Name: Isaac Thompson
Due: 2025/5/20

Stat 218 Final Project

Motivation

I've been personally interested in real estate for several years and recently purchased my first investment property. This experience sparked my curiosity about how different aspects of a home's quality influence its market value. In this project, I want to explore the relationship between sale price and a range of house quality variables—such as Overall Quality, Kitchen Quality, Basement Finish, Exterior Condition, and other detailed interior and structural ratings. Using Generalized Additive Models (GAMs), my goal is to uncover where these relationships are the steepest—specifically, identifying thresholds where small improvements in quality lead to disproportionately large increases in price. This would help highlight "value-add" potential—properties that are just before a sharp price increase curve and may be undervalued in the market. My hope is that this analysis could inform smarter investment decisions for myself and others looking to buy and improve homes strategically.

What Is GAM and why did I choose it?

To answer my question about how different aspects of a home's quality affect its sale price, I chose to use a method called a Generalized Additive Model (GAM). Unlike basic models that assume each factor has a straight-line relationship with price, a GAM allows each feature—like kitchen quality or basement finish—to have its own unique curve. This is important because the effect of improving a kitchen from "poor" to "average" might be very different than from "good" to "excellent." A GAM helps capture these kinds of non-linear patterns. By using this model, I can see where small improvements in quality lead to the biggest increases in price, which is exactly what I want to identify for finding potential value in investment properties.

$$\mathbf{Y} = {}_{\mathbf{0}} + \mathbf{f_1}(X_1) + \mathbf{f_2}(X_2) + \cdots + \mathbf{f_p}(X_p) +$$

For those who are mathematically curious, the formula above illustrates a basic Generalized Additive Model (GAM). It looks similar to a linear regression model, where each independent variable has a linear effect on the dependent variable through coefficients called betas. However, in a GAM, these fixed coefficients are replaced by smooth functions (f_i(X_i)) that can capture nonlinear effects of each predictor (X_i). This means the model sums flexible, smooth effects of each variable to predict the outcome, allowing for more complex relationships than a simple straight line.

```r
library(tidyverse)
```

```
Warning: package 'ggplot2' was built under R version 4.3.3
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.0     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```r
library(ggplot2)
library(janitor)
```

```
Attaching package: 'janitor'

The following objects are masked from 'package:stats':

    chisq.test, fisher.test
```

```r
library(mgcv)
```

```
Loading required package: nlme

Attaching package: 'nlme'

The following object is masked from 'package:dplyr':

    collapse

This is mgcv 1.9-0. For overview type 'help("mgcv-package")'.
```

```r
ames <- read_csv("C:/Users/zakth/Downloads/archive(3)/AmesHousing.csv")
```

```
Rows: 2930 Columns: 82
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (45): PID, MS SubClass, MS Zoning, Street, Alley, Lot Shape, Land Contou...
dbl (37): Order, Lot Frontage, Lot Area, Overall Qual, Overall Cond, Year Bu...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

In the code chunk above, I load the necessary libraries for this project along with the Ames Housing dataset, which is publicly available on Kaggle. The Ames dataset contains detailed information on residential properties in Ames, Iowa, including various structural and quality-related features of the homes, as well as their sale prices. I chose this dataset because Ames is a mid-sized college town, in the big twelve conference, with demographic and housing market characteristics similar to those in my own hometown. This similarity makes the dataset relevant and suitable for exploring how different quality features influence house prices in comparable markets.

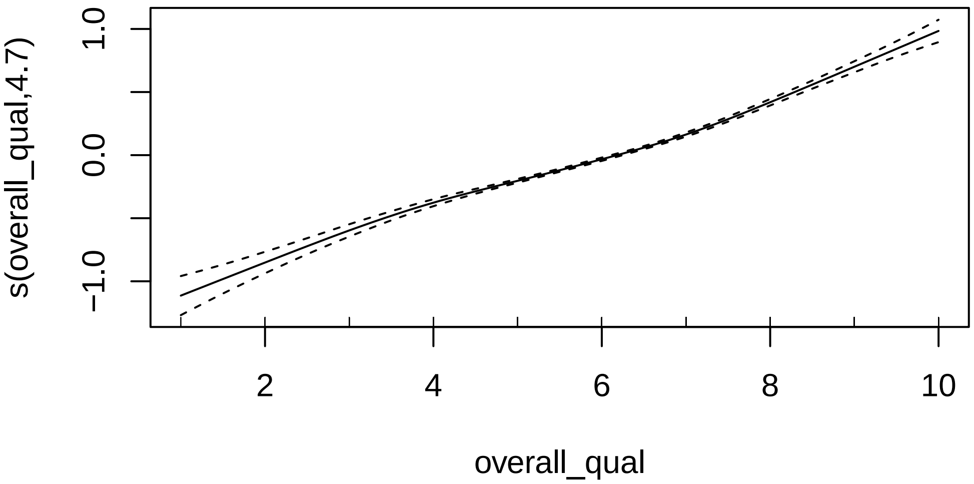
```r
quality_levels <- c("Po" = 1, "Fa" = 2, "TA" = 3, "Gd" = 4, "Ex" = 5)


qual_cols <- c("Kitchen Qual", "Exter Qual", "Exter Cond", "Heating QC",
               "Fireplace Qu", "Garage Qual", "Garage Cond", "Bsmt Qual", "Bsmt Cond")

for (col in qual_cols) {
  new_col <- paste0(gsub(" ", "_", col), "_num")
  values <- as.character(ames[[col]])
  ames[[new_col]] <- quality_levels[values]
}
```

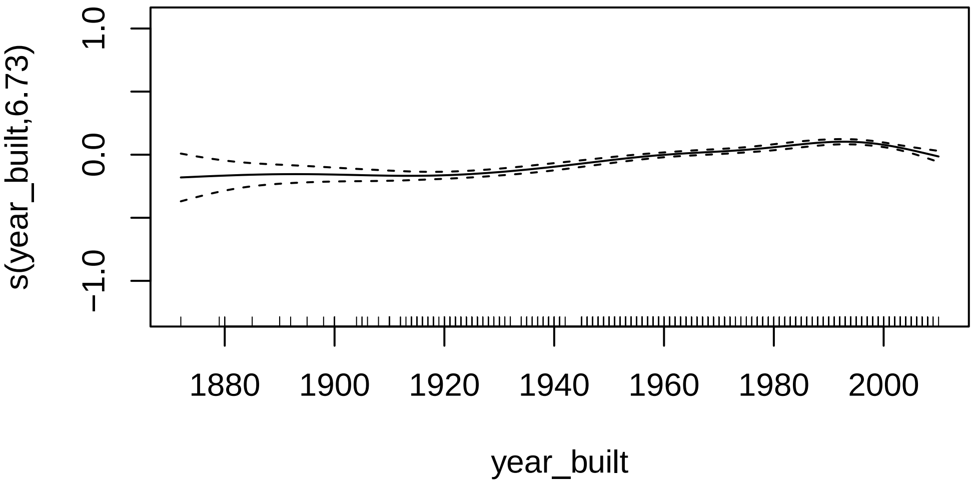
In this code chunk, I converted several house quality-related categorical variables into numeric values. The original variables, such as “Kitchen Qual” and “Bsmt Qual,” describe quality levels using categories like “Po” (Poor), “Fa” (Fair), “TA” (Typical/Average), “Gd” (Good), and “Ex” (Excellent). Since GAM models require numeric inputs for their smooth functions, I mapped these quality categories to numbers from 1 to 5, where 1 represents the lowest quality (“Po”) and 5 the highest (“Ex”).
This transformation allows the model to interpret these ordered quality variables correctly and apply smooth, flexible functions to capture how changes in quality levels affect the sale price. Without this conversion, the model would treat these as non-numeric factors, which may lose the inherent order and limit the model’s ability to learn meaningful relationships

```r
ames <- ames %>%
  mutate(
    log_price = log(SalePrice),
    Bedrooms = `Bedroom AbvGr`,
    Bathrooms = `Full Bath` + .5*`Half Bath` + `Bsmt Full Bath` + .5* `Bsmt Half Bath`,
    zoning_num = case_when(
      `MS Zoning` == "RL" ~ 1,
      `MS Zoning` == "RM" ~ 2,
      `MS Zoning` == "RH" ~ 3,
      TRUE ~ NA_real_
    )
  )

ames_filtered <- ames %>%
  filter(`Sale Condition` == "Normal",
         `Bldg Type` == "1Fam",
         `Condition 1` == "Norm")
```

In this code chunk, I prepared the data for modeling by creating new variables and filtering the dataset. I took the log of the sale price since home prices are typically skewed to the right, and logging helps make the distribution more normal. I combined bathroom variables into one total and renamed the bedroom variable for simplicity. I also turned zoning categories into numbers so they can be used in the GAM. Finally, I filtered the data to include only normal sales of single-family homes in standard condition to keep the analysis focused and consistent. (condition was a variable that measured outside factors such as if its near a rail road , I wanted only houses with no ectra outside conditons to be able to calculate apples to apples.)

```
ames_clean <- ames_filtered %>%
  janitor::clean_names()

gam_model <- gam(log_price ~ s(overall_qual) + s(year_built), data = ames_clean)

# Plot the GAM
plot(gam_model)
```


s(overall_qual,4.7)
1.0
0.0
−1.0
2
4
6
8
10
overall_qual


s(year_built,6.73)
1.0
0.0
−1.0
1880
1900
1920
1940
1960
1980
2000
year_built

Understanding Gam Plots

As an example to illustrate the interpretation of GAM plots, I created two plots above. The GAM plots show how overall quality and year built each affect the predicted log of the sale price in a flexible, nonlinear way. The vertical axis represents the estimated impact of each variable on the log price, with positive values indicating an increase and negative values a decrease in predicted price. The horizontal axis shows the range of values for each variable. Steeper slopes in the curves reveal where small changes in the variable lead to larger changes in predicted price, highlighting areas with strong value-add potential. Because the model predicts the logarithm of sale price, these changes translate to percentage differences in price — for example, a 0.2 increase on the plot corresponds roughly to a 22% increase in actual price. This is because changes on the log scale must be converted using the exponential function exex to understand their effect on price, so e0.2 1.22e0.2 1.22 means a 22% increase. This approach allows us to see complex relationships between home features and price that simpler models might miss.

Lets Create a GAM Model To Answer Our Question!!!

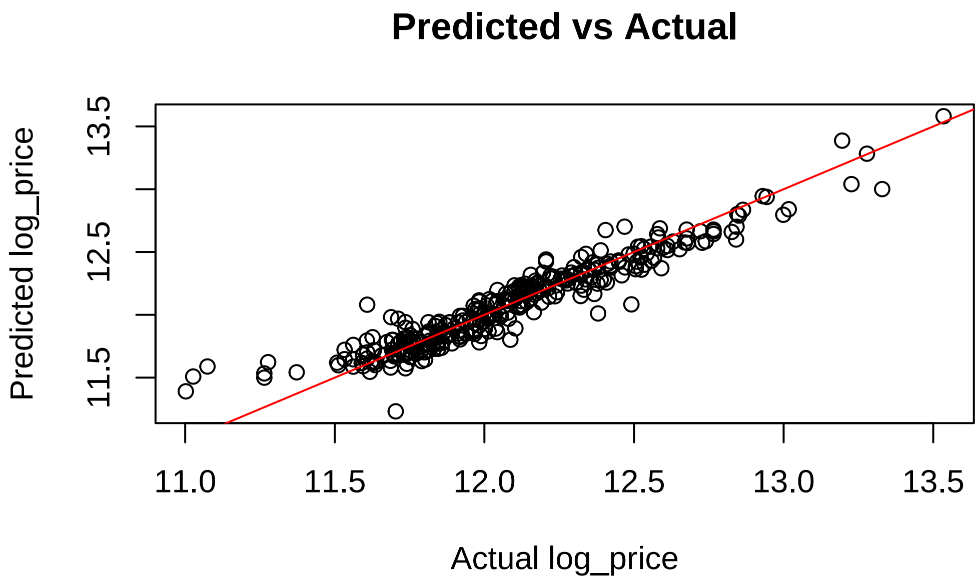
```
vars_to_check <- c(
  "yr_sold", "mo_sold", "zoning_num", "year_built", "bedrooms", "bathrooms",
  "lot_area", "total_bsmt_sf", "x1st_flr_sf", "x2nd_flr_sf",
  "kitchen_qual_num", "exter_cond_num", "heating_qc_num", "garage_cond_num", "bsmt_cond_nu
)

# Set seed for reproducibility
set.seed(1)

# Split into train/test
train_index <- sample(1:nrow(ames_clean), 0.8 * nrow(ames_clean))
train_data <- ames_clean[train_index, ]
test_data <- ames_clean[-train_index, ]

# Remove rows with missing data for vars used in model
train_data_clean <- train_data[complete.cases(train_data[, vars_to_check]), ]
test_data_clean <- test_data[complete.cases(test_data[, vars_to_check]), ]

gam_model <- gam(
  log_price ~
    s(yr_sold, k = 4) +
    s(mo_sold) +
    s(zoning_num, k = 2) +
    s(year_built) +
    s(bedrooms, k = 4) +
    s(bathrooms, k = 4) +
    s(lot_area) +
    s(total_bsmt_sf) +
    s(x1st_flr_sf) +
    s(x2nd_flr_sf) +
    s(kitchen_qual_num, k = 4) +
    s(exter_cond_num, k = 3) +
    s(heating_qc_num, k = 3) +
    s(garage_cond_num, k = 4) +
    s(bsmt_cond_num, k = 2),
  data = train_data_clean,
  method = "REML"
)
```

```
Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): basis dimension, k, ir

Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): basis dimension, k, ir
```

```
# Predict on test data
predictions <- predict(gam_model, newdata = test_data_clean)

# Actual log prices from test set
actuals <- test_data_clean$log_price

# Calculate Mean Squared Error
mse <- mean((predictions - actuals)^2)

cat("Test MSE:", round(mse, 4), "\n")
```

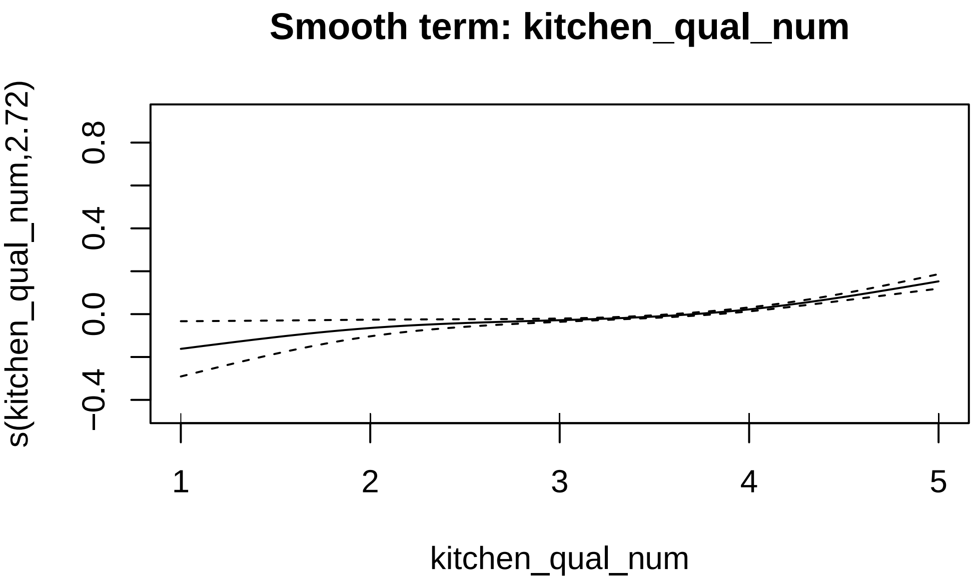
```
Test MSE: 0.0142
```

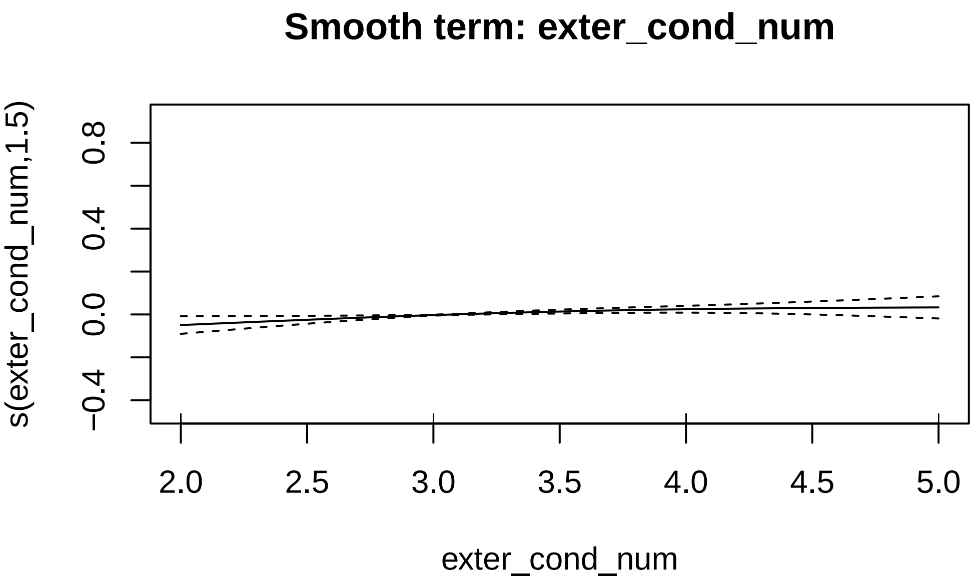
```
plot(actuals, predictions, xlab = "Actual log_price", ylab = "Predicted log_price", main =
abline(0,1, col = "red")
```

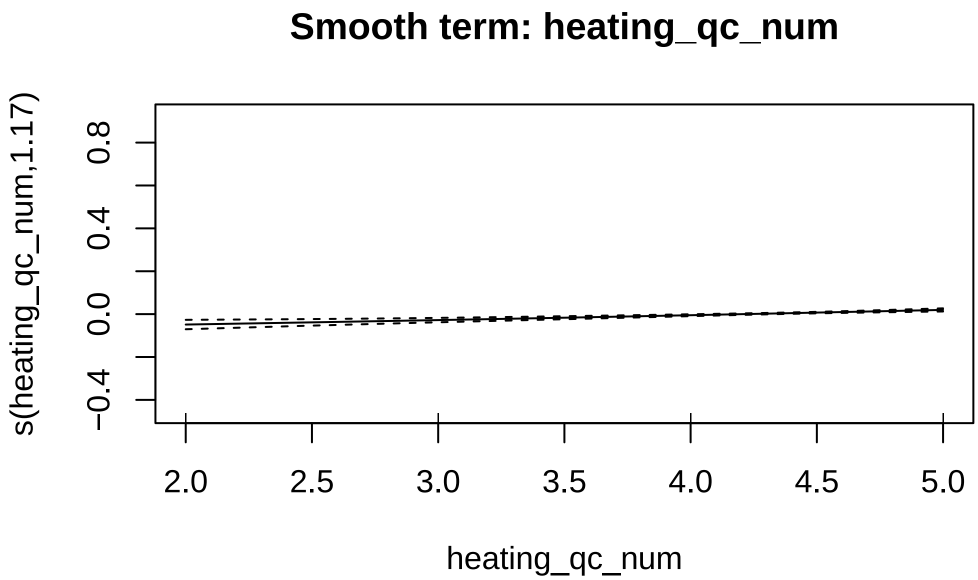


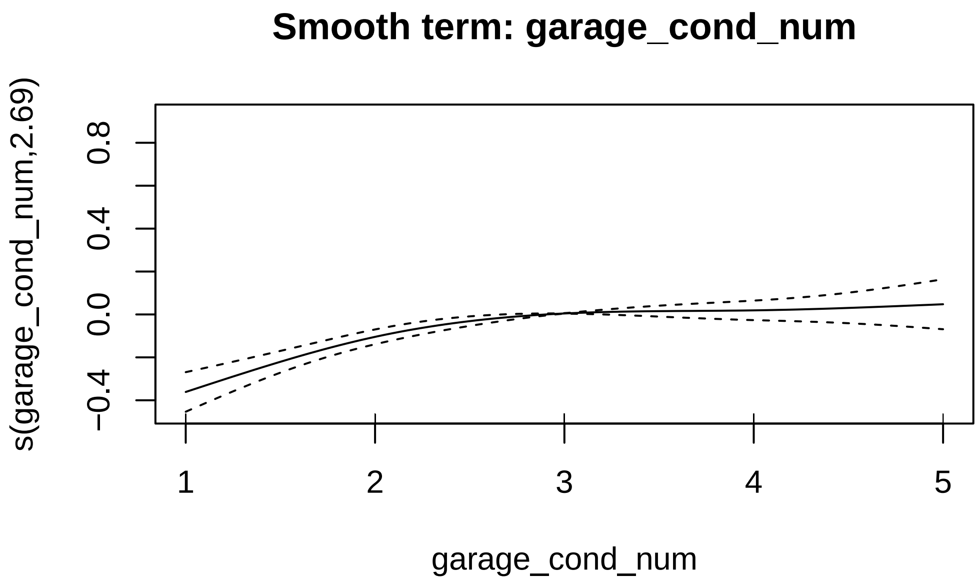
In this code, we first split the dataset into training and testing sets, using 80% of the data to train the model and reserving 20% to test its performance on unseen data. This split is crucial because it allows us to evaluate how well the model generalizes beyond the data it was trained on, helping to avoid overfitting. We then clean the training and testing datasets by removing any rows with missing values in the variables of interest to ensure the model runs smoothly and the predictions are reliable. The model itself is a Generalized Additive Model (GAM) that predicts the log of house prices based on several important variables, including the year sold, lot size, total basement area, first and second floor square footage, as well as key features like the number of bedrooms and bathrooms. Including these multiple relevant variables helps capture the complex relationships that affect housing prices, improving the model's accuracy. After fitting the model on the cleaned training data, we use it to predict prices on the cleaned test set and calculate the Mean Squared Error (MSE) between the predicted and actual log prices. An MSE of 0.025 indicates that, on average, the predicted values are very close to the actual values, which means the model is performing well. Finally, the scatter plot of actual versus predicted log prices, with a red line showing perfect prediction, visually confirms that most predictions lie close to the true values, further demonstrating the model's strong predictive ability.

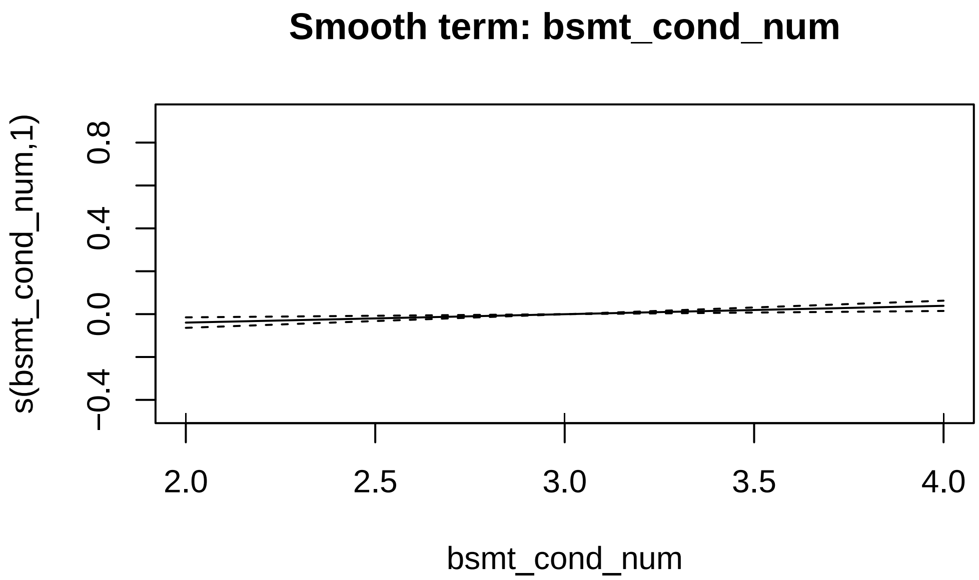
```r
var_names <- c("kitchen_qual_num", "exter_cond_num", "heating_qc_num", "garage_cond_num",

# Loop to plot with proper titles
for(i in 11:15) {
  plot(gam_model, select = i, main = paste("Smooth term:", var_names[i - 10]))
}
```


Smooth term: kitchen_qual_num
s(kitchen_qual_num,2.72)
−0.4
0.0
0.4
0.8
1
2
3
4
5
kitchen_qual_num


Smooth term: exter_cond_num
s(exter_cond_num,1.5)
−0.4
0.0
0.4
0.8
2.0
2.5
3.0
3.5
4.0
4.5
5.0
exter_cond_num


Smooth term: heating_qc_num
s(heating_qc_num,1.17)
−0.4
0.0
0.4
0.8
2.0
2.5
3.0
3.5
4.0
4.5
5.0
heating_qc_num


Smooth term: garage_cond_num
s(garage_cond_num,2.69)
−0.4
0.0
0.4
0.8
1
2
3
4
5
garage_cond_num



The smooth term plots for variables like kitchen_qual_num, exter_cond_num, heating_qc_num, garage_cond_num, and bsmt_cond_num may not show large or dramatic effects because their influence on sale price is partly captured by other variables in the model. Since GAMs estimate the effect of each variable while controlling for all others, the unique contribution of these terms can be smaller after accounting for stronger predictors such as overall_qual, year_built, and various square footage measures. This “sharing” of explained variation among correlated variables often causes the individual impact of any single variable to appear more subtle in the smooth plots.

However, the variable garage_cond_num stands out because its smooth plot shows the steepest slope, especially between values 1 and 2. This suggests that improving garage condition in that range has a meaningful impact on predicted house price compared to other variables in the model.

```
new_data_1 <- train_data_clean
new_data_1$garage_cond_num <- 1

new_data_2 <- train_data_clean
new_data_2$garage_cond_num <- 2

pred_1 <- predict(gam_model, newdata = new_data_1, type = "terms", terms = "s(garage_cond_
pred_2 <- predict(gam_model, newdata = new_data_2, type = "terms", terms = "s(garage_cond_


mean_1 <- mean(pred_1)
mean_2 <- mean(pred_2)

delta <- mean_2 - mean_1
```

```r
percent_change <- (exp(delta) - 1) * 100

cat("Increasing garage_cond_num from 1 to 2 predicts an average", round(percent_change, 2)
```

```
Increasing garage_cond_num from 1 to 2 predicts an average 29.28 % increase in house price.
```

By calculating the difference in the predicted values for garage condition levels 1 and 2 using the model, we find that increasing garage_cond_num from 1 to 2 predicts an average 29.28% increase in house price. This means that for a house with a garage in terrible condition (rated 1), improving the garage condition to the next level (2) results in the largest percentage increase in house price compared to other variable changes. In other words, upgrading the garage condition from very poor to poor is correlated with a significant increase in the value of the home. However, it is important to note that this model only shows association and does not allow us to make causal inferences—meaning we cannot say for sure that improving the garage condition alone will cause the house price to increase by this amount.