Data Visualization Final

John Searight

2022-04-06

Intro:

Unfortunately, my computer and my R Studio often could not handle the large data set, so I had to work with a smaller data set of just less than 600,000 observations.

```
ukpostcodes <- read.csv("C:/Users/jcsea/Downloads/ukpostcodes.csv")

df <- read.csv('C:/Users/jcsea/OneDrive/Desktop/BGSE Documents/ppdata_liter2.csv')

#Inspect
head(df)
```

```
##             transaction_unique_identifier  price date_of_transfer postcode
## 1 {281FD95E-FC9B-4BED-B828-068AC4F5390C} 126497 10/10/2003 00:00  CM3 5FX
## 2 {70429219-F3F6-40AF-968D-278F40B60026}  58500 28/04/1995 00:00   N2 0JU
## 3 {CAB692E6-E8A9-4639-AD7F-F541A76ADD32} 150000 26/08/2011 00:00  CV2 2FU
## 4 {BB43F7DD-D060-432E-981F-00A47F95DC22}  80000 21/08/1996 00:00 BN13 1HE
## 5 {A82457A1-961F-4C37-8DC2-D541D1688639} 149500 16/06/2011 00:00 YO41 5PZ
## 6 {E21362AA-80BE-479D-9EE3-2B7F84F6390F} 440000 12/10/2011 00:00 SE10 0JZ
##   property_type old_new duration PAON SAON          street
## 1             T       N        F   72       MELVILLE HEATH
## 2             F       N        L   57        DENISON CLOSE
## 3             S       N        F    5         PANDORA ROAD
## 4             S       N        F   75          CHURCH ROAD
## 5             S       N        F   35      MOORFIELD DRIVE
## 6             T       N        F  116       ANNANDALE ROAD
##                locality  town_city                 district
## 1 SOUTH WOODHAM FERRERS CHELMSFORD               CHELMSFORD
## 2                LONDON     LONDON                   BARNET
## 3                         COVENTRY                 COVENTRY
## 4              WORTHING   WORTHING                 WORTHING
## 5            WILBERFOSS       YORK EAST RIDING OF YORKSHIRE
## 6                           LONDON                GREENWICH
##                      county PPD_category_type record_status
## 1                     ESSEX                 A             A
## 2            GREATER LONDON                 A             A
## 3             WEST MIDLANDS                 A             A
## 4               WEST SUSSEX                 A             A
## 5 EAST RIDING OF YORKSHIRE                  A             A
## 6            GREATER LONDON                 A             A
```

```
#Get dimensions
dim(df)
```

```
## [1] 567642     16
```

```
#Create log price
df$logprice <- log(df$price, 10)
```

Task A: First look at counties to see what constitutes London

```
unique(df$county)
```

```
##   [1] "ESSEX"                        "GREATER LONDON"
##   [3] "WEST MIDLANDS"                "WEST SUSSEX"
##   [5] "EAST RIDING OF YORKSHIRE"     "TYNE AND WEAR"
##   [7] "MONMOUTHSHIRE"                "GLOUCESTERSHIRE"
##   [9] "LANCASHIRE"                   "BLACKPOOL"
##  [11] "WARWICKSHIRE"                 "MILTON KEYNES"
##  [13] "GREATER MANCHESTER"           "STAFFORDSHIRE"
##  [15] "SOUTH YORKSHIRE"              "CAERPHILLY"
##  [17] "CHESHIRE"                     "WORCESTERSHIRE"
##  [19] "WEST YORKSHIRE"               "NORTHAMPTONSHIRE"
##  [21] "HAMPSHIRE"                    "SUFFOLK"
##  [23] "LEICESTERSHIRE"               "MERSEYSIDE"
##  [25] "SHROPSHIRE"                   "BRIGHTON AND HOVE"
##  [27] "WINDSOR AND MAIDENHEAD"       "HERTFORDSHIRE"
##  [29] "DEVON"                        "DERBYSHIRE"
##  [31] "EAST SUSSEX"                  "BATH AND NORTH EAST SOMERSET"
##  [33] "DORSET"                       "OXFORDSHIRE"
##  [35] "NORTHUMBERLAND"               "NORTH YORKSHIRE"
##  [37] "MEDWAY"                       "CORNWALL"
##  [39] "BRACKNELL FOREST"             "CARDIFF"
##  [41] "SURREY"                       "NORFOLK"
##  [43] "CITY OF BRISTOL"              "LINCOLNSHIRE"
##  [45] "KENT"                         "NOTTINGHAMSHIRE"
##  [47] "LUTON"                        "NORTH EAST LINCOLNSHIRE"
##  [49] "POOLE"                        "STOKE-ON-TRENT"
##  [51] "SWINDON"                      "WEST BERKSHIRE"
##  [53] "SOMERSET"                     "BUCKINGHAMSHIRE"
##  [55] "WILTSHIRE"                    "CAMBRIDGESHIRE"
##  [57] "RHONDDA CYNON TAFF"           "DARLINGTON"
##  [59] "POWYS"                        "TORFAEN"
##  [61] "FLINTSHIRE"                   "CITY OF PLYMOUTH"
##  [63] "WOKINGHAM"                    "LEICESTER"
##  [65] "WREKIN"                       "THURROCK"
##  [67] "READING"                      "PORTSMOUTH"
##  [69] "BRIDGEND"                     "HARTLEPOOL"
##  [71] "MID GLAMORGAN"                "HALTON"
##  [73] "CUMBRIA"                      "SOUTH GLOUCESTERSHIRE"
##  [75] "NORTH SOMERSET"               "YORK"
##  [77] "CHESHIRE WEST AND CHESTER"    "SOUTHEND-ON-SEA"
##  [79] "BLACKBURN WITH DARWEN"        "SLOUGH"
##  [81] "DURHAM"                       "ISLE OF WIGHT"
##  [83] "SWANSEA"                      "THE VALE OF GLAMORGAN"
##  [85] "BOURNEMOUTH"                  "STOCKTON-ON-TEES"
##  [87] "NORTH LINCOLNSHIRE"           "CITY OF KINGSTON UPON HULL"
##  [89] "PEMBROKESHIRE"                "CITY OF NOTTINGHAM"
##  [91] "NEWPORT"                      "CITY OF PETERBOROUGH"
##  [93] "BEDFORDSHIRE"                 "BEDFORD"
##  [95] "COUNTY DURHAM"                "CITY OF DERBY"
##  [97] "WREXHAM"                      "WARRINGTON"
##  [99] "SOUTHAMPTON"                  "TORBAY"
## [101] "CHESHIRE EAST"                "BERKSHIRE"
## [103] "CONWY"                        "MERTHYR TYDFIL"
## [105] "CENTRAL BEDFORDSHIRE"         "HUMBERSIDE"
## [107] "MIDDLESBROUGH"                "CARMARTHENSHIRE"
## [109] "REDCAR AND CLEVELAND"         "ISLE OF ANGLESEY"
## [111] "GWYNEDD"                      "CLWYD"
## [113] "NEATH PORT TALBOT"            "THAMESDOWN"
## [115] "BLAENAU GWENT"                "CEREDIGION"
## [117] "HEREFORDSHIRE"                "DENBIGHSHIRE"
## [119] "RUTLAND"                      "DYFED"
## [121] "AVON"                         "HEREFORD AND WORCESTER"
## [123] "GWENT"                        "CLEVELAND"
## [125] "WEST GLAMORGAN"               "SOUTH GLAMORGAN"
## [127] "ISLES OF SCILLY"
```

```
#Create london df feat only london
london <-df[df$county=="GREATER LONDON",]

#Inspect districts (boroughs)
unique(london$district)
```

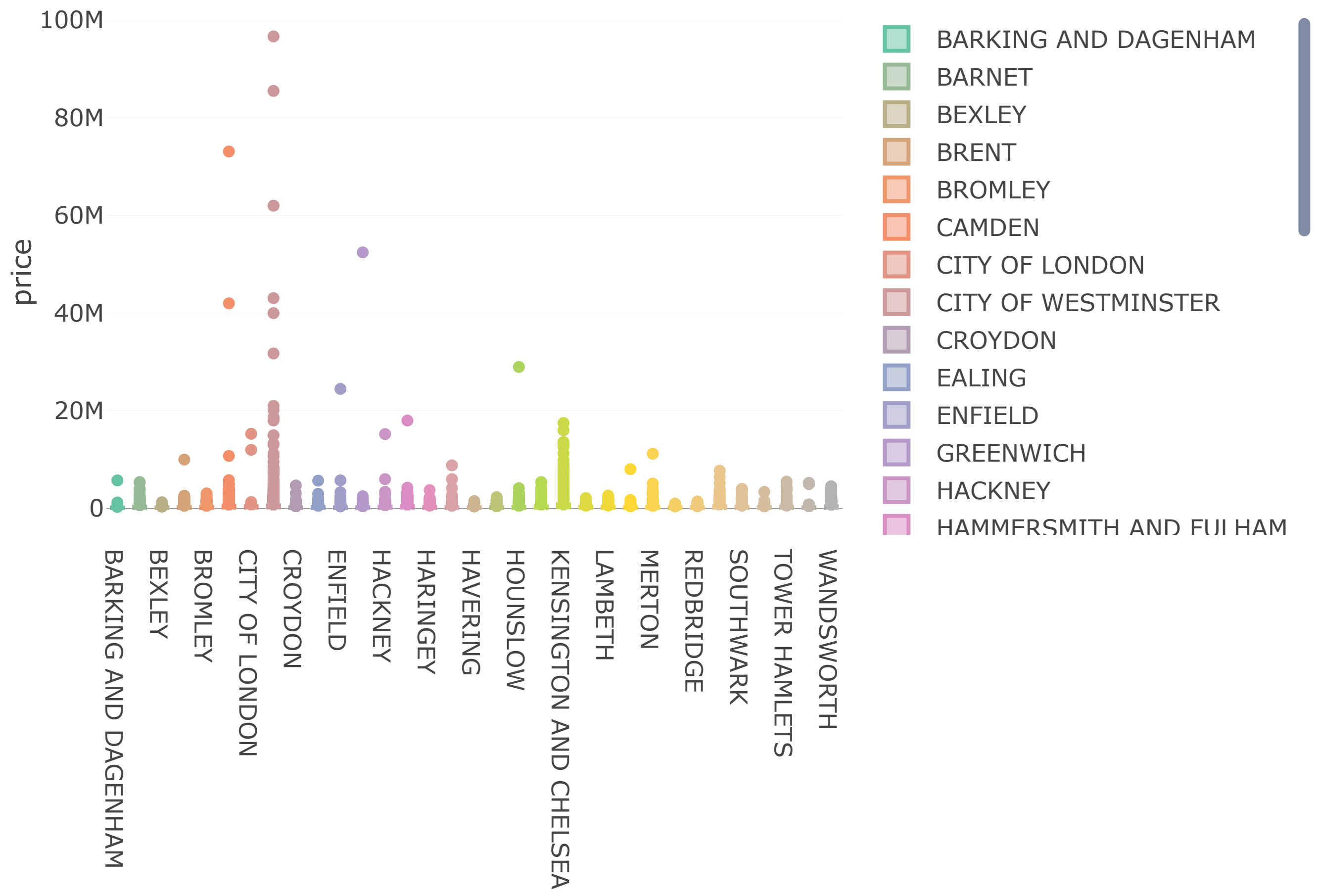
```
##  [1] "BARNET"                 "GREENWICH"              "TOWER HAMLETS"
##  [4] "KINGSTON UPON THAMES"   "KENSINGTON AND CHELSEA" "SOUTHWARK"
##  [7] "LEWISHAM"               "MERTON"                 "BRENT"
## [10] "HAVERING"               "CITY OF LONDON"         "HOUNSLOW"
## [13] "WANDSWORTH"             "CROYDON"                "ISLINGTON"
## [16] "CAMDEN"                 "LAMBETH"                "HACKNEY"
## [19] "EALING"                 "BROMLEY"                "CITY OF WESTMINSTER"
## [22] "RICHMOND UPON THAMES"   "ENFIELD"                "HARROW"
## [25] "WALTHAM FOREST"         "HILLINGDON"             "HARINGEY"
## [28] "SUTTON"                 "BARKING AND DAGENHAM"   "HAMMERSMITH AND FULHAM"
## [31] "NEWHAM"                 "BEXLEY"                 "REDBRIDGE"
```

```
#Sort alphabetically
london <- london[order(london$district), ]
```

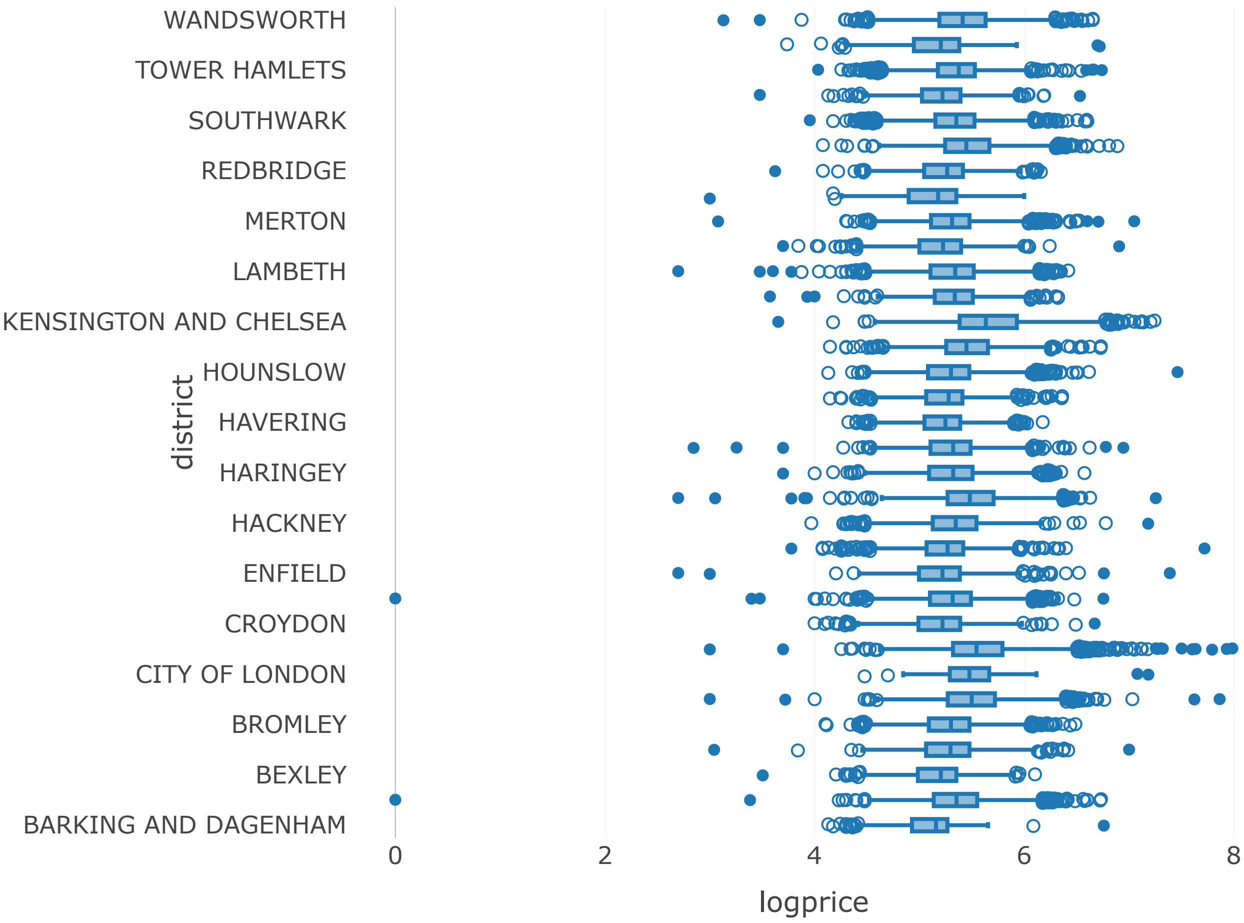
Now I use Plotly. Here are the simplest box plots featuring price and log prices

```
fig <- plot_ly(london, y = ~price, color = ~district, type = "box")
fig
```

```
## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2
is 8
## Returning the palette you asked for with that many colors

## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2
is 8
## Returning the palette you asked for with that many colors
```



```
fig2 <- plot_ly(london, x = ~logprice, y = ~district, jitter = 0.5, pointpos = 0, type = "box",
 boxpoints='suspectedoutliers')
fig2
```

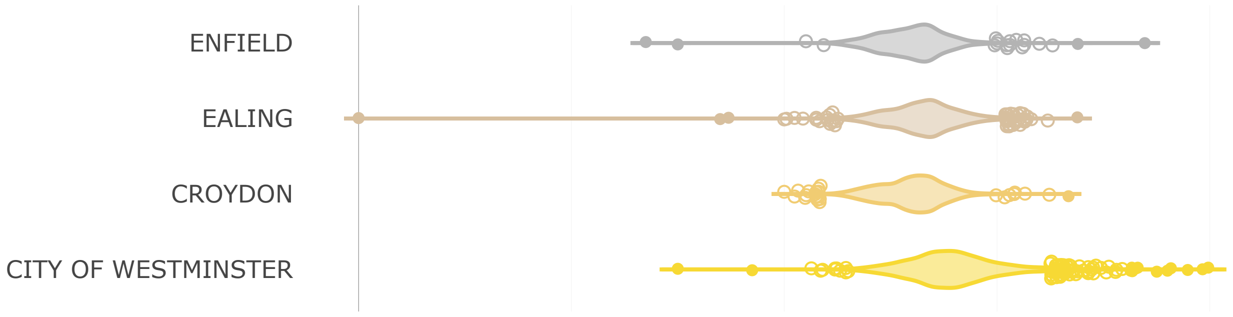


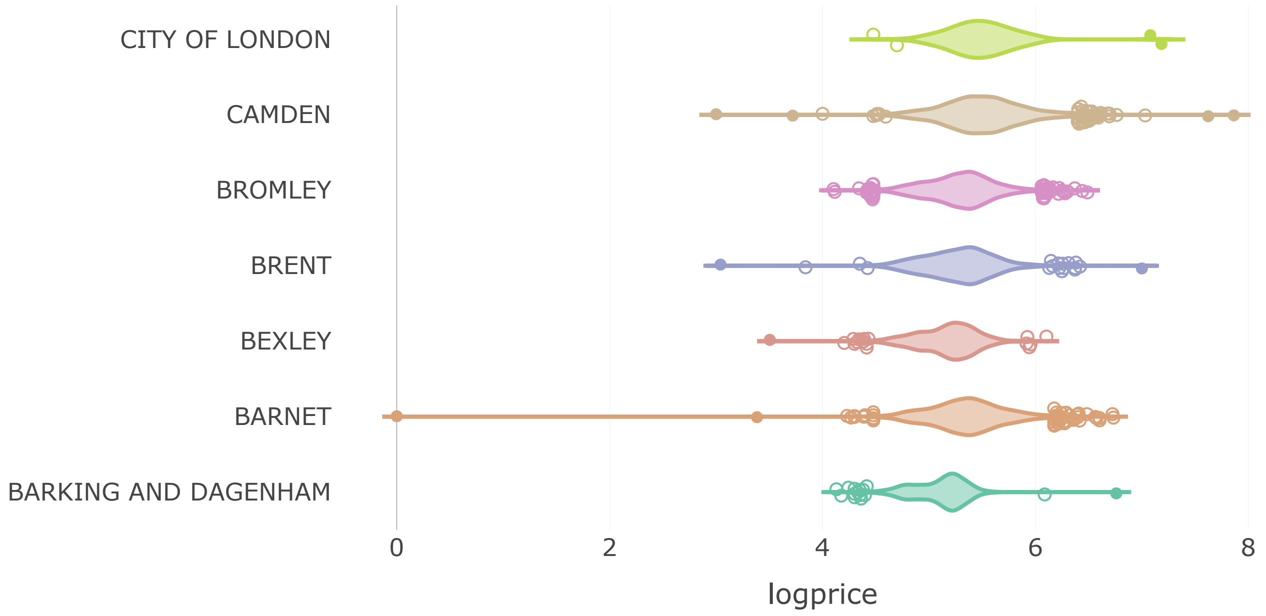
The above plots districts on the vertical access and gets rid of the legend for redundancy. I also removed the colors for minimilast feel, but it is still interactive and you can see the name of the district when hovering over. Notice the scale is now log price so that the outliers are minimized.Kensington and Chelsea continue to stand out.

It would be easier to see wider plots, so I will visualize subplots and violin plots I split so there are 10 or 11 observations in each, though could have also split it to have roughly equal number of total observations in each

```
#Violin plots might be more useful here, so I split the data evenly into 3 groups

london_ae = {london %>% filter(str_detect(district, '^[A-E]'))}
london_gk = {london %>% filter(str_detect(district, '^[G-K]'))}
london_lz = {london %>% filter(str_detect(district, '^[L-Z]'))}

fig_ae <- plot_ly(london_ae, x = ~logprice, jitter = 0.5, pointpos = 0, color = ~district, type
 = "violin", points='suspectedoutliers', showlegend = FALSE)
fig_gk <- plot_ly(london_gk, x = ~logprice, jitter = 0.5, pointpos = 0, color = ~district, type
 = "violin", points='suspectedoutliers', showlegend = FALSE)
fig_lz <- plot_ly(london_lz, x = ~logprice, jitter = 0.5, pointpos = 0, color = ~district, type
 = "violin", points='suspectedoutliers', showlegend = FALSE)

#I can inspect each one individually for a closer look
fig_ae
```

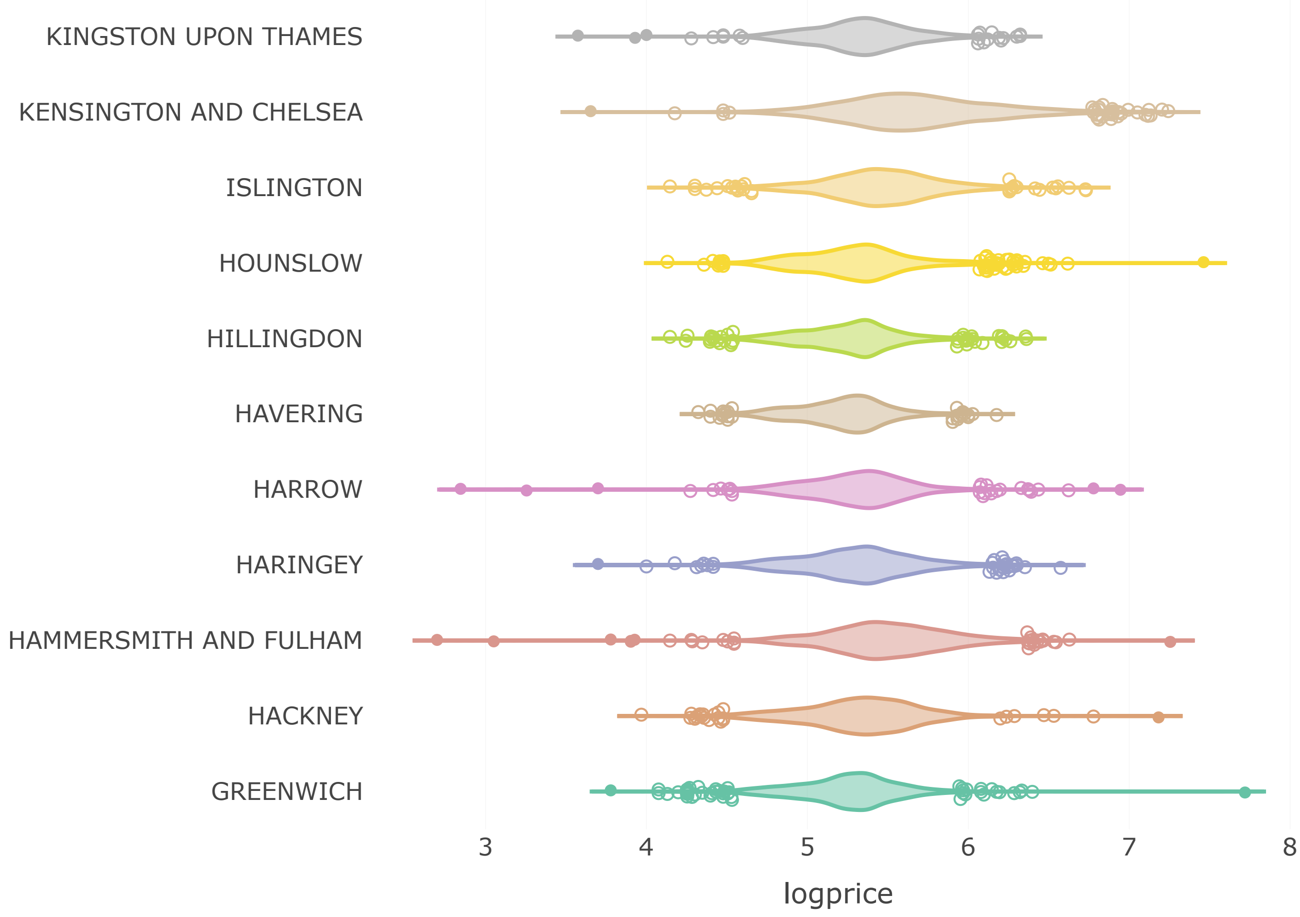
```
## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2
is 8
## Returning the palette you asked for with that many colors

## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2
is 8
## Returning the palette you asked for with that many colors
```





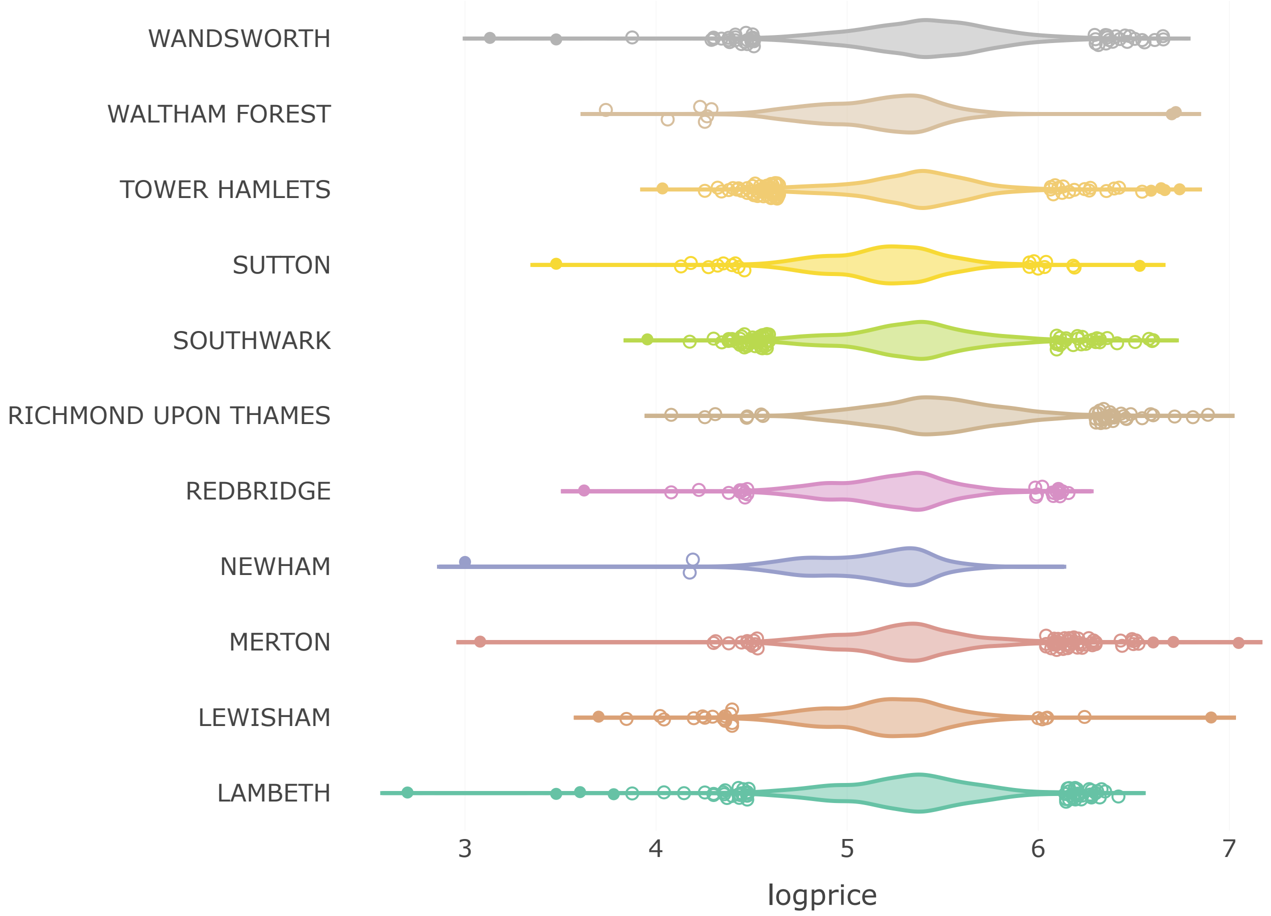
```
fig_gk
```

```
## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2
is 8
## Returning the palette you asked for with that many colors

## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2
is 8
## Returning the palette you asked for with that many colors
```



```
fig_lz
```

```
## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2
is 8
## Returning the palette you asked for with that many colors

## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2
is 8
## Returning the palette you asked for with that many colors
```



A few observations include that Kensington and Chelsea seem to have the highest home prices, while Greenwich, Hackney, and Fulham in particular have some outliers. There is one house in Hackney that is listed at zero and, which is peculiar, but perhaps it was given away.

#A2

Floor level is not obviously indicated in the dataframe, however SAON does indicate floor level in some features.

So the entire dataset could not be used to estimate the relationship between price of flats and floor level, but those that have floors indicated can be used.

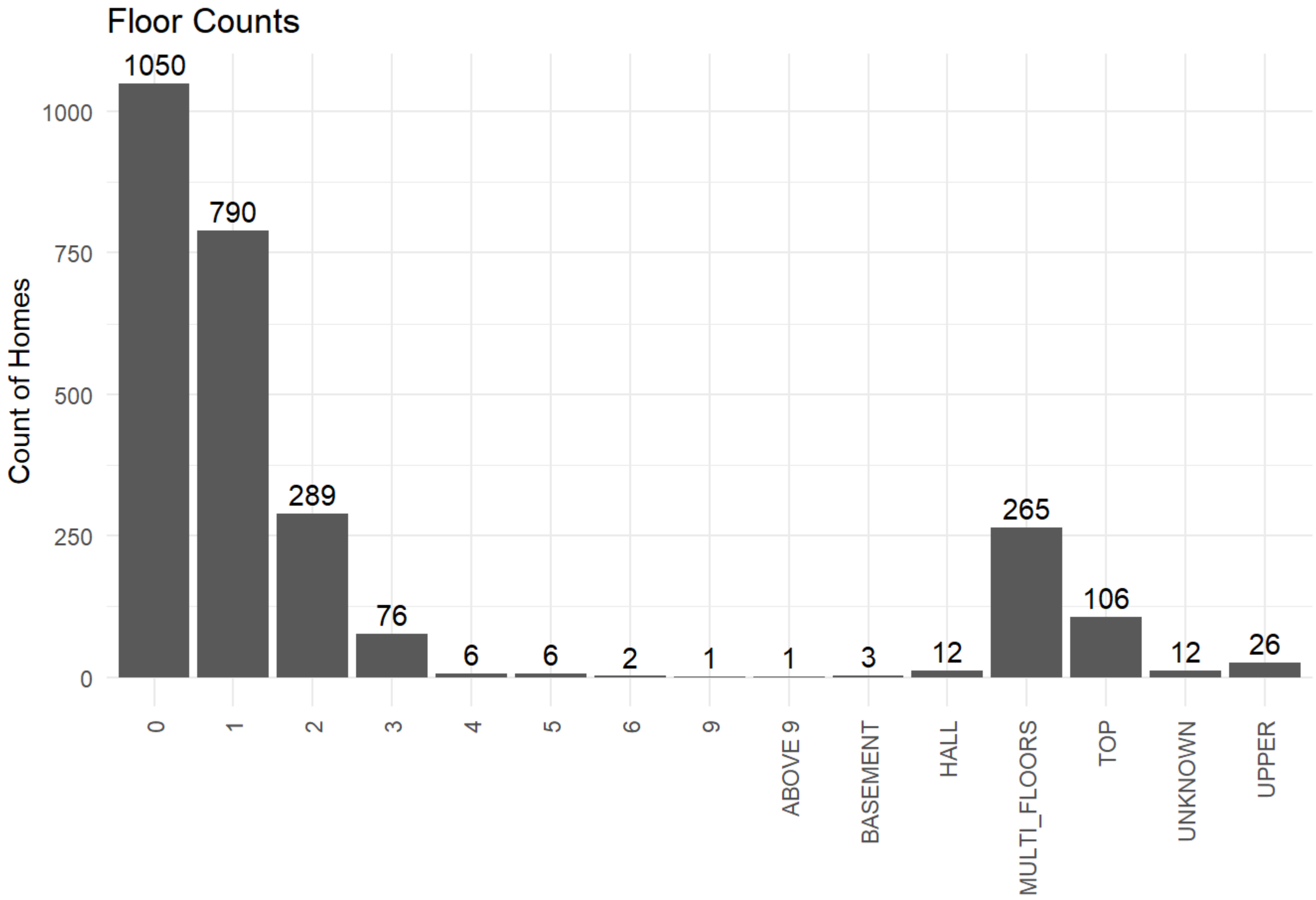
```
#Filter data to only include those with floors listed
floors_df <- filter(df, grepl("FLOOR", SAON, ignore.case = TRUE))

#Look at unique floor levels
unique(floors_df$SAON)
```

```
##   [1] "FIRST FLOOR FLAT"
##   [2] "FIRST AND SECOND FLOOR FLAT"
##   [3] "SECOND FLOOR FLAT 2"
##   [4] "GROUND FLOOR FLAT"
##   [5] "TOP FLOOR FLAT"
##   [6] "GROUND AND FIRST FLOOR FLAT"
##   [7] "LOWER GROUND AND GROUND FLOORS"
##   [8] "SECOND AND THIRD FLOOR FLAT"
##   [9] "SECOND FLOOR FLAT"
##  [10] "FIRST FLOOR"
##  [11] "BASEMENT AND GROUND FLOOR FLAT"
##  [12] "FIRST AND SECOND FLOOR GRAND APARTMENT"
##  [13] "THIRD FLOOR SUITE"
##  [14] "THE GROUND FLOOR FLAT"
##  [15] "SECOND AND THIRD FLOOR MAISONETTE"
##  [16] "GROUND FLOOR FLAT AT"
##  [17] "THIRD FLOOR FLAT"
##  [18] "SECOND FLOOR FLAT AT"
##  [19] "SECOND AND THIRD FLOOR MAISIONETTE"
##  [20] "FIRST FLOOR FLAT 2"
##  [21] "THE GROUND FLOOR FLAT AT"
##  [22] "LOWER GROUND FLOOR FLAT"
##  [23] "GROUND AND LOWER GROUND FLOOR FLAT"
##  [24] "GROUND FLOOR OFFICES"
##  [25] "UPPER FLOOR"
##  [26] "PART GROUND FLOOR AND BASEMENT"
##  [27] "GROUND AND FIRST FLOOR MAISONETTE"
##  [28] "GROUND FLOOR FLAT 1"
##  [29] "HALL FLOOR FLAT"
##  [30] "GROUND FLOOR MAISONETTE"
##  [31] "SECOND FLOOR PREMISES"
##  [32] "FIRST FLOOR ROOMS"
##  [33] "FIRST AND SECOND FLOORS MAISONETTE"
##  [34] "GROUND FLOOR AND BASEMENT"
##  [35] "GROUND FLOOR SHOP"
##  [36] "SECOND FLOOR"
##  [37] "THE SECOND FLOOR FLAT AT"
##  [38] "THIRD AND FOURTH FLOOR FLAT"
##  [39] "THE FIRST FLOOR FLAT AT"
##  [40] "RAISED GROUND FLOOR FLAT"
##  [41] "FIRST FLOOR FLAT AT"
##  [42] "LOWER GROUND FLOOR FLAT B"
##  [43] "FOURTH FLOOR FLAT"
##  [44] "THE SECOND FLOOR FLAT"
##  [45] "GROUND FLOOR FLAT 2"
##  [46] "GROUND FLOOR FLAT 3"
##  [47] "FIRST FLOOR FLAT 2E"
##  [48] "GROUND FLOOR"
##  [49] "SECOND FLOOR FLAT 3"
##  [50] "THE LOWER GROUND FLOOR AT"
##  [51] "FIRST FLOOR FLAT 1"
##  [52] "NINTH FLOOR APARTMENT 19"
##  [53] "GARDEN FLOOR FLAT 3"
##  [54] "SECOND FLOOR FLAT 9"
##  [55] "PART OF LOWER GROUND AND GROUND FLOORS"
##  [56] "THE SECOND AND THIRD FLOOR MAISONETTE AT"
##  [57] "FLAT 7 FIFTH FLOOR"
##  [58] "FIFTH FLOOR"
##  [59] "FRONT GROUND FLOOR FLAT"
##  [60] "LOWER GROUND FLOOR"
##  [61] "LOWER AND GROUND FLOOR FLAT"
##  [62] "FIRST FLOOR FLAT B"
##  [63] "LOWER GROUND FLOOR FLAT 1"
##  [64] "GROUND FLOOR SHOP PREMISES"
##  [65] "UPPER FLOOR FLAT"
##  [66] "SECOND FLOOR FLAT 1"
##  [67] "LOWER FLOOR FLAT"
##  [68] "UPPER GROUND FLOOR FLAT"
##  [69] "THE GROUND AND FIRST FLOOR FLAT 1"
##  [70] "THE LOWER GROUND FLOOR FLAT"
##  [71] "THE TOP FLOOR"
##  [72] "GARDEN FLOOR FLAT"
##  [73] "GROUND FIRST AND SECOND FLOOR"
##  [74] "FIRST AND SECOND FLOOR MAISONETTE AT"
##  [75] "FLAT 3 SECOND FLOOR"
##  [76] "GROUND, MEZZANINE AND FIRST FLOORS"
##  [77] "THE FIRST AND SECOND FLOOR FLAT"
##  [78] "MIDDLE FLOOR FLAT"
##  [79] "REAR GROUND FLOOR FLAT"
##  [80] "GROUND FLOOR FLAT AND BIN SPACE"
##  [81] "LOWER GROUND FLOOR FLAT 2"
##  [82] "GROUND FLOOR PREMISES"
##  [83] "THE FIRST AND SECOND FLOOR FLAT AT"
##  [84] "GROUND FLOOR FLAT CAMERON HOUSE"
##  [85] "THIRD AND FOURTH FLOOR MAISONETTE"
##  [86] "BASEMENT AND GROUND FLOOR FLATS"
##  [87] "BASEMENT AND GROUND FLOOR RETAIL UNITS"
##  [88] "FIRST FLOOR OFFICE SUITE"
##  [89] "FLAT 1 (GROUND FLOOR)"
##  [90] "LOWER GROUND FLOOR APARTMENT"
##  [91] "GROUND FLOOR FLAT 5"
##  [92] "FIRST FLOOR REAR FLAT"
##  [93] "FIRST AND SECOND FLOORS"
##  [94] "FIRST FLOOR TOILET"
##  [95] "TOP FLOOR"
##  [96] "THE FIRST FLOOR FLAT"
##  [97] "THE FIRST FLOOR FLAT BEING FLAT 2"
##  [98] "SECOND FLOOR OFFICES"
##  [99] "GROUND FLOOR COMMERCIAL UNIT"
## [100] "FIRST AND SECOND FLOOR OFFICES"
## [101] "GROUND & LOWER FLOOR FLAT"
## [102] "BASEMENT AND GROUND FLOOR SHOP"
## [103] "SECOND & THIRD FLOOR FLAT 4"
## [104] "LOWER GROUND AND GROUND FLOOR FLAT"
## [105] "FLAT 3 (FIRST FLOOR)"
## [106] "FLAT 19 THIRD FLOOR"
## [107] "BASEMENT AND GROUND FLOOR PREMISES"
## [108] "THE GROUND FLOOR FLAT BEING FLAT 6"
## [109] "FIRST AND SECOND FLOOR MAISONETTE"
## [110] "THIRD FLOOR FLAT 7"
## [111] "FIRST AND SECOND FLOOR"
## [112] "THIRD FLOOR FLAT E"
## [113] "BOTTOM FLOOR FLAT"
## [114] "FIRST AND SECOND FLOOR FLAT, UNIT 10"
## [115] "THE TOP FLOOR FLAT AT"
## [116] "GROUND FIRST AND SECOND FLOOR FLAT"
## [117] "FIRST FLOOR MAISONETTE"
## [118] "LOWER GROUND AND UPPER GROUND FLOOR FLAT"
## [119] "BASEMENT AND GROUND FLOOR FLAT B"
## [120] "FIRST FLOOR FLAT 3"
## [121] "LOWER FLOOR MAISONETTE"
## [122] "GROUND AND FIRST FLOORS"
## [123] "THE THIRD FLOOR FLAT"
## [124] "GROUND FLOOR FLAT AT 1"
## [125] "FIRST FLOOR FLAT A"
## [126] "GROUND FLOOR APARTMENT"
## [127] "BASEMENT FLOOR FLAT"
## [128] "LOWER FLOOR"
## [129] "SIXTH FLOOR APARTMENT 4"
## [130] "FIRST FLOOR FLAT 4"
## [131] "TOP FLOOR FLAT A"
## [132] "SECOND THIRD AND FOURTH FLOOR FLAT"
## [133] "FIRST AND SECOND FLOOR APARTMENT"
## [134] "THIRD FLOOR STUDIO FLAT"
## [135] "FIFTH FLOOR FLAT 5"
## [136] "GROUND FLOOR FLAT 145"
## [137] "SECOND  FLOOR FLAT"
## [138] "GROUND FLOOR AND BASEMENT SHOP"
## [139] "2ND FLOOR APARTMENT"
## [140] "BASEMENT TO THIRD FLOOR PREMISES"
## [141] "FLAT 2 FIRST FLOOR"
## [142] "TOP FLOOR FLAT AT"
## [143] "FOURTH AND FIFTH FLOOR FLAT"
## [144] "FIRST AND SECOND FLOOR FLAT AT"
## [145] "SECOND FLOOR FLAT 4"
## [146] "GROUND FLOOR SHOP AND BASEMENT PREMISES"
## [147] "FIRST FLOOR MAISONETTE AT"
## [148] "1ST & 2ND FLOOR FLAT"
## [149] "1ST FLOOR FLAT"
## [150] "BASEMENT AND GROUND FLOOR"
## [151] "GROUND FLOOR SHOP AND BASEMENT FLAT"
## [152] "GROUND FLOOR FLAT,"
## [153] "BEING TOP FLOOR FLAT AT"
## [154] "APARTMENT 6 GROUND FLOOR"
## [155] "SECOND AND THIRD FLOOR FLAT 3"
## [156] "FLAT 4 LOWER GROUND FLOOR"
## [157] "FIRST AND SECOND FLOOR 2"
## [158] "FIRST FLOOR FLAT 1 ST JOHNS HOUSE"
## [159] "GROUND FLOOR FRONT FLAT 1"
## [160] "FIRST FLOOR FLAT C"
## [161] "GROUND FLOOR FLAT A"
## [162] "GROUND FLOOR AND BASEMENT PREMISES"
## [163] "THE GROUND FLOOR MAISONETTE AT"
## [164] "THE FRONT FIRST FLOOR FLAT AT"
## [165] "THE FIRST AND SECOND FLOOR MAISONETTE AT"
## [166] "GROUND FLOOR REAR FLAT"
## [167] "21A GROUND FLOOR FLAT"
## [168] "THE SECOND AND THIRD FLOOR FLAT"
## [169] "FIRST  FLOOR FLAT"
## [170] "GROUND FLOOR AND LOWER GROUND FLOOR"
## [171] "GROUND FLOOR MAISONETTE AT"
## [172] "LOWER GROUND FLOOR FLAT FRONT"
## [173] "FIRST FLOOR APARTMENT"
## [174] "GROUND FLOOR FLAT FRONT"
## [175] "GROUND FLOOR STUDIO FLAT"
## [176] "GROUND FLOOR FLAT (FRONT)"
## [177] "GROUND FLOOR FLAT AND BASEMENT"
## [178] "FIRST FLOOR FRONT"
## [179] "APARTMENT A, SIXTH FLOOR"
## [180] "THIRD  FLOOR FLAT"
## [181] "GROUND AND FIRST FLOOR FLAT 4"
## [182] "SECOND FLOOR FLAT 8"
## [183] "SECOND AND THIRD FLOOR MAISONETTE AT"
## [184] "FIRST FLOOR FLAT 6"
## [185] "FIRST AND SECOND FLOOR MAISONETTE 1"
## [186] "GROUND AND FIRST FLOOR FLAT AT"
## [187] "FIRST FLOOR FLAT 5"
## [188] "BASEMENT, GROUND AND FIRST FLOOR FLAT"
## [189] "THIRD FLOOR FLAT 5"
## [190] "GROUND FLOOR FLAT B"
## [191] "FIRST SECOND & THIRD FLOORS & LOFT SPACE"
## [192] "THE LOWER GROUND FLOOR FLAT AT"
## [193] "UPPER FLOORS"
## [194] "FIFTH FLOOR ROOF PREMISES"
## [195] "2ND FLOOR FLAT"
## [196] "SECOND AND TOP FLOOR FLAT 4"
## [197] "BASEMENT AND GROUND FLOOR MAISONETTE 1"
## [198] "FLAT D 2ND FLOOR"
## [199] "RIGHT GROUND FLOOR FRONT FLAT"
## [200] "FLAT 23 THIRD FLOOR"
## [201] "FIRST FLOOR FLAT AND MEZZANINE FLOOR"
## [202] "THIRD FLOOR"
## [203] "LOWER GROUND FLOOR FLAT 3"
## [204] "FIRST & SECOND FLOOR FLAT"
## [205] "FIRST FLOOR AND ATTIC"
## [206] "FLAT NUMBER 5 ON THE FIRST FLOOR"
## [207] "FIRST FLOOR 172"
## [208] "THIRD FLOOR FLAT 9"
## [209] "BASEMENT AND GROUND FLOOR MAISONETTE"
## [210] "PART FIRST FLOOR"
## [211] "FOURTH FLOOR"
## [212] "TOP FLOOR FLAT 1"
## [213] "UPPER FLOOR MAISONETTE"
## [214] "FIRST FLOOR FLAT AT A"
## [215] "TENTH FLOOR FLAT 5"
## [216] "GROUND FLOOR NORTH OFFICE"
## [217] "FLAT 4 FIFTH FLOOR"
## [218] "GROUND FLOOR FLAT 4"
## [219] "TENTH AND ELEVENTH FLOOR APARTMENT P2"
## [220] "PART OF FIRST FLOOR"
## [221] "FIRST, SECOND AND THIRD FLOOR FLAT"
## [222] "FIRST, SECOND AND THIRD FLOORS"
## [223] "FIRST SECOND AND THIRD FLOOR FLAT"
## [224] "SECOND AND ATTIC FLOOR FLAT"
## [225] "LOWER FLOOR FLAT AT"
## [226] "FIRST FLOOR FRONT FLAT"
## [227] "GROUND FLOOR FRONT FLAT"
## [228] "GROUND FLOOR  STORAGE UNIT"
## [229] "TOP FLOOR FLAT 5"
## [230] "LOWER GROUND AND GROUND FLOOR MAISONETTE"
## [231] "STORE ROOM AT GROUND FLOOR SHOP"
## [232] "FIFTH AND SIXTH FLOOR"
## [233] "LOWER GROUND FLOOR FLAT & ENTRANCE HALL"
## [234] "GARDEN FLOOR FLAT 12"
## [235] "FIRST & SECOND FLOOR MAISONETTE"
## [236] "SECOND FLOOR FLAT AND LOFT"
## [237] "LOWER AND RAISED GROUND FLOORS"
## [238] "GROUND FLOOR FLAT AND GARDEN GROUND"
## [239] "FIRST FLOOR 3"
## [240] "SECOND FLOOR COMMERCIAL UNIT"
## [241] "GROUND & FIRST FLOOR FLAT"
## [242] "THE BASEMENT AND GROUND FLOOR FLAT AT"
## [243] "GROUND FLOOR AND BASEMENT FLAT"
## [244] "GARDEN FLOOR FLAT AT"
## [245] "LOWER GROUND FLOOR REAR FLAT"
## [246] "SECOND FLOOR FLAT C"
## [247] "GROUND AND FIRST FLOOR UNIT 5"
## [248] "GROUND AND FIRST FLOOR"
## [249] "FIRST FLOOR REAR FLAT 2"
## [250] "GROUND FLOOR SUITE"
## [251] "THE GROUND AND BASEMENT FLOOR FLAT AT"
## [252] "FLAT 19 FOURTH FLOOR"
## [253] "LOWER GROUND AND GROUND FLOOR FLAT 3"
## [254] "TOP FLOOR FLAT 6"
## [255] "SECOND AND THIRD FLOOR  FLAT"
## [256] "SECOND FLOOR FLAT B"
## [257] "LOWER GROUND FLOOR FLAT AND GARDEN"
## [258] "THE GARDEN FLOOR FLAT AT"
## [259] "THIRD AND FOURTH FLOOR FLAT AND STAIRS"
## [260] "THE SECOND AND THIRD FLOOR MAISONETTE"
## [261] "GROUND FLOOR REAR FLAT 2"
## [262] "THIRD FLOOR REAR FLAT"
## [263] "FLAT 25 FIRST FLOOR"
## [264] "GROUND TO THIRD FLOORS AND ROOF"
```

There are a lot of ways I could categorize, I walk through the process below.

```
#Create new column with Unknown as the bast
floors_df$floor <- "UNKNOWN"

#Use grep to auto fill in columns based on substrings
#I commented out floors 7-10 as they
floors_df$floor[grep("BASEMENT", floors_df$SAON)] <- "BASEMENT"
floors_df$floor[grepl("GROUND", floors_df$SAON)]<- 0
floors_df$floor[grepl("GARDEN", floors_df$SAON)]<- 0
floors_df$floor[grepl("FIRST", floors_df$SAON)]<- 1
floors_df$floor[grepl("1ST", floors_df$SAON)]<- 1
floors_df$floor[grepl("MEZZANINE", floors_df$SAON)]<- 'MEZZANINE'
floors_df$floor[grepl("SECOND", floors_df$SAON)]<- 2
floors_df$floor[grepl("2ND", floors_df$SAON)]<- 2
floors_df$floor[grepl("THIRD", floors_df$SAON)]<- 3
floors_df$floor[grepl("3RD", floors_df$SAON)]<- 3
floors_df$floor[grepl("FOURTH", floors_df$SAON)]<- 4
floors_df$floor[grepl("4TH", floors_df$SAON)]<- 4
floors_df$floor[grepl("FIFTH", floors_df$SAON)]<- 5
floors_df$floor[grepl("5TH", floors_df$SAON)]<- 5
floors_df$floor[grepl("SIXTH", floors_df$SAON)]<- 6
floors_df$floor[grepl("6TH", floors_df$SAON)]<- 6
floors_df$floor[grepl("SEVENTH", floors_df$SAON)]<- 7
floors_df$floor[grepl("7TH", floors_df$SAON)]<- 7
floors_df$floor[grepl("EIGHTH", floors_df$SAON)]<- 8
floors_df$floor[grepl("8TH", floors_df$SAON)]<- 8
floors_df$floor[grepl("NINTH", floors_df$SAON)]<- 9
floors_df$floor[grepl("9TH", floors_df$SAON)]<- 9
floors_df$floor[grepl("TENTH", floors_df$SAON)]<- 'ABOVE 9'
floors_df$floor[grepl("10TH", floors_df$SAON)]<- 'ABOVE 9'
floors_df$floor[grepl("ELEVENTH", floors_df$SAON)]<- 'ABOVE 9'
floors_df$floor[grepl("11TH", floors_df$SAON)]<- 'ABOVE 9'
floors_df$floor[grepl("TOP", floors_df$SAON)]<- 'TOP'
floors_df$floor[grepl("PENTHOUSE", floors_df$SAON)]<- 'PENTHOUSE'
floors_df$floor[grepl("UPPER", floors_df$SAON)]<- 'UPPER'
floors_df$floor[grepl("HALL", floors_df$SAON)]<- 'HALL'
floors_df$floor[grepl("AND", floors_df$SAON)]<- 'MULTI_FLOORS'
```

```
#First we inspect frequencies of each observation
ggplot(floors_df, aes(x=floor))+
geom_bar()+
theme_minimal() +
 theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
        ggtitle("Floor Counts")+
        geom_text(aes(label = ..count..), stat = "count", position = position_dodge(width = 0.9
), vjust=-0.40)+
        xlab("Number of Floors") + ylab("Count of Homes")
```

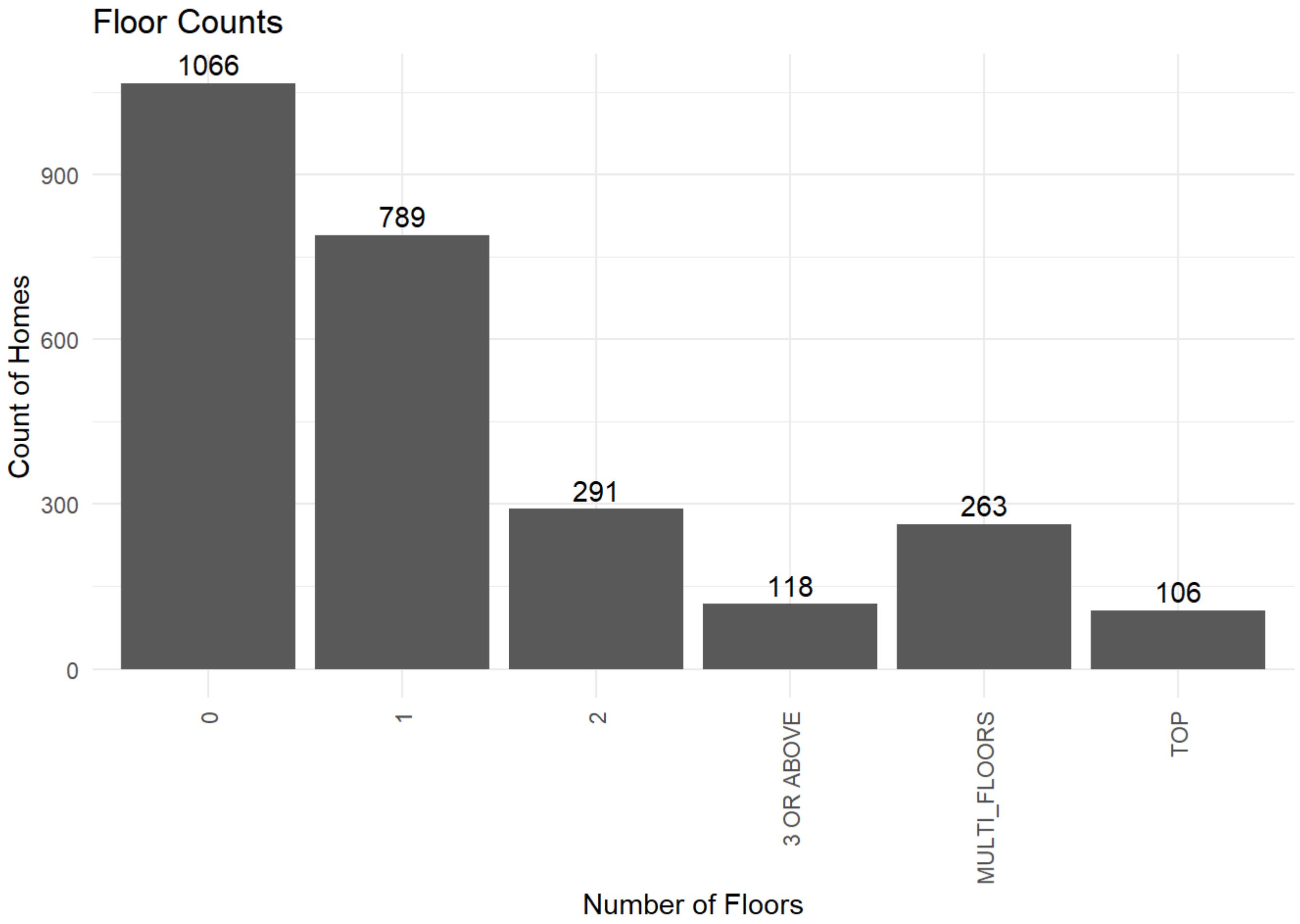


Clearly I have an error in the above and need bin the data further.

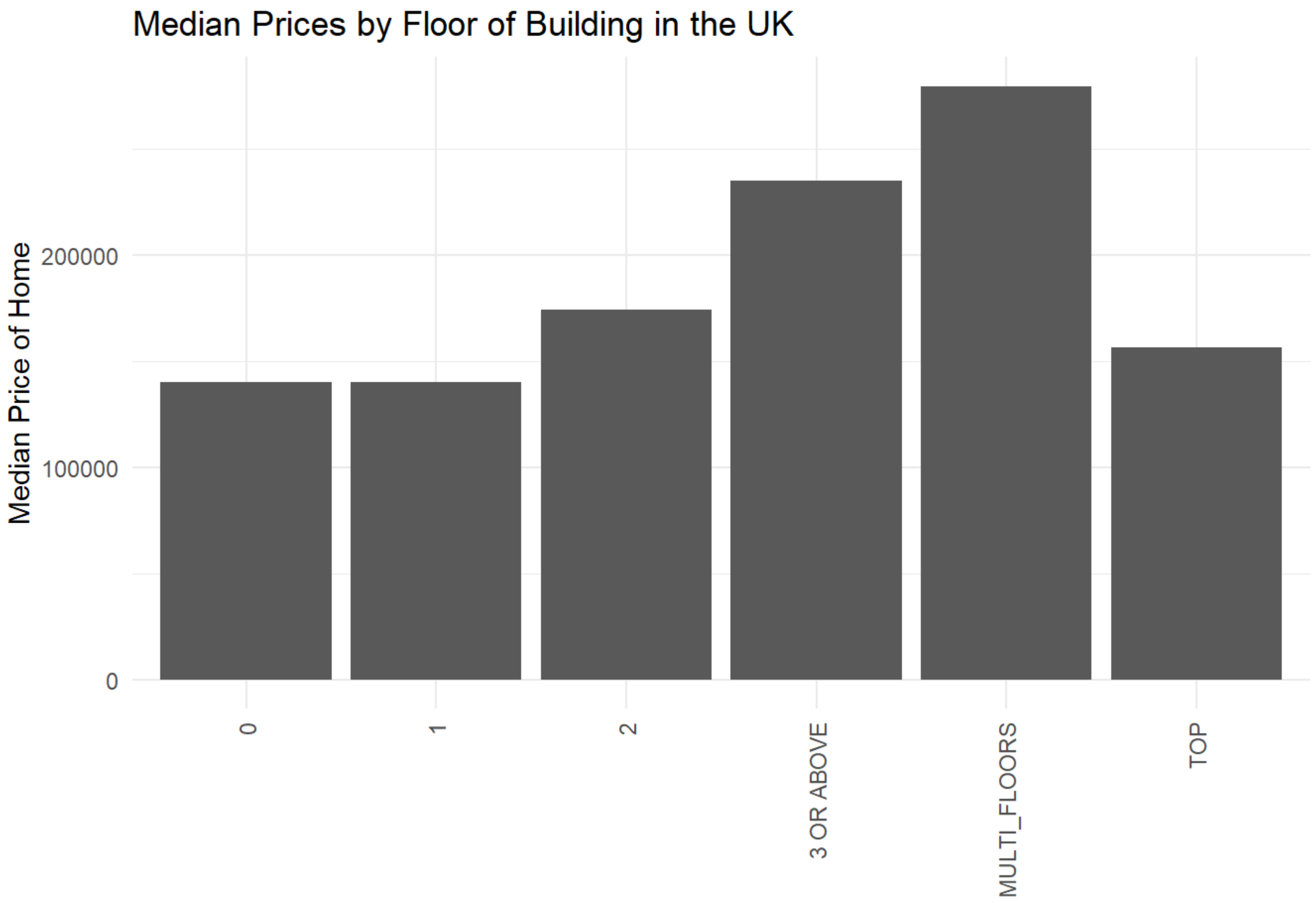
```
floors_df$floor[grep("BASEMENT", floors_df$SAON)] <- 0 #Given so few baseement observations I gr
ouped with ground
floors_df$floor[grepl("GROUND", floors_df$SAON)]<- 0
floors_df$floor[grepl("GARDEN", floors_df$SAON)]<- 0
floors_df$floor[grepl("THIRD", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("3RD", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("FOURTH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("4TH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("FIFTH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("5TH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("SIXTH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("6TH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("SEVENTH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("7TH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("EIGHTH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("8TH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("NINTH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("9TH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("TENTH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("10TH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("ELEVENTH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("11TH", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("TOP", floors_df$SAON)]<- 'TOP'
floors_df$floor[grepl("PENTHOUSE", floors_df$SAON)]<- 'TOP'
floors_df$floor[grepl("UPPER", floors_df$SAON)]<- '3 OR ABOVE'
floors_df$floor[grepl("HALL", floors_df$SAON)]<- '0'
floors_df$floor[grepl("AND", floors_df$SAON)]<- 'MULTI_FLOORS'
floors_df$floor[grepl("MEZZANINE", floors_df$SAON)]<- 2


#Drop any that couldn't be characterized
floors_df <- floors_df[!grepl("UNKNOWN", floors_df$floor),]
```

Now inspect new frequencies

```
#First we inspect frequencies of each observation
ggplot(floors_df, aes(x=floor))+
geom_bar()+
theme_minimal() +
 theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
        ggtitle("Floor Counts")+
        geom_text(aes(label = ..count..), stat = "count", position = position_dodge(width = 0.9
), vjust=-0.40)+
        xlab("Number of Floors") + ylab("Count of Homes")
```



```
#Now I plot with median values to account for outliers
options(scipen=1000)

ggplot(floors_df, aes(x=floor, y=price))+
  geom_bar(stat="summary", fun ="median")+
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  ggtitle("Median Prices by Floor of Building in the UK")+
  xlab("Number of Floors") + ylab("Median Price of Home")
```

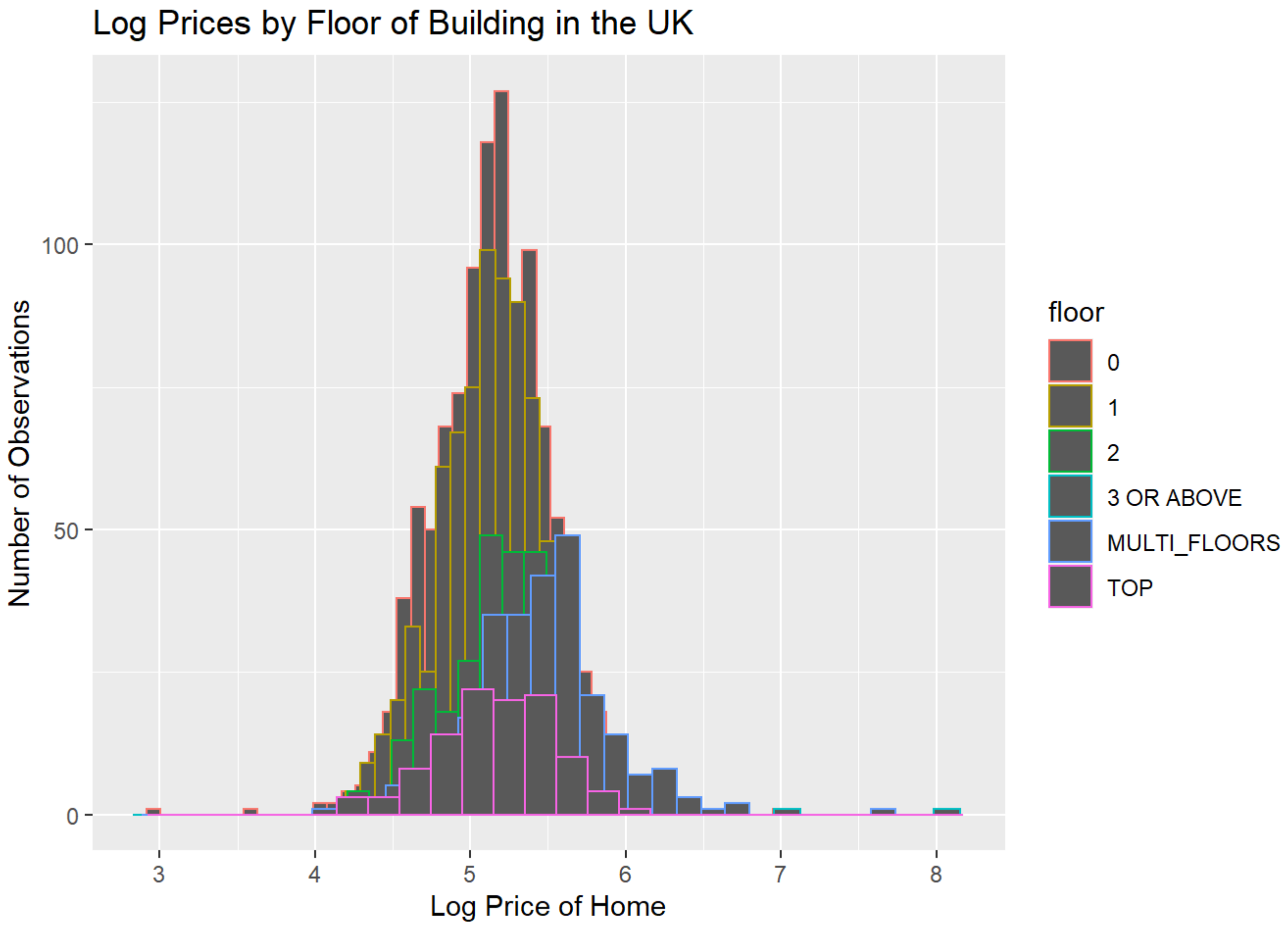


Now that I have fewer categories, I can inspect the distribution of plots using a custom function.

The below binning function I got from the tidyverse website, and is “particularly useful when faceting along variables with different ranges”.

```
ggplot(floors_df, aes(logprice, color=floor)) +
  #facet_grid(~floor, scales = 'fixed') +
  geom_histogram(binwidth = function(x) 2 * IQR(x) / (length(x)^(1/3)))+
  ggtitle("Log Prices by Floor of Building in the UK")+
  xlab("Log Price of Home") + ylab("Number of Observations")
```

```
## Warning: position_stack requires non-overlapping x intervals
```



I like the above plot because it’s aesthetically pleasing and indicates that multi-floor units tend to have a higher price, but you can see the progression from 0, 1, 2, to 3 or above and multi floors.

I also tried a facet wrap but found the above to be a more efficient way to inspect the floor level and housing prices.

#Task B

```
head(ukpostcodes)
```

```
##   id postcode latitude longitude
## 1  1 AB10 1XG 57.14417 -2.114848
## 2  2 AB10 6RN 57.13788 -2.121487
## 3  3 AB10 7JB 57.12427 -2.127190
## 4  4 AB11 5QN 57.14270 -2.093015
## 5  5 AB11 6UL 57.13755 -2.112696
## 6  6 AB11 8RQ 57.13598 -2.072115
```

```
df <- read.csv('C:/Users/jcsea/OneDrive/Desktop/BGSE Documents/ppdata_liter2.csv')

data <- merge(df, ukpostcodes, by = "postcode")
dim(data)
```

```
## [1] 566391     19
```

```
#Now mutate to create new columns
data <- data %>%
  group_by(postcode) %>%
  #Mutate function creates new variables
  mutate(min = min(price), mean = mean(price), median = median(price), max = max(price))
```

Get high medium and low categories by splitting evenly into 3 groups. Used Hmisc library for this

```
data$Price_Category <- as.numeric(cut2(data$median, g=3))
```

Method 1

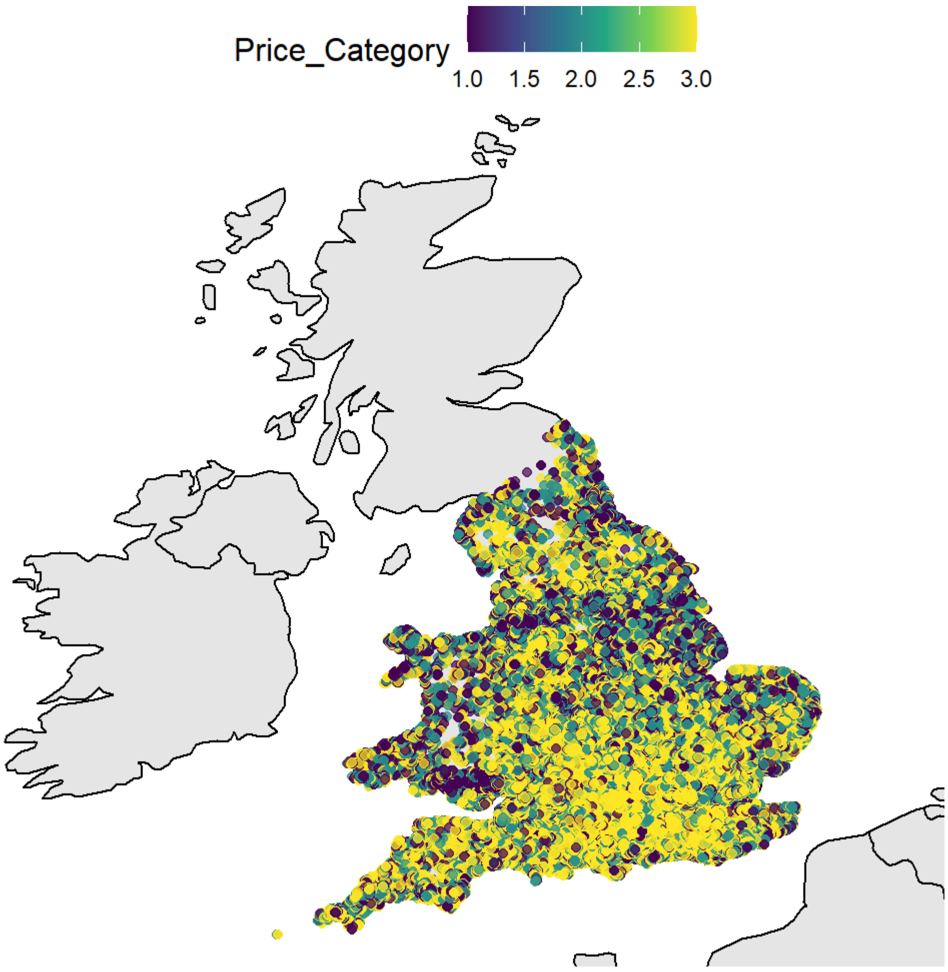
I created a spatial dataframe and plotted all of the houses based on price.

I used several sources to guide me, but this one that looks at UK geospatial data was the one I used most closely.

```
#Get a Spatial Dataframe. #19, 20 are locations of latitude and longitude
sdf <- SpatialPointsDataFrame(data[,18:19], data)

plot(sdf, main="Map of House Locations")
```

Map of House Locations

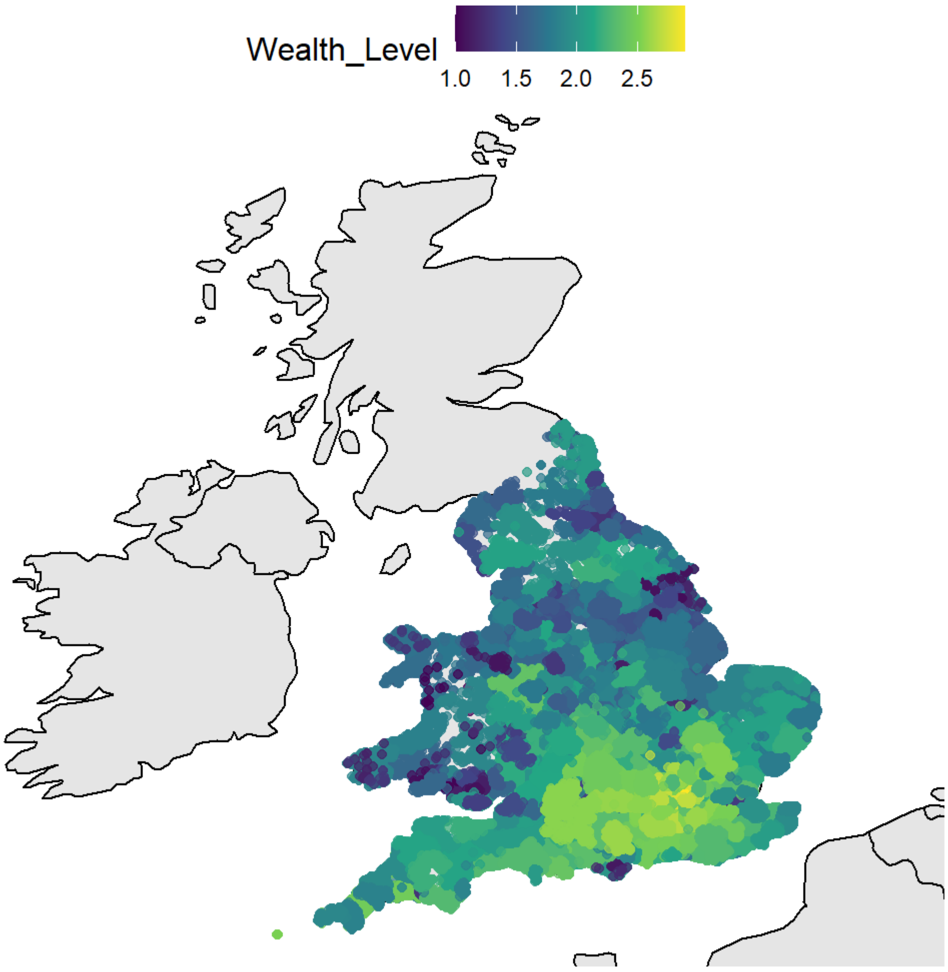
```
#

#Get world map from maps library
worldmap = map_data('world')

#Tweak to just get map of UK
ggplot() +
  geom_polygon(data = worldmap,
               aes(x = long, y = lat, group = group),
               fill = 'gray90', color = 'black') +
  coord_fixed(ratio = 1.3, xlim = c(-10,3), ylim = c(50, 59)) +
  theme_void() +
  geom_point(data = data,
             aes(x = longitude,
                 y = latitude, color = Price_Category), alpha = .7) +
  scale_size_area(max_size = 8) +
  scale_color_viridis_c() +
  theme(legend.position = 'top') +
  theme(title = element_text(size = 12))
```



Here

I improve upon this

```
agg <- data %>%
  group_by(district) %>%
  summarise_at(vars(Price_Category), list(Wealth_Level = mean))

agg2 <- merge(data, agg, by = "district")

ggplot() +
  geom_polygon(data = worldmap,
               aes(x = long, y = lat, group = group),
               fill = 'gray90', color = 'black') +
  coord_fixed(ratio = 1.3, xlim = c(-10,3), ylim = c(50, 59)) +
  theme_void() +
  geom_point(data = agg2,
             aes(x = longitude,
                 y = latitude, color = Wealth_Level), alpha = .7) +
  scale_size_area(max_size = 8) +
  scale_color_viridis_c() +
  theme(legend.position = 'top') +
  theme(title = element_text(size = 12))
```



To get the geojson I can use the code from the starter notebook. I ended up not needing it.

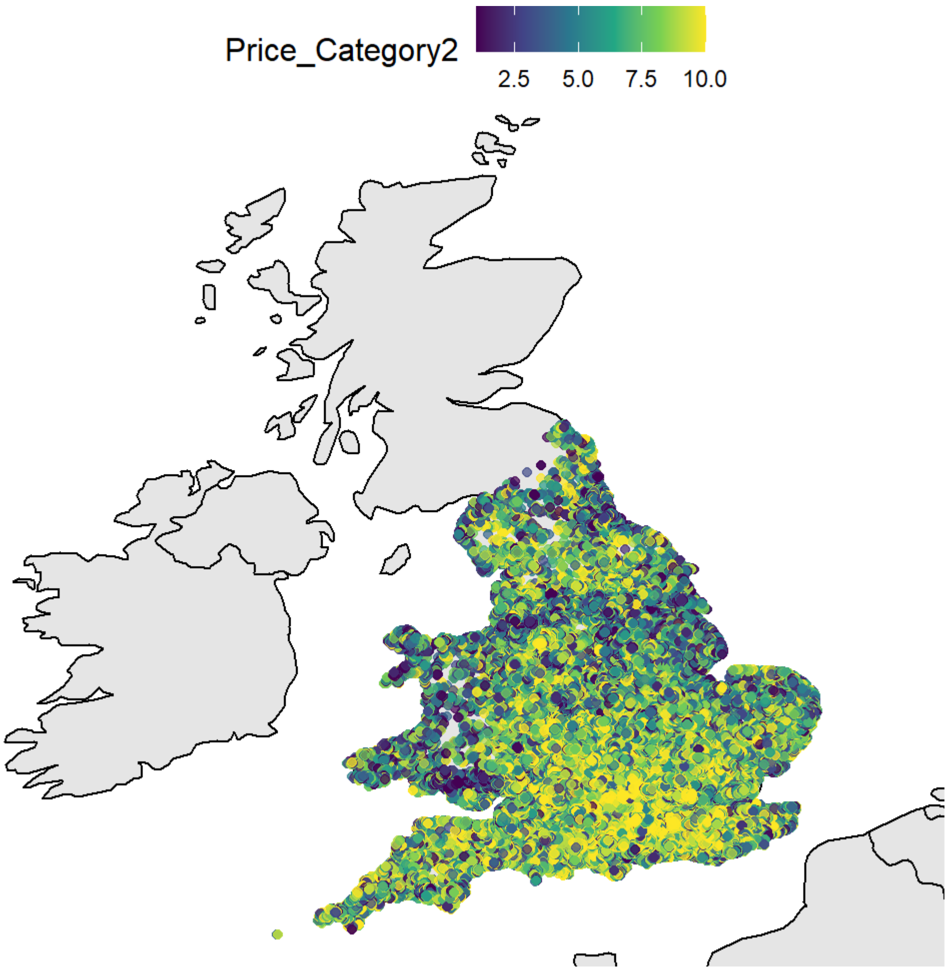
```
#From starter notebook
#coordinates(data) <- c("latitude", "longitude")
#writeOGR(data, "test_geojson.geojson", layer = "data", driver = "GeoJSON")
```

B3

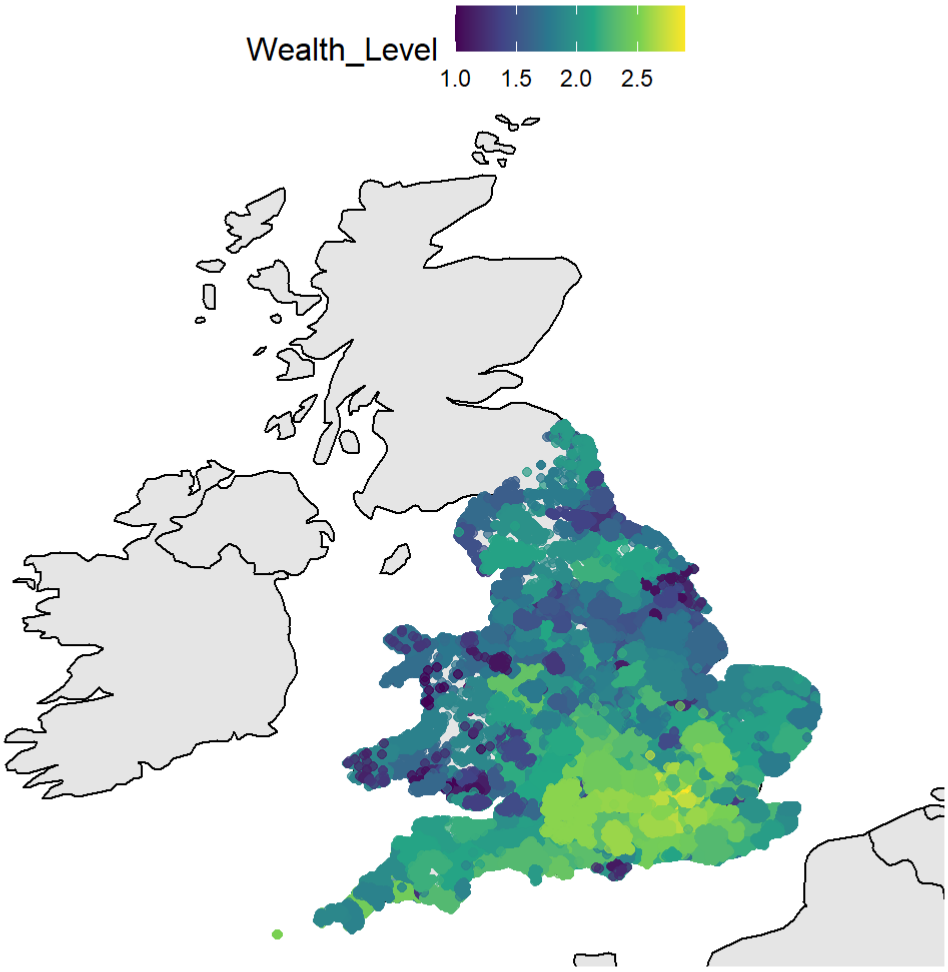
I don’t think mean would have improved the plot, because as we saw some of the housing prices are enormous outliers, over 100 milllion pounds. This would heavily skew the mean housing prices.

We could remove the outliers and then plot housing prices, or plot log house prices as I have done elsewhere in this notebook, to plot it. Given splitting it into 3 equal categories will produce roughly the same map regardless of outliers, I could also use 10 categories here to see how the map looks different.

```
data$Price_Category2 <- as.numeric(cut2(data$mean, g=10))

ggplot() +
  geom_polygon(data = worldmap,
               aes(x = long, y = lat, group = group),
               fill = 'gray90', color = 'black') +
  coord_fixed(ratio = 1.3, xlim = c(-10,3), ylim = c(50, 59)) +
  theme_void() +
  geom_point(data = data,
             aes(x = longitude,
                 y = latitude, color = Price_Category2), alpha = .7) +
  scale_size_area(max_size = 8) +
  scale_color_viridis_c() +
  theme(legend.position = 'top') +
  theme(title = element_text(size = 12))
```



```
agg3 <- data %>%
  group_by(district) %>%
  summarise_at(vars(Price_Category), list(Wealth_Level = median))

agg4 <- merge(data, agg3, by = "district")

ggplot() +
  geom_polygon(data = worldmap,
               aes(x = long, y = lat, group = group),
               fill = 'gray90', color = 'black') +
  coord_fixed(ratio = 1.3, xlim = c(-10,3), ylim = c(50, 59)) +
  theme_void() +
  geom_point(data = agg2,
             aes(x = longitude,
                 y = latitude, color = Wealth_Level), alpha = .7) +
  scale_size_area(max_size = 8) +
  scale_color_viridis_c() +
  theme(legend.position = 'top') +
  theme(title = element_text(size = 12))
```



Task C

#Create new df only consisting of 2015 (recognize this could also be done within ggplot using filter function, however I wanted to inspect the DF too)

```
df <- read.csv('C:/Users/jcsea/OneDrive/Desktop/BGSE Documents/ppdata_liter2.csv')

df$logprice <- log(df$price, 10)
```

First give labels to the property types

```
df$property_type[df$property_type == "D"] <- "Detached"
df$property_type[df$property_type == "S"] <- "Semi-Detached"
df$property_type[df$property_type == "T"] <- "Terraced"
df$property_type[df$property_type == "F"] <- "Flats/Maisonettes"
df$property_type[df$property_type == "O"] <- "Other"
```

Below I do some date transformations.

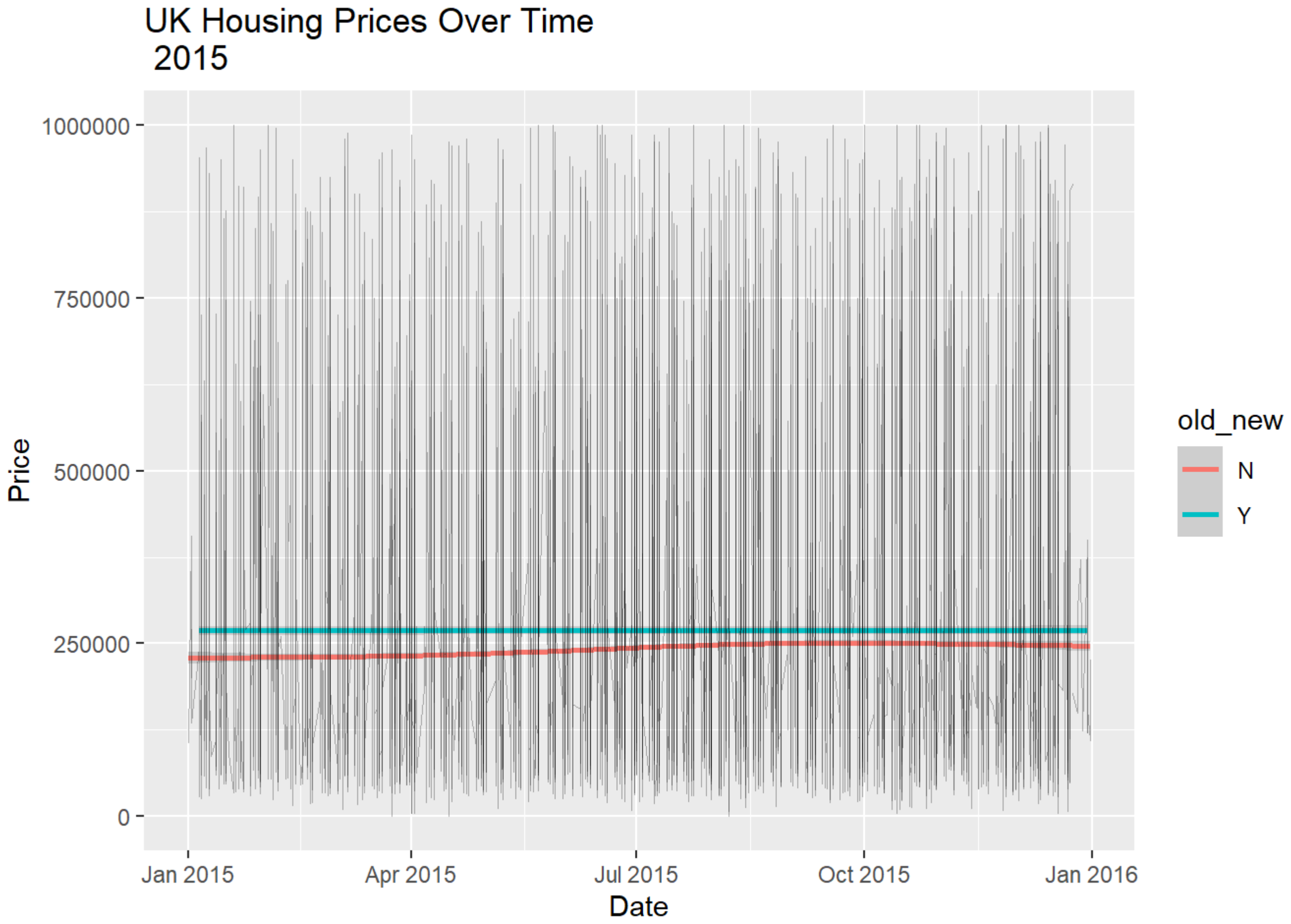
```
#Trim date of transfer column to remove minutes/hours
df$date_of_transfer <- strtrim(df$date_of_transfer,10)

#Turn into dates instead of character
df$date <- as.Date(df$date_of_transfer, "%d/%m/%Y")

#Drop leap day in 1996 that caused null values
#df <- na.omit(df)

#Extract month in case needed
df$month <- as.numeric(format(df$date, "%m"))

#Create season
df$Season[df$month == 12] <- "Winter"
df$Season[df$month == 1] <- "Winter"
df$Season[df$month == 2] <- "Winter"
df$Season[df$month == 3] <- "Spring"
df$Season[df$month == 4] <- "Spring"
df$Season[df$month == 5] <- "Spring"
df$Season[df$month == 6] <- "Summer"
df$Season[df$month == 7] <- "Summer"
df$Season[df$month == 8] <- "Summer"
df$Season[df$month == 9] <- "Autumn"
df$Season[df$month == 10] <- "Autumn"
df$Season[df$month == 11] <- "Autumn"

#Get only 2015 observations. Need to use dplyr:: to avoid a reoccurring error that the object was not found
dC <- dplyr::filter(df, grepl("2015", df$date_of_transfer))

#Define month labels for later
monthlabels <- list('Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec')
```

Create plot with smooth line

```
#Create plot with smooth line
ggplot(dC) +
  geom_smooth(aes(x=date, y=price, color=old_new))+
  theme(text=element_text(size=8))+
  theme_update(plot.title = element_text(hjust = 0.5))+
  ylim(0,1000000)  +
  labs(title="UK Housing Prices Over Time \n 2015",
        x ="Date", y = "Price")+
  geom_line(aes(date, price), alpha = 0.3, size = .1) #Used alpha/size to make thel ines very transparent and small
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

```
## Warning: Removed 526 rows containing non-finite values (stat_smooth).
```



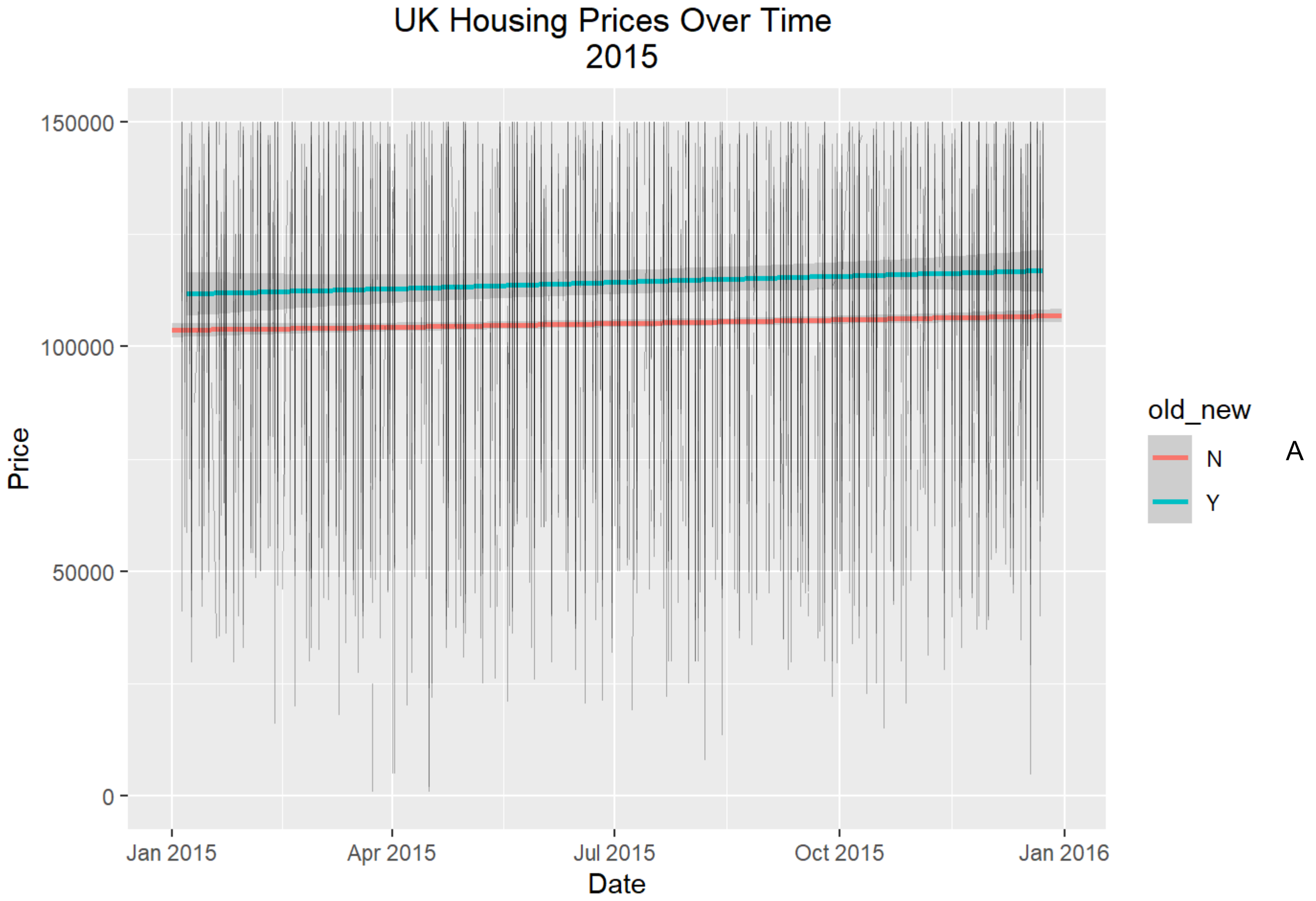
There is not much of a price change over time. It increases slightly. If I decrease the y upper limit we can see a moderate trend

```
#Create plot with y limit
ggplot(dC) +
  geom_smooth(aes(x=date, y=price, color=old_new))+
  theme(text=element_text(size=8))+
  theme_update(plot.title = element_text(hjust = 0.5))+
  ylim(0,150000)  +
  labs(title="UK Housing Prices Over Time \n 2015",
        x ="Date", y = "Price")+
  geom_line(aes(date, price), alpha = 0.3, size = .1)
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

```
## Warning: Removed 18093 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```



you can see in the plot above for houses under 150,000, for new houses the prices have increased slightly more than for old houses, which have stayed relatively flat with a slight upward trend.

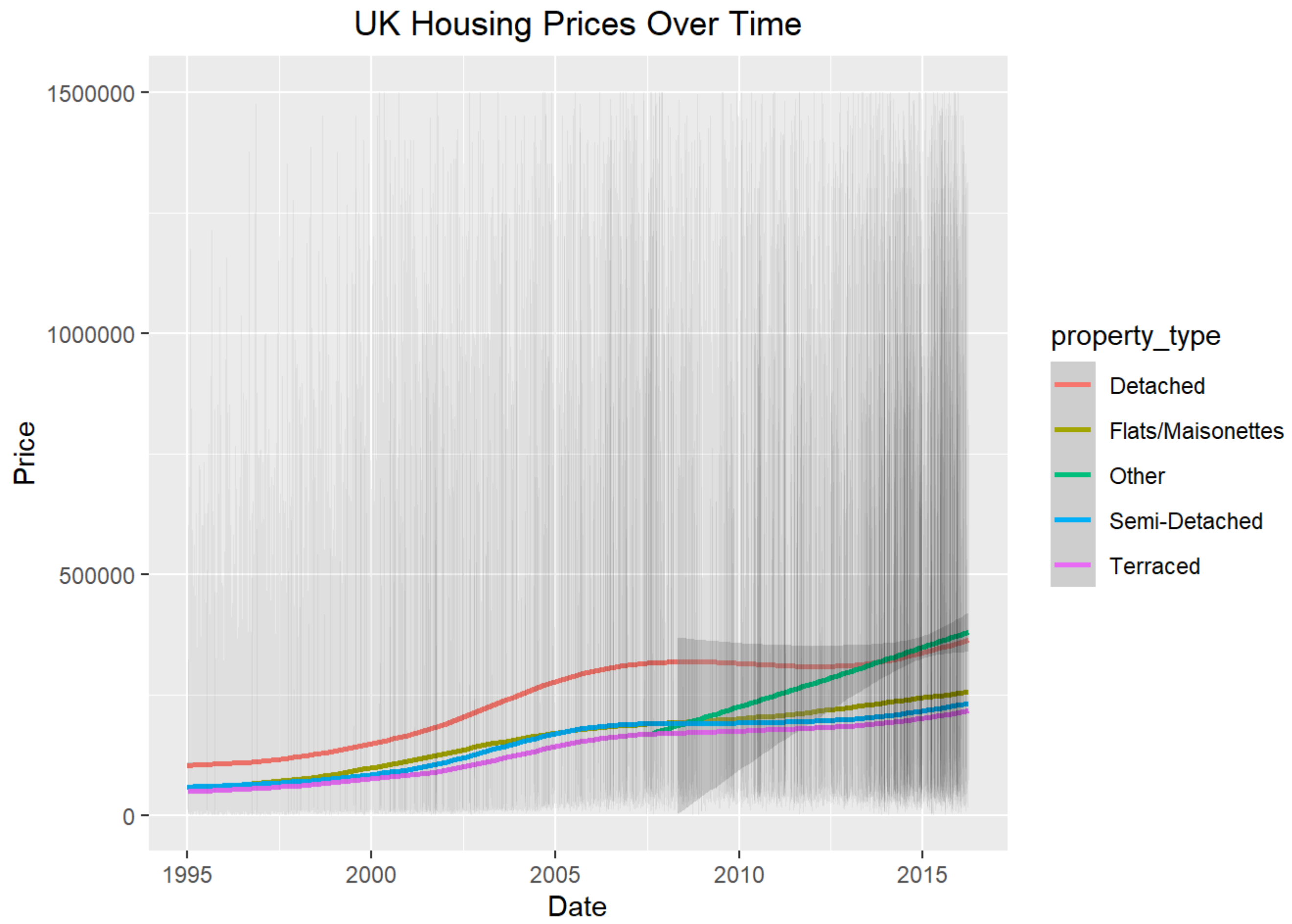
C2

Is there a significant relationship between the price of a property and the time of year it is sold? Does this vary with type of property?

```
options(scipen=10000)
ggplot(df) +
  geom_smooth(aes(x=date, y=price, color=property_type))+
  theme(text=element_text(size=8))+
  theme_update(plot.title = element_text(hjust = 0.5))+
  ylim(0,1500000)  +
  labs(title="UK Housing Prices Over Time",
        x ="Date", y = "Price")+
  geom_line(aes(date, price), alpha = 0.05, size = .05)
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

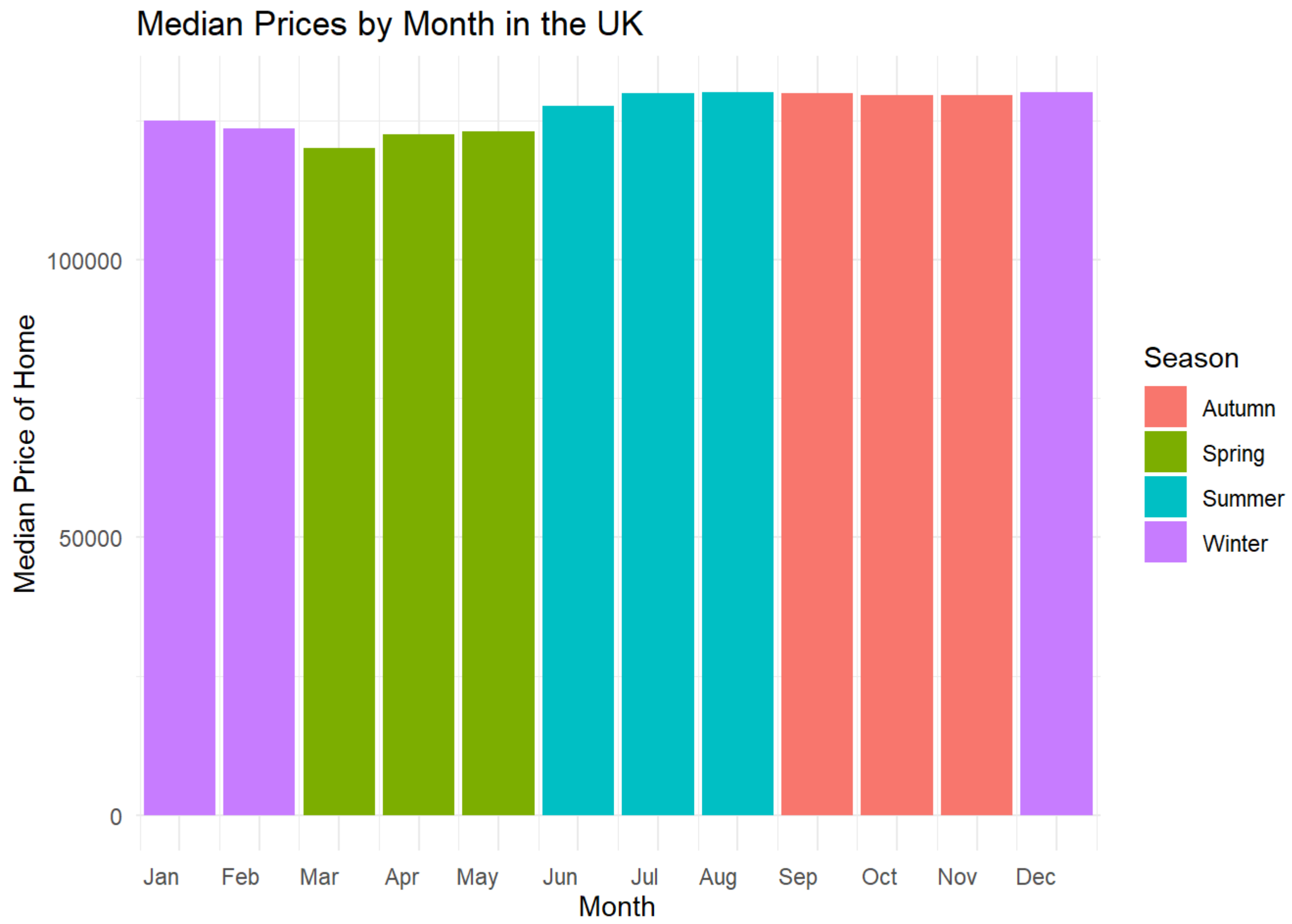
```
## Warning: Removed 1419 rows containing non-finite values (stat_smooth).
```



It

looks like the 'Other' category of home does not align with the categorized types of home. It also just emerged in 2006 - certainly worth looking into what this could be, as the price has skyrocketed. You can also see the impact of the housing crisis over time, where it flattens out and dips after 2007. To look at just the months (irrespective of the year) I can do the following:

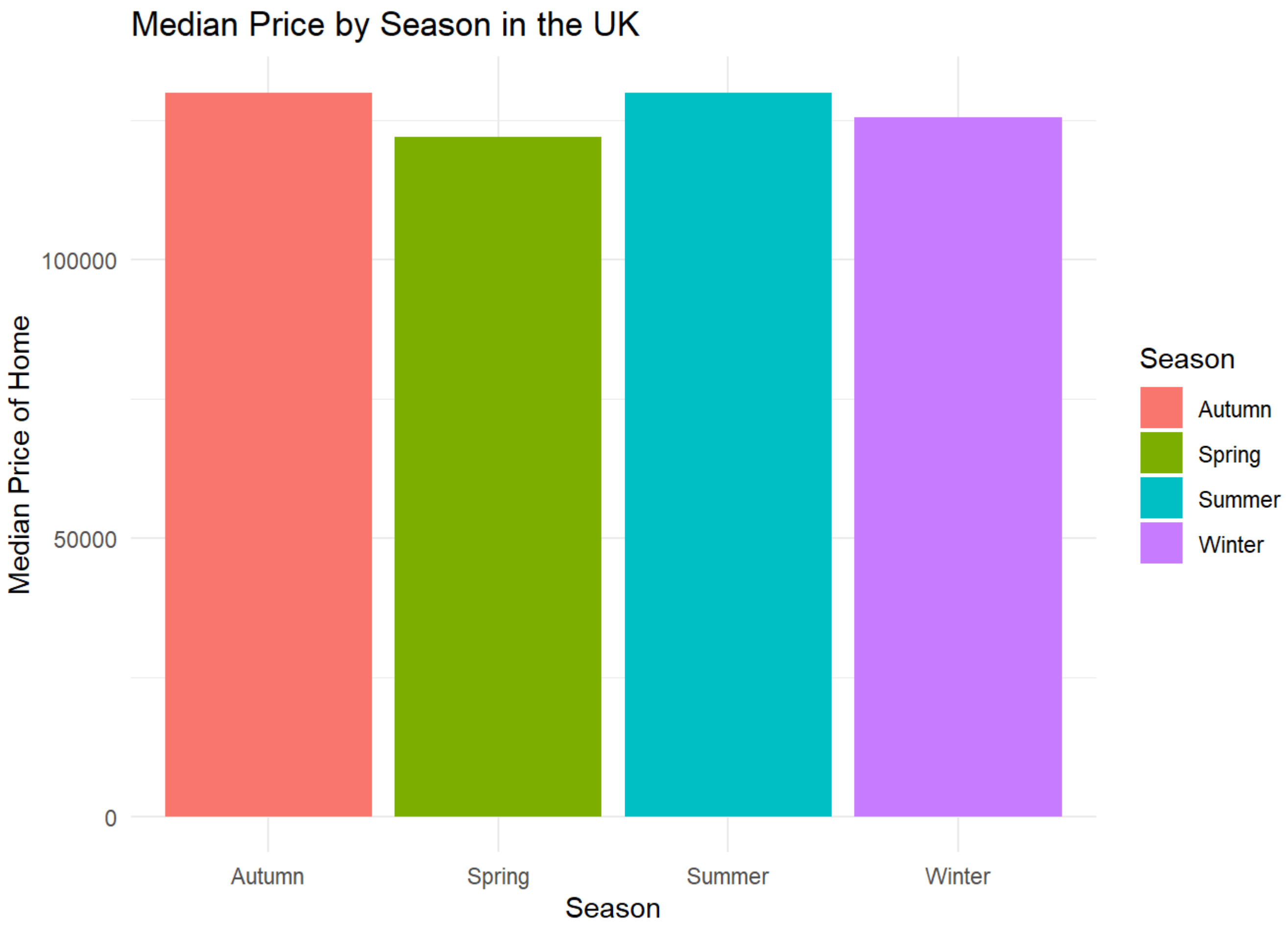
```
options(scipen=1000)
ggplot(df, aes(x=month, y=price, fill=Season))+
  geom_bar(stat="summary", fun ="median")+
  theme_minimal() +
  theme(axis.text.x = element_text(vjust = 0.5, hjust=1)) +
  ggtitle("Median Prices by Month in the UK")+
  xlab("Month") + ylab("Median Price of Home") +
  scale_x_continuous(breaks=min(df$month):max(df$month), expand=c(0,0.1), labels=monthlabels)
```



It

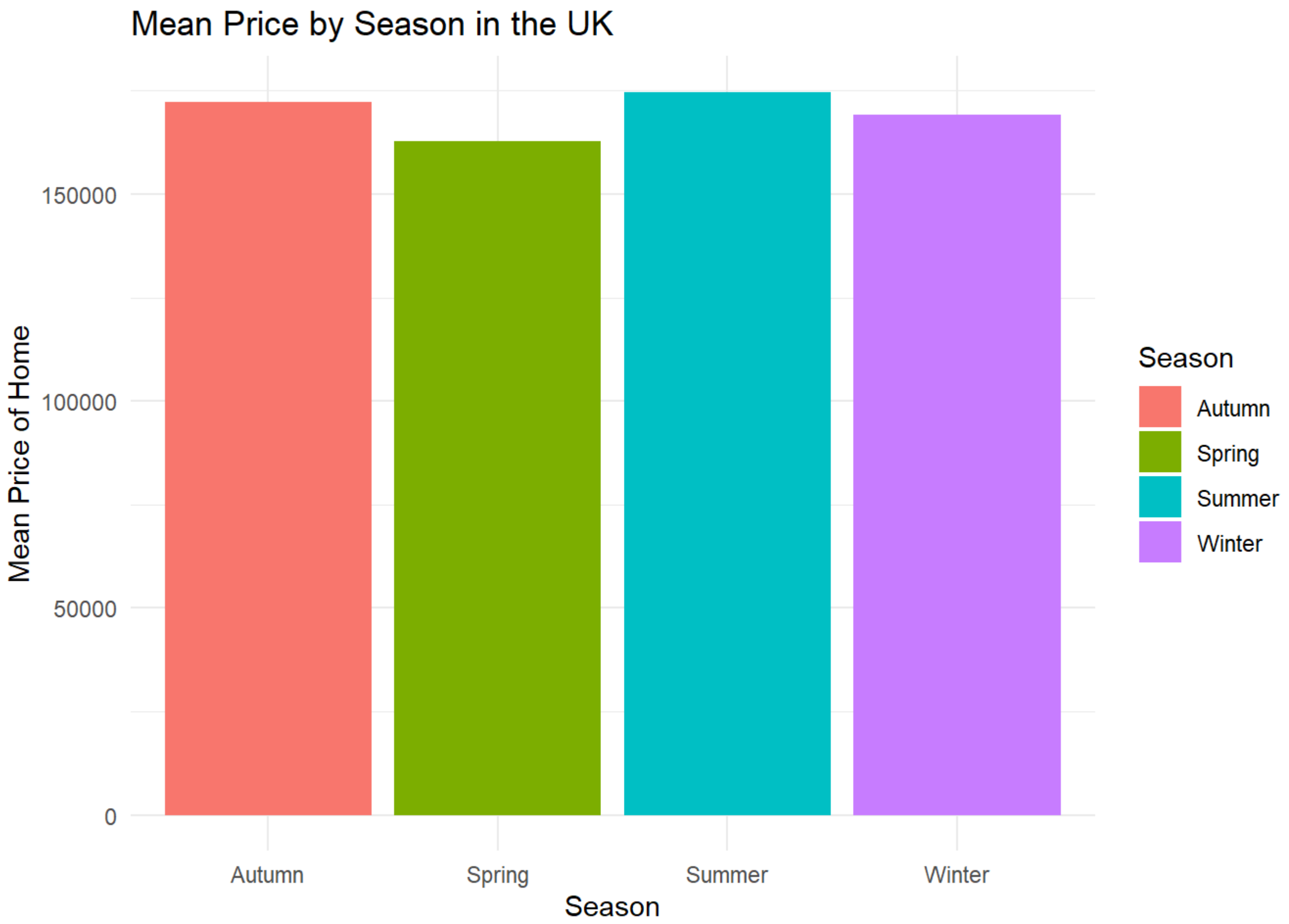
looks like over time median prices tend to peak in August and September.

```
options(scipen=1000)
ggplot(df, aes(x=Season, y=price, fill=Season))+
  geom_bar(stat="summary", fun ="median")+
  theme_minimal() +
  ggtitle("Median Price by Season in the UK")+
  xlab("Season") + ylab("Median Price of Home")
```



It does appear that there is a trend for higher home prices in the summer and autumn months. We can use a box plot to visualize more clearly and facet by property type.

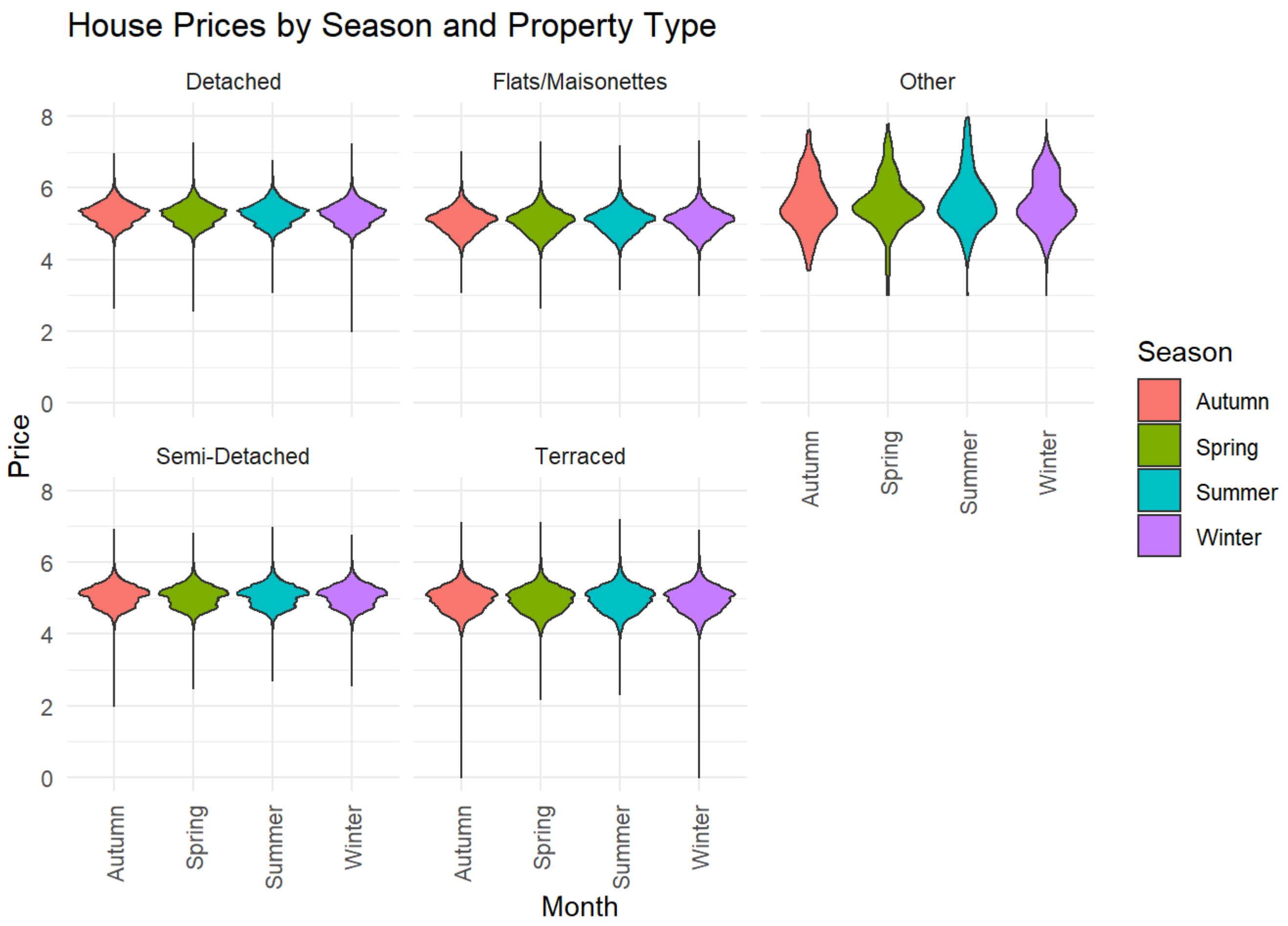
```
options(scipen=1000)
ggplot(df, aes(x=Season, y=price, fill=Season))+
  geom_bar(stat="summary", fun ="mean")+
  theme_minimal() +
  ggtitle("Mean Price by Season in the UK")+
  xlab("Season") + ylab("Mean Price of Home")
```



The

mean seems to show a similar trend, with Summer and Autumn being higher priced seasons.

```
ggplot(df, aes(y=logprice, x=Season, fill=Season)) +
    geom_violin() +
    theme_minimal() +
    labs(x = "Month", y = "Price")+
    ggtitle("House Prices by Season and Property Type") +
    #scale_x_continuous(breaks=min(df$month):max(df$month) +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))+
    #scale_y_continuous(expand=c(0,0.2))+
    facet_wrap(~property_type)
```

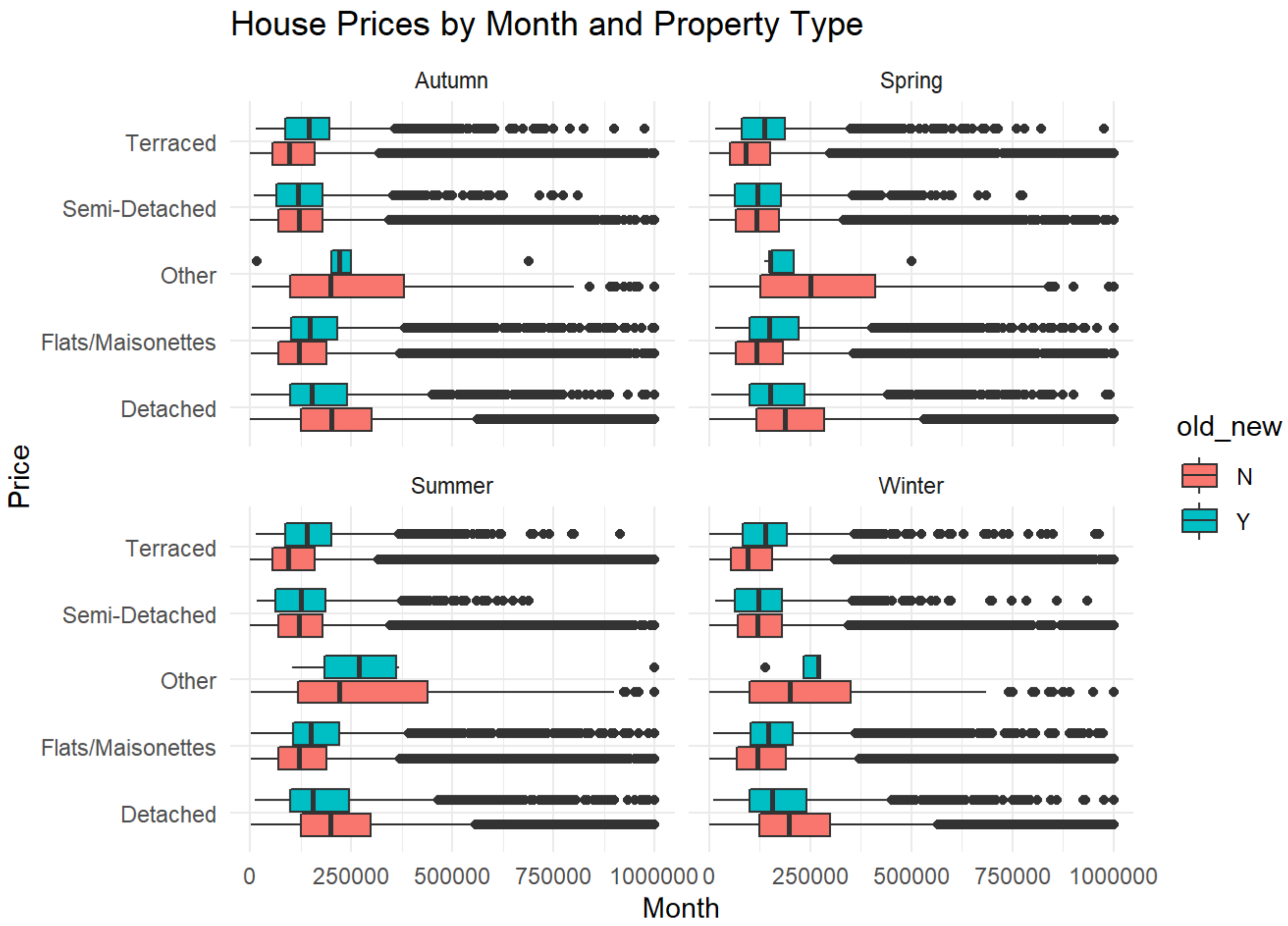


The

above does not seem to indicate a clear trend over time. Below I can plot it differently with the property_types on the y axis, looking at old_new and facet_wrapping by season.

```
ggplot(df, aes(fill= old_new, x=price, y=property_type)) +
    geom_boxplot() +
    theme_minimal() +
    labs(x = "Month", y = "Price")+
    ggtitle("House Prices by Month and Property Type") +
    #scale_x_continuous(breaks=min(df$month):max(df$month), expand=c(0,0.1)) +
    # scale_y_continuous(expand=c(0,0.2))+
    #theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))+
    facet_wrap(~Season) + xlim(c(0, 1000000))
```

```
## Warning: Removed 3159 rows containing non-finite values (stat_boxplot).
```



It does not appear that there is a significant trend depending on the season for each property type.