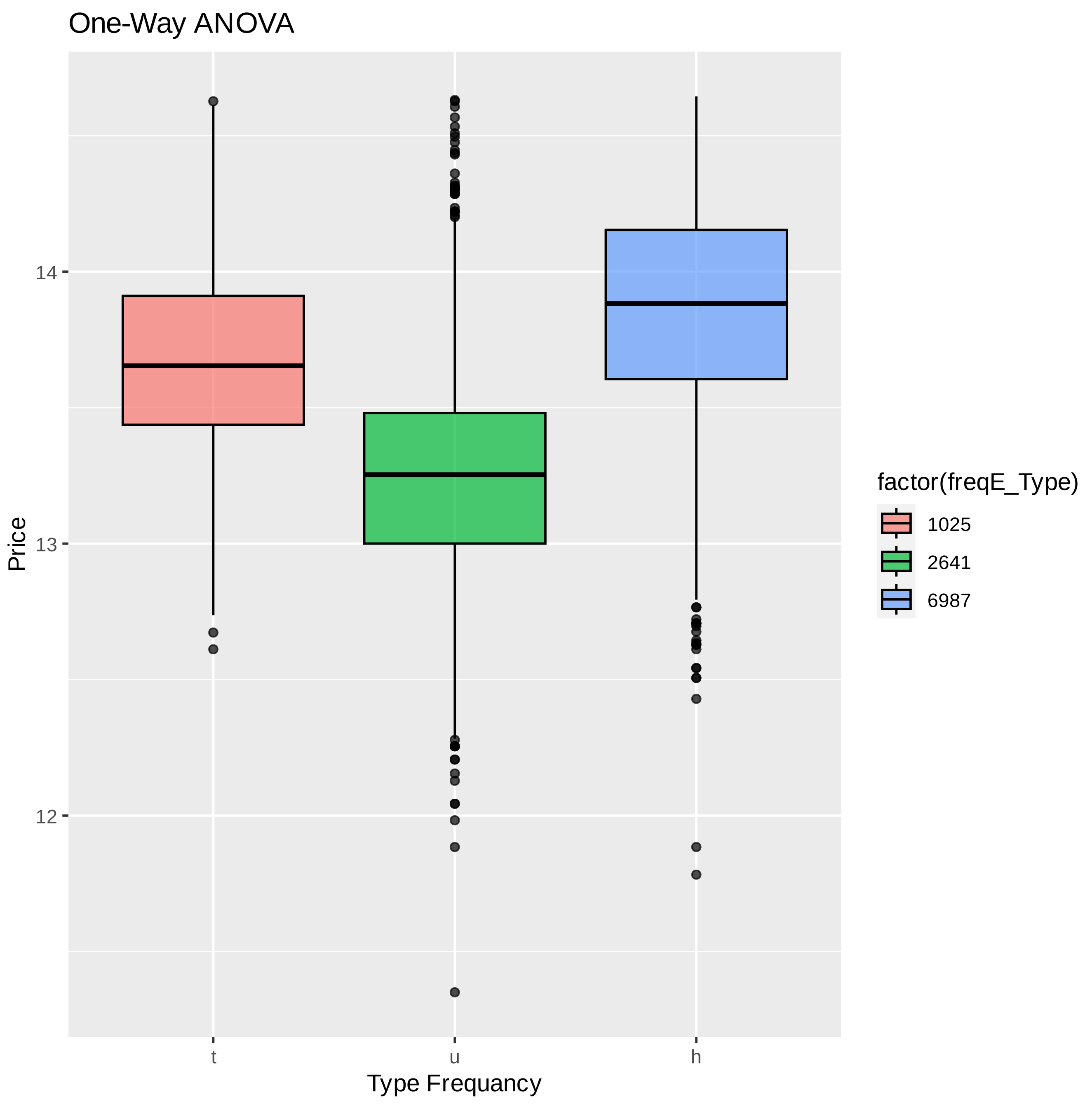
```
# Goodness of fit for test data
##### GAMs #####
print("##### GAMs #####")

y_true = test$Price
pred_gam = predict(gamod2, newdata=test, type="response")


# RMSE
mse = mean((pred_gam - test$Price)^2)
rmse = sqrt(mse)

cat("RMSE on test set is: ", rmse, "\n")


# R^2
rss = sum((y_true - pred_gam)^2)
tss = sum((y_true - mean(y_true))^2)

r_squared = 1 - rss / tss
```

```
cat("R-squared:", r_squared, "\n")

# adj R^2
n = length(y_true)
p = length(coef(gamod2)) - 1  # remove intercept

adj_r_squared = 1 - (1 - r_squared) * (n - 1) / (n - p - 1)
cat("Adjusted R-squared:", adj_r_squared)
```

```
[1] "##### GAMs #####"
RMSE on test set is:  0.2465613
R-squared: 0.7120694
Adjusted R-squared: 0.7081141
```

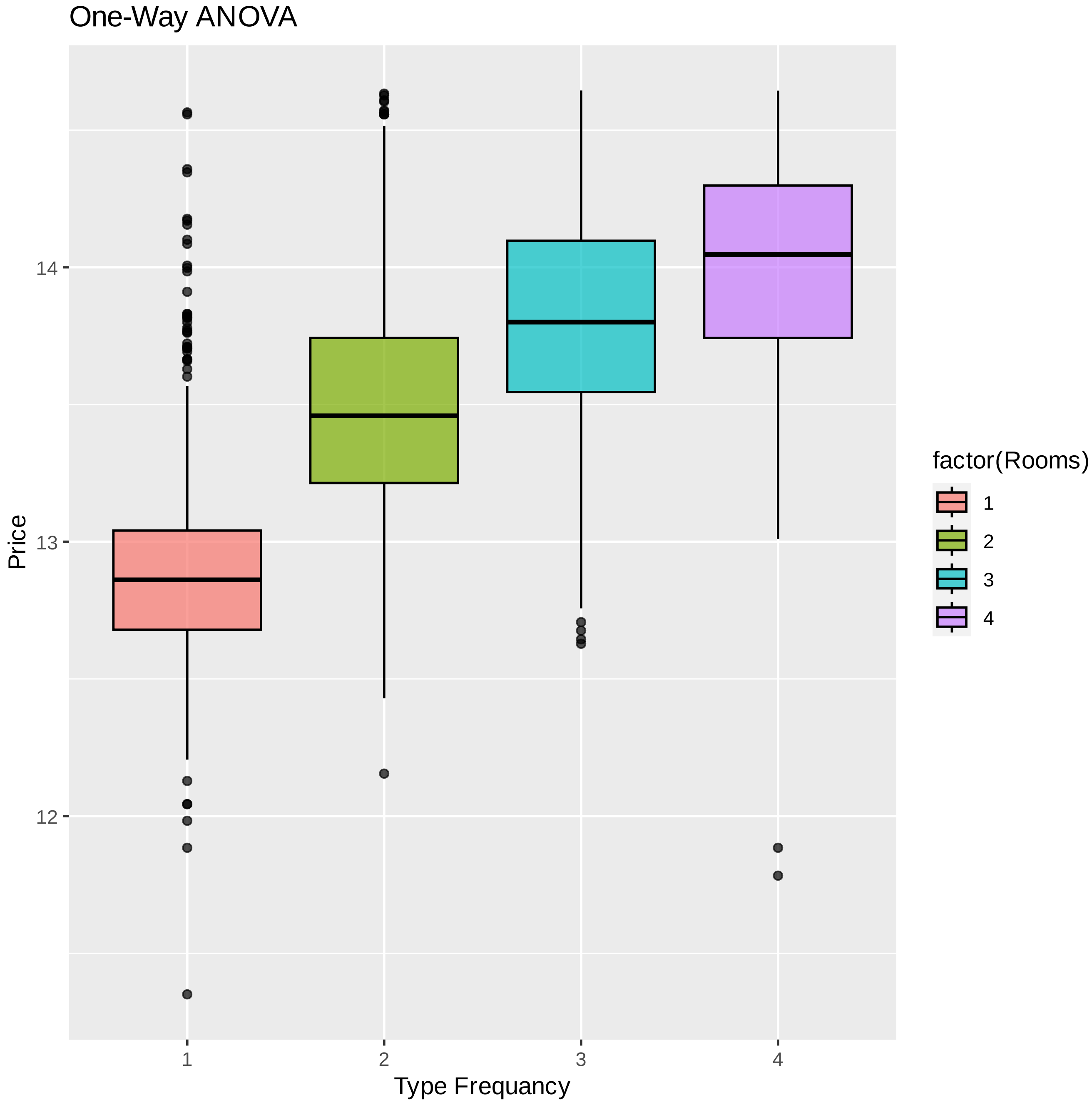
In [29]:
```
# One-way Anova (freqE_Type)
anova_type = aov(Price ~ freqE_Type , data=df_processed)
summary(anova_type)

library(ggplot2)
plot_type = ggplot(df_processed, aes(x = factor(freqE_Type), y = Price, fill = factor(freqE_Type))) +
  geom_boxplot(color = "black", alpha = 0.7) +
  labs(title = "One-Way ANOVA",
       x = "Type Frequancy",
       y = "Price") +
  scale_x_discrete(labels = c("1025" = "t", "2641" = "u", "6987" = "h"))

plot_type
# h:6987, u:2641, t:1025
```

```
               Df Sum Sq Mean Sq F value Pr(>F)
freqE_Type      1  509.1   509.1    3078 <2e-16 ***
Residuals   10651 1761.4     0.2
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```



In [30]:
```
# One-way Anova (Rooms)
anova_type2 = aov(Price ~ Rooms, data=df_processed)
summary(anova_type2)

plot_room = ggplot(df_processed, aes(x = factor(Rooms), y = Price, fill = factor(Rooms))) +
  geom_boxplot(color = "black", alpha = 0.7) +
  labs(title = "One-Way ANOVA",
       x = "Type Frequancy",
       y = "Price") +
  scale_x_discrete(labels = c("1025" = "t", "2641" = "u", "6987" = "h"))

plot_room
```

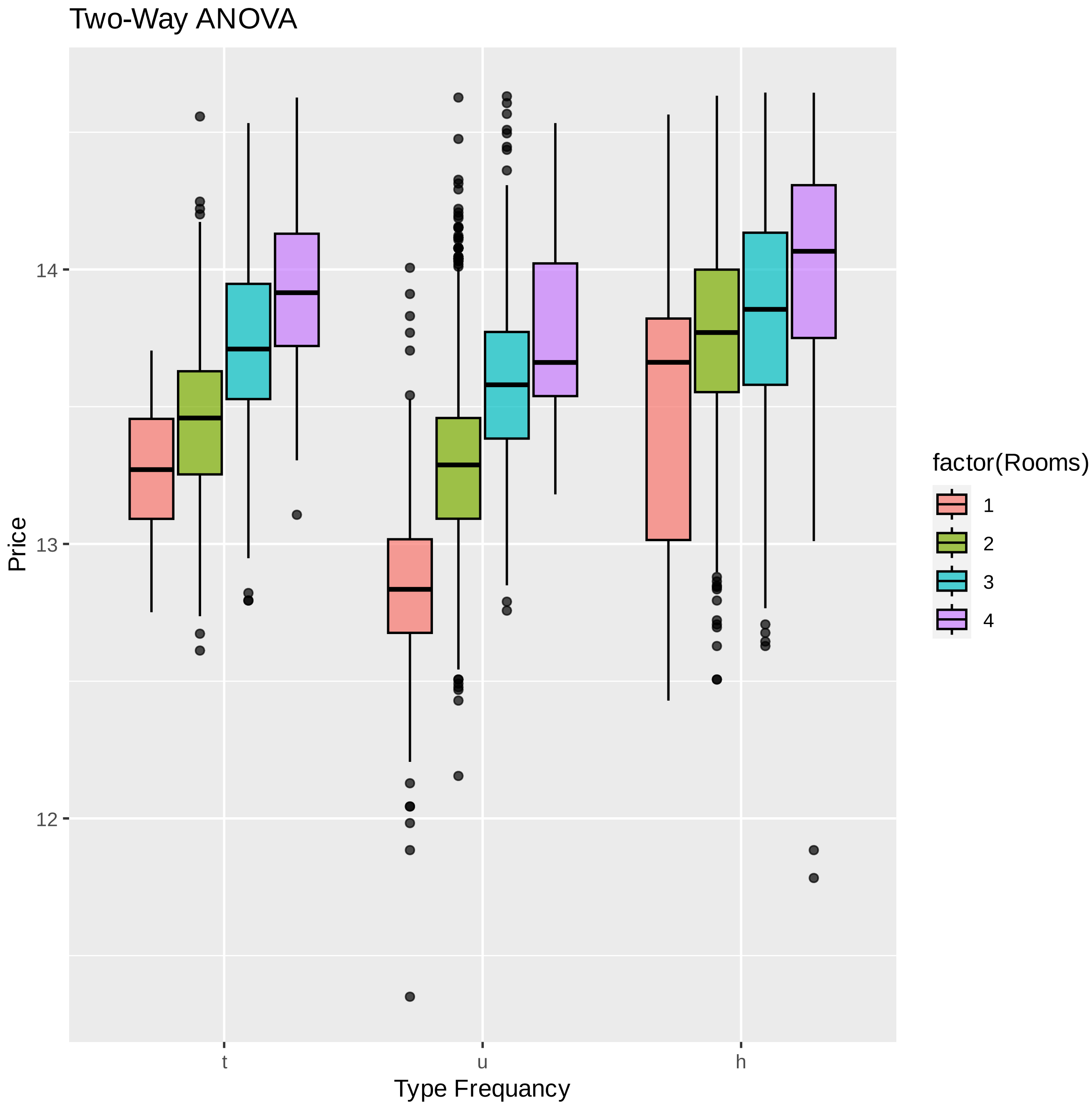
```
                Df Sum Sq Mean Sq F value Pr(>F)    
Rooms            1  720.1   720.1    4947 <2e-16 ***
Residuals    10651 1550.4     0.1                   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```



In [31]:

```
# Two-way Anova (Type and Rooms)
anova2 = aov(Price ~ Rooms * freqE_Type, data=df_processed)
summary(anova2)

plot_room = ggplot(df_processed, aes(x = factor(freqE_Type), y = Price, fill = factor(Rooms))) +
  geom_boxplot(color = "black", alpha = 0.7) +
  labs(title = "Two-Way ANOVA",
       x = "Type Frequancy",
       y = "Price") +
  scale_x_discrete(labels = c("1025" = "t", "2641" = "u", "6987" = "h"))

plot_room
```

```
                    Df Sum Sq Mean Sq F value Pr(>F)    
Rooms                1  720.1   720.1  5795.7 <2e-16 ***
freqE_Type           1  144.8   144.8  1165.7 <2e-16 ***
Rooms:freqE_Type     1   82.5    82.5   664.1 <2e-16 ***
Residuals        10649 1323.1     0.1                   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```



In [32]:
```
gamod3 = gam(Price ~ Rooms + s(Distance) + Bathroom + Car + s(Landsize) + s(Lattitude) + s(Longtitude) + s(Propertycount)
            + Year + Month + freqE_Method + freqE_Type + freqE_SellerG + freqE_Regionname + freqE_Type:Rooms
           , data=train, family=gaussian())

summary(gamod3)
# Remove Month

gamod4 = gam(Price ~ Rooms + s(Distance) + Bathroom + Car + s(Landsize) + s(Lattitude) + s(Longtitude) + s(Propertycount)
            + Year + freqE_Method + freqE_Type + freqE_SellerG + freqE_Regionname + freqE_Type:Rooms
           , data=train, family=gaussian())

summary(gamod4)
```

```
Family: gaussian
Link function: identity

Formula:
Price ~ Rooms + s(Distance) + Bathroom + Car + s(Landsize) +
    s(Lattitude) + s(Longtitude) + s(Propertycount) + Year +
    Month + freqE_Method + freqE_Type + freqE_SellerG + freqE_Regionname +
    freqE_Type:Rooms

Parametric coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)      -3.199e-04  3.287e-04  -0.973    0.330
Rooms             4.459e-01  1.110e-02  40.189  < 2e-16 ***
Bathroom          9.761e-02  6.169e-03  15.822  < 2e-16 ***
Car               4.093e-02  4.693e-03   8.721  < 2e-16 ***
Year              5.891e-03  1.693e-05 347.954  < 2e-16 ***
Month            -5.739e-04  1.104e-03  -0.520    0.603
freqE_Method      1.251e-05  1.036e-06  12.070  < 2e-16 ***
freqE_Type        1.653e-04  5.385e-06  30.696  < 2e-16 ***
freqE_SellerG     3.257e-05  6.730e-06   4.840 1.33e-06 ***
freqE_Regionname  3.484e-05  5.472e-06   6.368 2.05e-10 ***
Rooms:freqE_Type -4.305e-05  1.888e-06 -22.798  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
                   edf Ref.df      F p-value
s(Distance)      8.209  8.802  18.90  <2e-16 ***
s(Landsize)      8.517  8.914  97.72  <2e-16 ***
s(Lattitude)     7.769  8.632 175.77  <2e-16 ***
s(Longtitude)    8.832  8.991 122.87  <2e-16 ***
s(Propertycount) 8.641  8.953  15.45  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Rank: 55/56
R-sq.(adj) =  0.747   Deviance explained = 74.9%
GCV = 0.054716  Scale est. = 0.054289  n = 6667
Family: gaussian
Link function: identity

Formula:
Price ~ Rooms + s(Distance) + Bathroom + Car + s(Landsize) +
    s(Lattitude) + s(Longtitude) + s(Propertycount) + Year +
    freqE_Method + freqE_Type + freqE_SellerG + freqE_Regionname +
    freqE_Type:Rooms

Parametric coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)      -2.106e-04  3.612e-04  -0.583     0.56
Rooms             4.459e-01  1.109e-02  40.188  < 2e-16 ***
Bathroom          9.762e-02  6.169e-03  15.824  < 2e-16 ***
Car               4.085e-02  4.691e-03   8.709  < 2e-16 ***
Year              5.889e-03  1.657e-05 355.339  < 2e-16 ***
freqE_Method      1.250e-05  1.036e-06  12.068  < 2e-16 ***
freqE_Type        1.653e-04  5.384e-06  30.693  < 2e-16 ***
freqE_SellerG     3.259e-05  6.730e-06   4.843 1.31e-06 ***
freqE_Regionname  3.483e-05  5.471e-06   6.366 2.07e-10 ***
Rooms:freqE_Type -4.304e-05  1.888e-06 -22.795  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
                   edf Ref.df      F p-value
s(Distance)      8.205  8.801  18.90  <2e-16 ***
s(Landsize)      8.516  8.913  97.81  <2e-16 ***
s(Lattitude)     7.771  8.633 175.81  <2e-16 ***
s(Longtitude)    8.832  8.991 122.87  <2e-16 ***
s(Propertycount) 8.640  8.953  15.43  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Rank: 54/55
R-sq.(adj) =  0.747   Deviance explained = 74.9%
GCV = 0.054701  Scale est. = 0.054283  n = 6667
```

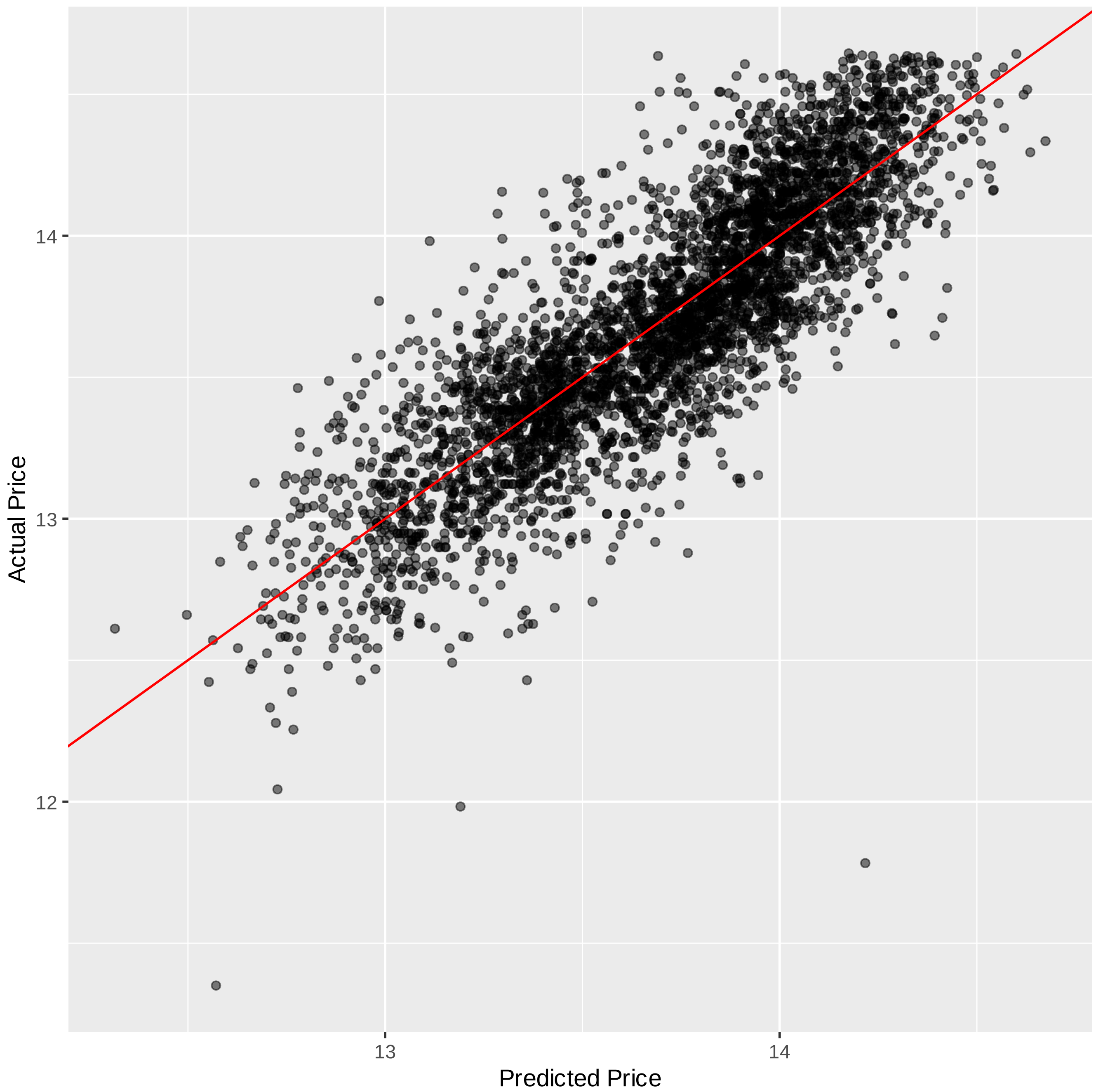
In [33]:

```
# Goodness of fit for test data
##### GAMs with interaction #####
print("##### GAMs #####")

y_true = test$Price
pred_gam = predict(gamod4, newdata=test, type="response")


# RMSE
mse = mean((pred_gam - test$Price)^2)
rmse = sqrt(mse)
```

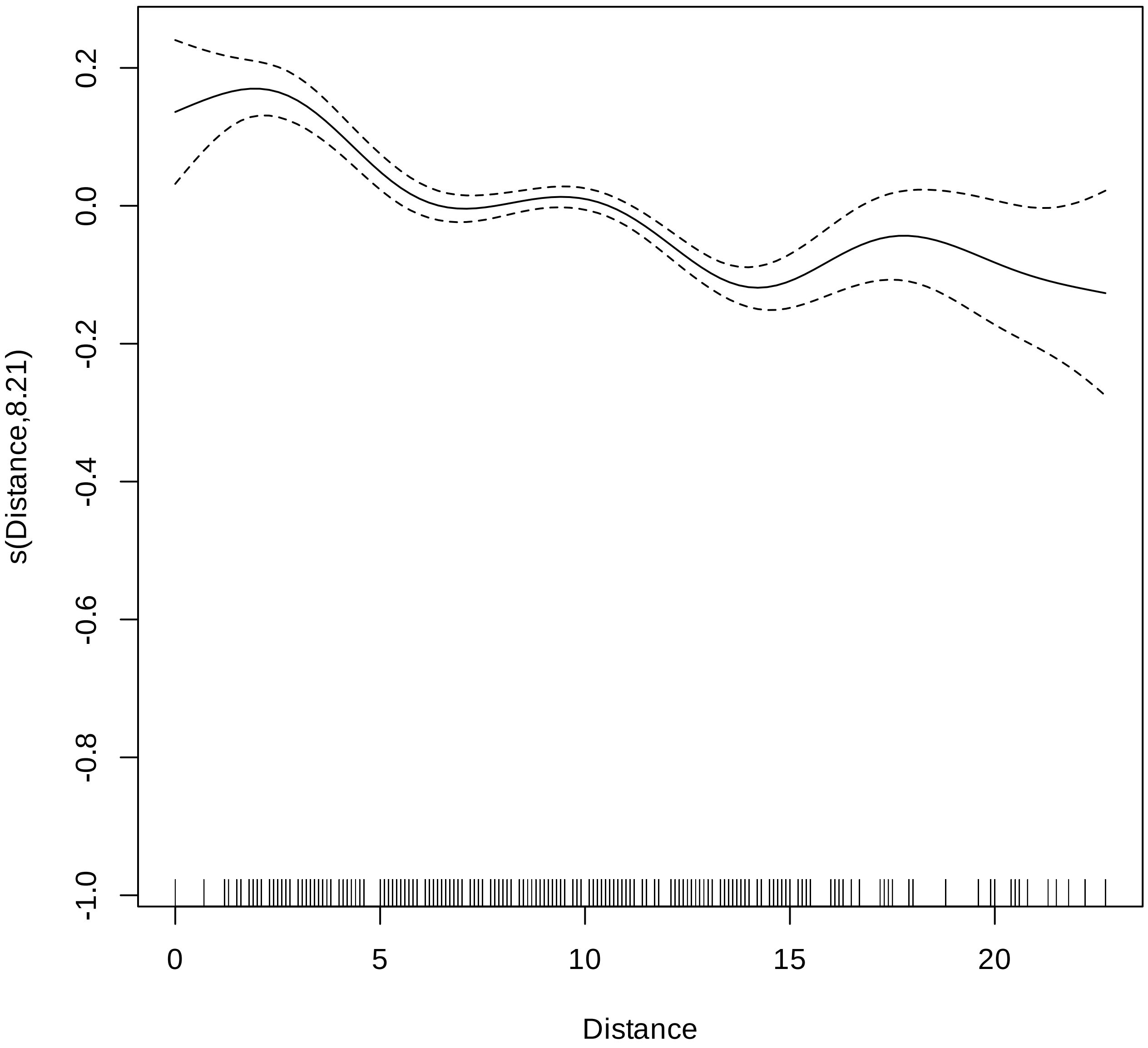
```
cat("RMSE on test set is: ", rmse, "\n")


# R^2
rss = sum((y_true - pred_gam)^2)
tss = sum((y_true - mean(y_true))^2)

r_squared = 1 - rss / tss
cat("R-squared:", r_squared, "\n")

# adj R^2
n = length(y_true)
p = length(coef(gamod4)) - 1  # remove intercept

adj_r_squared = 1 - (1 - r_squared) * (n - 1) / (n - p - 1)
cat("Adjusted R-squared:", adj_r_squared)
```

```
[1] "##### GAMs #####"
RMSE on test set is:  0.2398953
R-squared: 0.7274279
Adjusted R-squared: 0.7236835
```

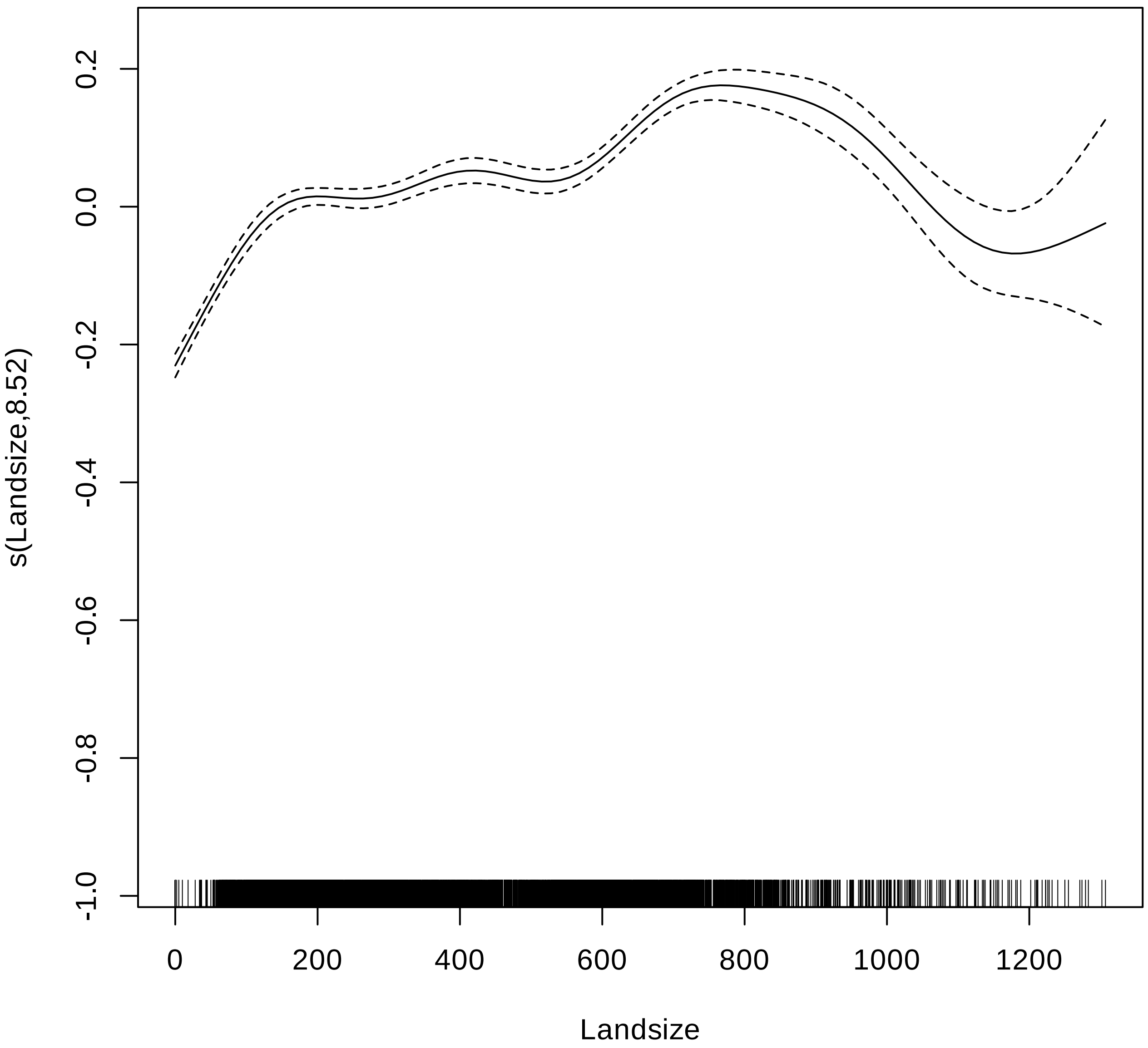
In [34]:
```
# Predicted values vs Actual values
library(ggplot2)
ggplot(test, aes(x = pred_gam, y = Price)) +
  geom_point(alpha = 0.5) +
  geom_abline(slope = 1, intercept = 0, color = "red") +
  labs(x = "Predicted Price", y = "Actual Price")
```



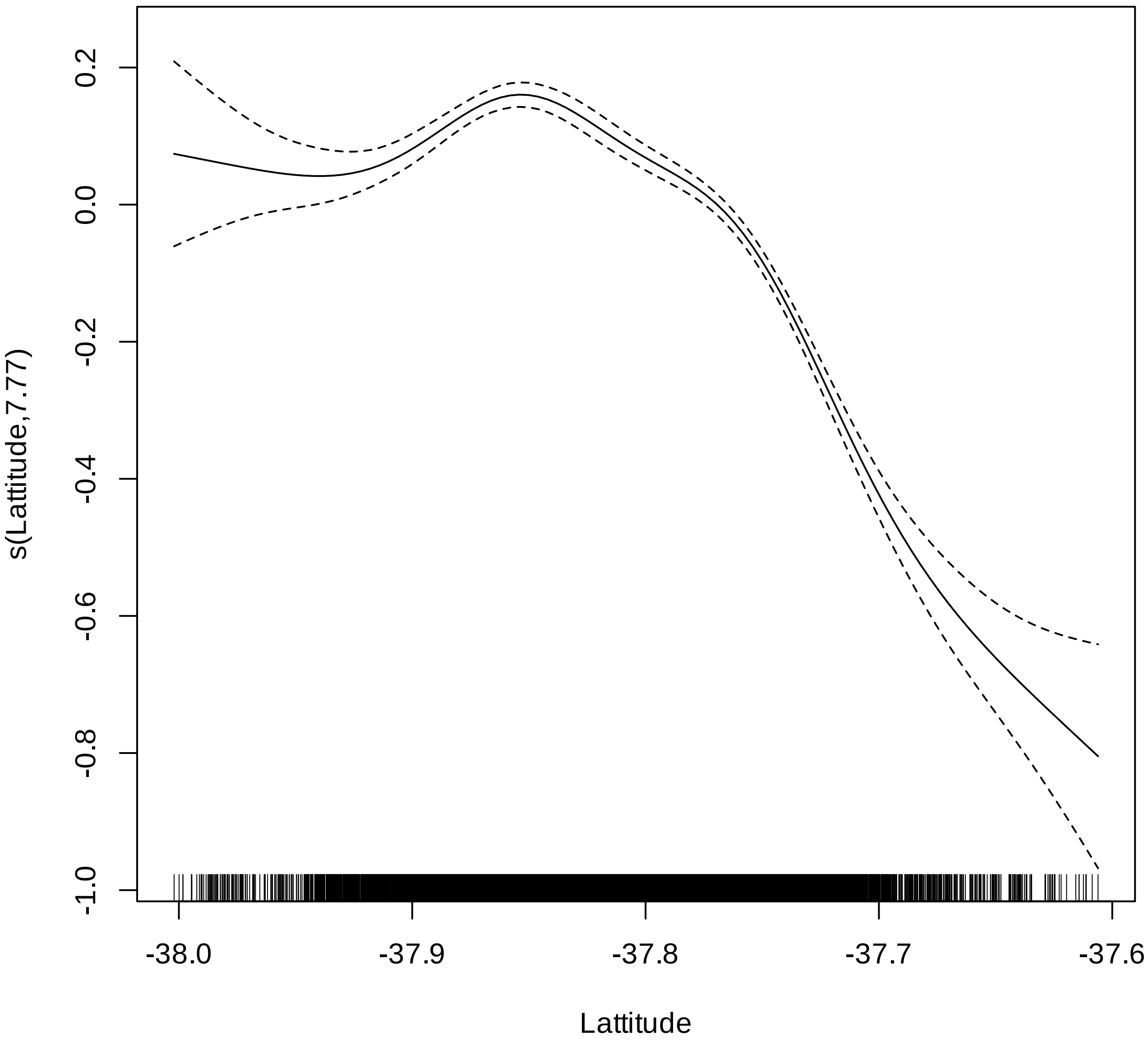
In [35]:
```
plot.gam(gamod4)
```



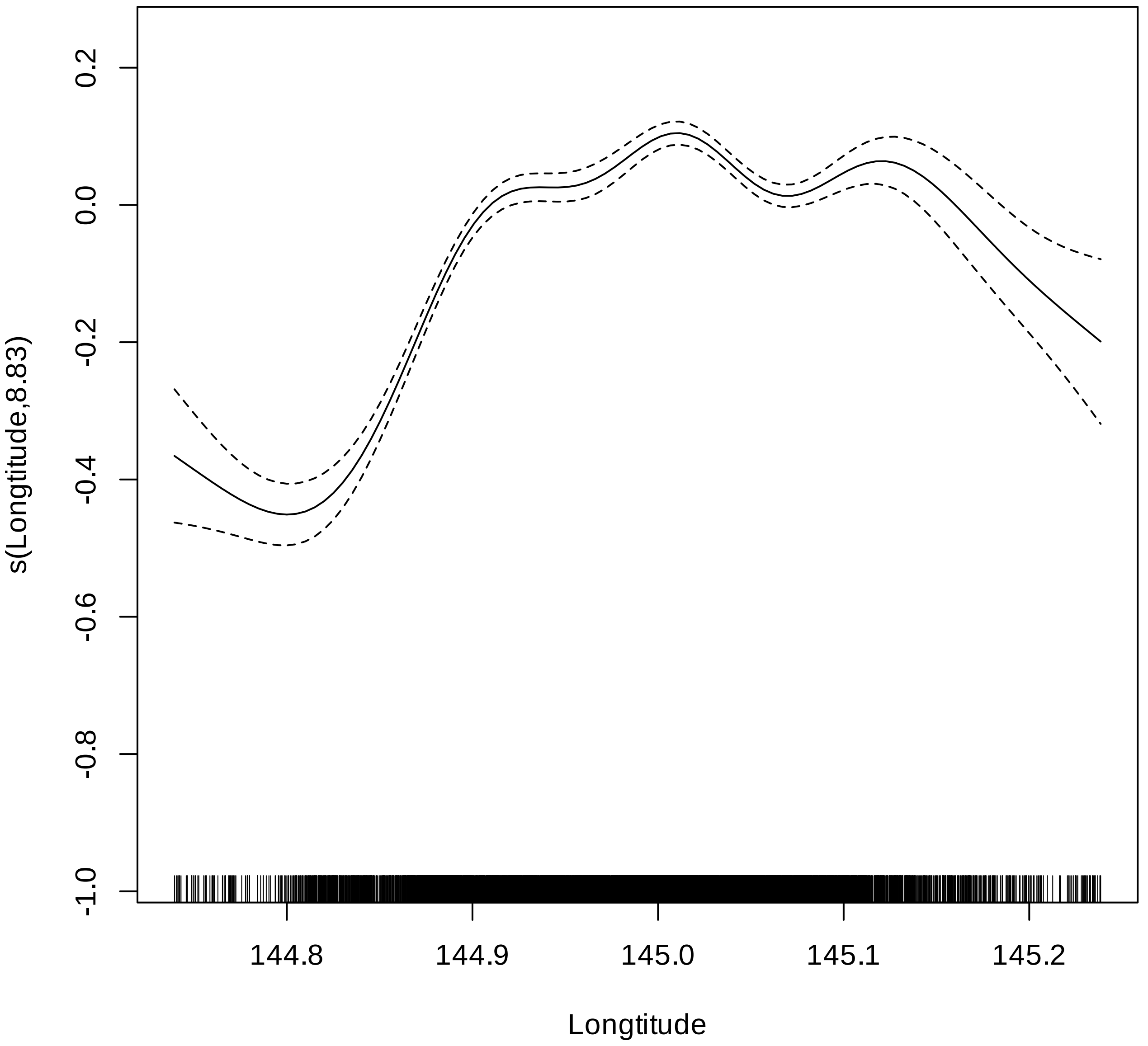
s(Distance,8.21)
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
0
5
10
15
20
Distance



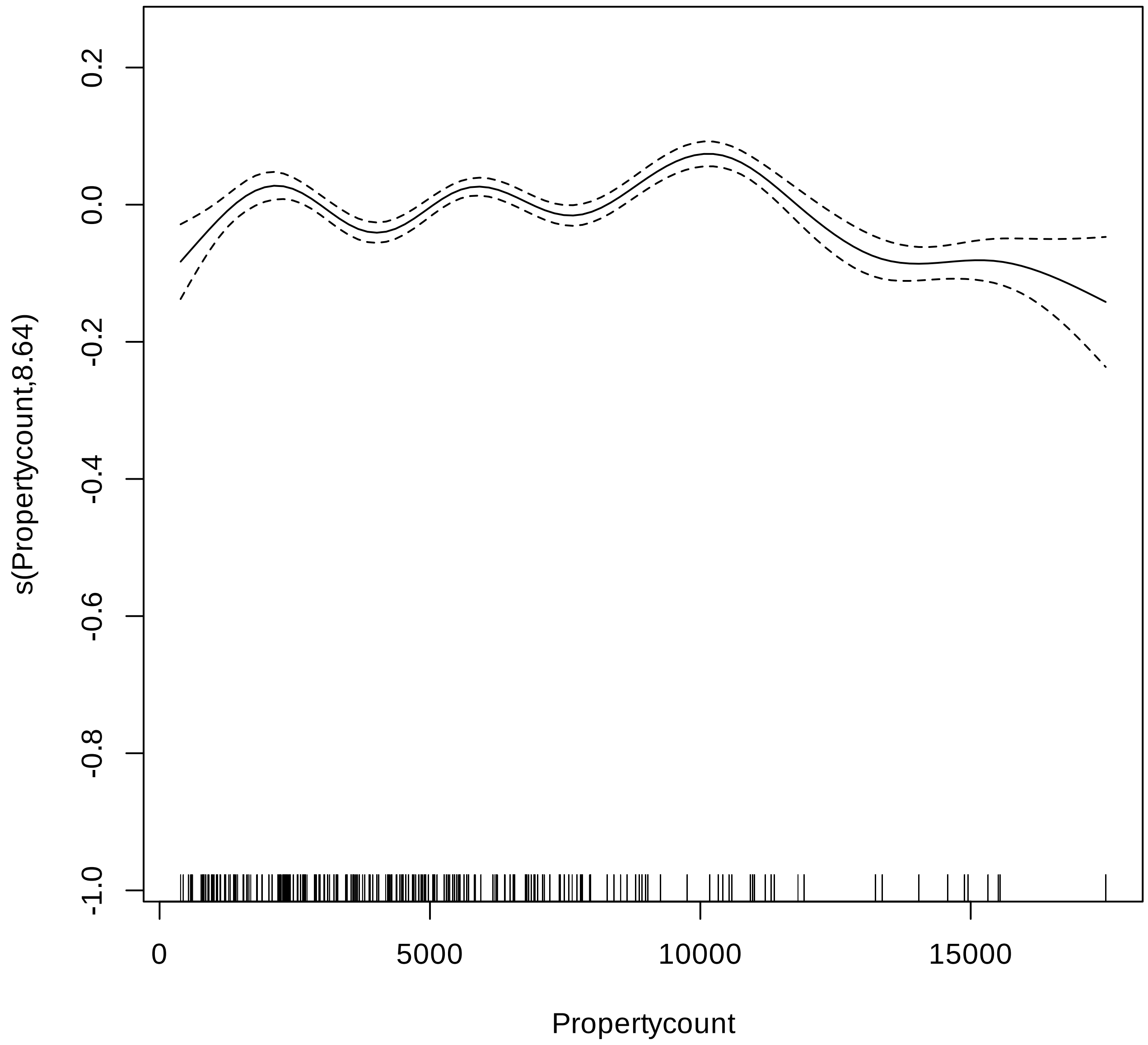
s(Landsize,8.52)
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
0
200
400
600
800
1000
1200
Landsize



s(Lattitude,7.77)
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
-38.0
-37.9
-37.8
-37.7
-37.6
Lattitude



s(Longtitude,8.83)
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
144.8
144.9
145.0
145.1
145.2
Longtitude



3, Data Analysis

1. Data Preprocessing

Before applying any models, I conducted data preprocessing to clean and prepare the dataset. I removed unnecessary columns such as `Address` and `Postcode`, as these are unique identifiers and not useful for modeling. I also removed the column `Bedroom2` because it showed a high positive correlation with `Bedroom`, making it redundant.

For missing values:

- `Car` : 62 missing values : filled with the median value.
- `BuildingArea` (6450 missing) and `YearBuilt` (5375 missing) : these columns were removed to avoid introducing bias through imputation.

I also removed outliers from numeric variables to prevent skewed results.

From the `Date` column, I created three new features: `Year`, `Month`, and `Day`. For categorical variables, I applied frequency encoding to convert them into numeric form.

Finally, I generated a correlation heatmap to check for multicollinearity. The highest correlation was 0.73 between `freqE_suburb` and `Propertycount`, which is below the commonly used threshold of 0.8, so no further action was needed.

2. Modeling and Analysis

In my final project, I applied several techniques from the course:

- regression modeling
- Multiple Linear Regression (MLR)
- Model diagnostics (goodness of fit, residual analysis)
- t-tests
- F-tests
- Model selection using AIC, BIC, MSPE, R^2, Adjusted R^2
- Generalized Additive Models (GAM)
- ANOVA

2.1 Multiple Linear Regression

I first applied a Multiple Linear Regression model using `Price` as the response variable and all other relevant features as predictors.
I chose MLR because it's simple, interpretable, and provides insight into the significance of predictors via t-tests. The overall model's significance was evaluated using an F-test.

Using diagnostic plots like the QQ-plot and Residuals vs Fitted plot, I checked assumptions (normality of residuals) and considered whether a log transformation was needed.

To detect multicollinearity, I used the Variance Inflation Factor(VIF).

2.2 Model Selection

I used AIC, BIC, and R^2 to compare and select models:

- Lower AIC and BIC values indicate better models.
- Higher R^2 and Adjusted R^2 indicate better explanatory power.

2.3 Model Implementation

Using the data set I selected from above process, I splitted it into train and test set.

- train : 80% of data set
- test : 20% of data set

Then, I trained two models: MLR and GAM with train set.

2.4 Goodness of Fit Comparison

I compared the goodness of fit for two models
To evaluate performance, I used:

- RMSE
- R^2
- Adjusted R^2

This helped me identify which model had the highest predictive accuracy among the three.

2.5 Checking the interaction

I used ANOVA to check if there is interaction or not in important variables.

4, Results and Conclusions

4-1, Used Techniques

- regression modeling
- Multiple Linear Regression (MLR)
- Model diagnostics (goodness of fit, residual analysis)
- t-tests
- F-tests
- Model selection using AIC, BIC, MSPE, R^2, Adjusted R^2
- Generalized Additive Models (GAM)
- ANOVA

4-2, Results

After applied MLR, I used "summry function" to output statistical information about MLR. From the output, I found that `Day`, `freqE_Suburb`, `freqE_CouncilArea` are not statistically significant as these p-values are over 0.05. So, I deleted three columns. As for F-test, the F-value is 1072, which means that the model explains a large part of the variance in housing prices. Also, the p-value is 2.2e-16, indicating that we reject the null hypothesis that none of the predictors are related to the response. Then, using the MLR model, I created Q-Q plot and residual vs fitted plot. From the Q-Q plot, the upper tail does not follow the line. So, we can say that it is skewed. Also, from the residuals vs fitted plot, we can see that the variance is non-constant. Because there is no "rectangular" shape, and it looks like corn shaped. Also, the red line is not straight and looks curved. So, we can say that there is a heteroscdasticity. Therefore, I applied the log transformation to response variable. Additionally, I applied VIF to the model, then the output showed that there is no columns that the vif values exceeds 5, indicating no multicollinearity exists.

After that, I split the data into train and test sets. Using the train set, I checked the AIC, BIC, and R^2 using regression model to get predictors that improve the model accuracy. As a result, the model using all predictors had the greatest scores for all metrics, so I decided to use all of them in my models.

Next, I applied three model; MLR and GAM, and evaluated performance and checked the goodness of fit.

For Multiple Linear Regression, the RMSE, R-squared, and adjusted R-squared are 0.270, 0.655, and 0.654, respectively. For, Generalized Additive Model, the RMSE, R-squared, and adjusted R-squared are 0.247, 0.712, and 0.708, respectively. From the results, I can say that GAM is the better mode.

Next, I analyzed the difference in mean prices across the `freqE_Type` and `Rooms` columns using ANOVA. First, I perforemd one-way anova. From the p-value in `freqE_Type`, I can rejected the null hypothesis that all means are equal. Next, I created a boxplot to visualize the mean price for each property type. The box plot shows that Type h has the highest mean price, while Type u has the lowest mean price. Therefore, properties that are houses, cottages, villas, semis, or terraces are typically more expensive, while units and duplexes are generally more affordable. Additionally, from the p-value in `Rooms`, I can rejected the null hypothesis that all means are equal. Next, I created a boxplot to visualize the mean price for each the number of room. The box plot shows that properties with 4 rooms has the highest mean price, while properties with 1 rooms has the lowest mean price. So, the more the properties have rooms, the more expensive the properties are. Moreover, from the result of two-way anova, both `freqE_Type` and `Rooms` have significant effect on the `Price` and the interaction between `Rooms` and `freqE_Type` is also significant. So, `Rooms` and `freqE_Type` are not independent and I need to consider both the number of room and property type.

Finally, I added the interaction between `Rooms` and `freqE_Type` to the GAM model. I found that `Month` column is not significant in the model with the interaction, so I removed it, then I applied the model again. The RMSE, R-squared, and adjusted R-squared are 0.240, 0.727, and 0.724, respectively. So, the new model is greater than the previous model and I'll use the new GAM for conclustions.

4-3, Conclustions

From the GAM results, the estimated price model is:

$$Price = -187.7 + 0.446 \cdot Rooms + 0.0976 \cdot Bathroom + 0.0409 \cdot Car + 5.889 \times 10^{-3} \cdot Year + 1.250 \times 10^{-5} \cdot freqEMethod + 1.653$$
$$+ f_1(Distance) + f_2(Landsize) + f_3(Lattitude) +$$

where, the f_i represents the smooth function.

For linear features:

I found that `Rooms`, `freqE_Type`, and `Year` have the largest coefficients and t-values, suggesting they significantly affect housing prices. For `Rooms`, more rooms tend to increase the price. For `freqE_Type`, a higher value for `freqE_Type`, which indicates more popular or common property types, is also associated with higher prices. For `Year`, the newer properties are more expensive than the old ones.

For non-linear features:

The smooth terms with the largest F-values are `Lattitude`, `Longtitude`, and `Landsize`, meaning these varialbes ave strong non-linear effects on price. From the plots from GAM, I observed that properties located between 144.9 and 145.1 in longitude and between -37.9 and -37.8 in latitude tend to have higher prices. In contrast, homes located outside of these ranges, particularly below 144.8 or above 145.2 in longitude, and above -37.7 or below -37.9 in latitude, tend to be less expensive. This implies that homes in the central or slightly northern parts of Melbourne are generally more valuable.

Regarding the land size, smaller properties (under 200 [m^2]) do not significantly affect the price, while those between 200 and 800 [m^2] are associated with higher prices. However, when the land size exceeds 800 [m^2], the effect on price starts to decline. Therefore, the most favorable land size appears to be around 800 [m^2].

The features of expensive properties are:

- Rooms: more is better
- Property Type: houses, cottages, villas, semis, or terraces
- Year: newer properties are more expensive
- Lattitude: -37.9 and -37.8
- Longtitude: 144.9 and 145.1
- Landsize: around 800 [m^2]

From these findings, I can conclude that if you're considering buying a home in Melbourne, you should pay close attention to six key features: the number of rooms, the type of property, the old of houses, the location (latitude and longitude), and the land size. These factors have a significant influence on price. If a property has poor values in these features but is still priced high, it may be overvalued, and requires caution. On the other hand, if a property has strong values in these features but is priced low, it might be undervalued and could represent a good buying opportunity.

```
In [ ]:
```