Southern Methodist University

Predicting Sales Prices in the Housing Market:

A Comparative Analysis of Regression Models

O'Neal Gray & Matthew David

MSDS 6371 Stat Foundations

Dr, Bevin Sadler

10 April. 2023

Introduction

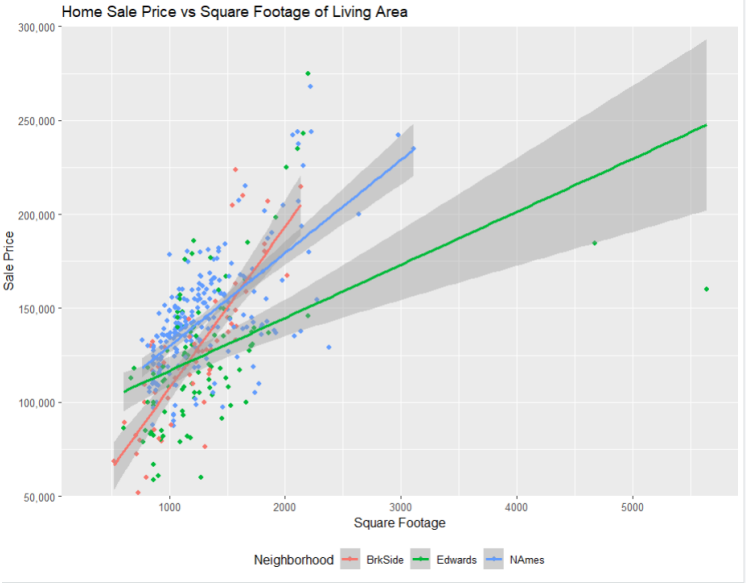
Century 21 Ames, a real estate company based out of Ames Iowa, would like to better understand their local market. Century 21 Ames wants to know how the sale price and square footage of homes are related to the neighborhoods they primarily sell in. This encompasses neighborhoods such as North Ames, Edwards, and Brookside, and they would also like to know if the relationship between the sale price and square footage is affected by which neighborhood a house is sold in. Century 21 Ames also requested the most predictive model for the sale prices of homes in all of Ames Iowa. For this study, we first built and fit a linear regression model to estimate the relationship between the sale price and square footage taking into account which neighborhood houses were sold. We made sure to validate the assumption of the model by visually checking the distribution, q-qplots, and plots of the residuals to ensure homoscedasticity and equal variance of the residuals. Thus we proceeded under the assumption of independence of observation within and between each variable. We made sure to address outliers and missing values in the dataset. To ensure we got the most predictive model for sale price in Ames Iowa, we produced four separate models including a forward selection, backward elimination, stepwise selection, and a custom model. We include a table that highlights key statistics for each model, and finally a brief conclusion suggesting which model would be the best fit for Century 21 Ames.

Data Description

The data for this study is pulled from the Ames Housing dataset compiled by Dean De Cock. The data includes a training and test dataset. The training dataset includes 1,460 observations of 81 variables, most notably the dataset includes **GrLivArea**(square footage of living area), and **Sale Price** which are both key variables in our case study. The data also includes a training set of 1,459 different observations of the same variables excluding the Sale Price variable. With the purpose of testing predictive Sale Price models generated utilizing the training dataset. More information on the data used in this study can be found on the cited webpage below "House Prices-Advanced Regression Techniques".

Neighborhood's Effect on the Relationship between Sales Price & Square Footage

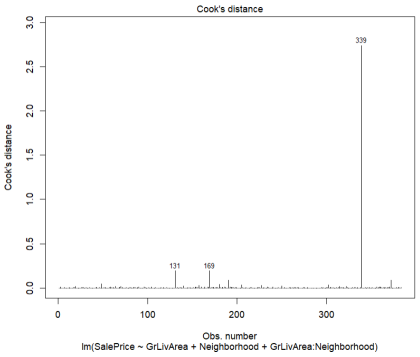
Century 21 Ames wants to enhance their comprehension of the local market by investigating if the relationship between Sale Prices and square footage of the living area is influenced by the location of homes, particularly those situated in the neighborhoods where they conduct business. Our first course of action was to check the training dataset for missing values. We found a total of 6,965 missing observations across 19 variables. However, the variables relevant to the model Sale Price, Neighborhood, and GrLivArea all had zero missing observations. Therefore we will leave the missing values in the dataset to preserve the total degrees of freedom of the model and will address them more thoroughly before analyzing question 2. Century 21 Ames operates in three of the twenty-five local neighborhoods in Ames so after narrowing the range of the model we produced a scatter-plot to depict the linear relationship between Sale Price and Square footage of living area taking into account which neighborhood houses were sold in. The resulting scatter-plot gives us some indication that there may be influential outliers specifically noting the outliers in the Edwards, and North Ames neighborhoods. There also appears to be some visual evidence suggesting different slopes of the regression lines, therefore we will account for this in our model by allowing for interactions.

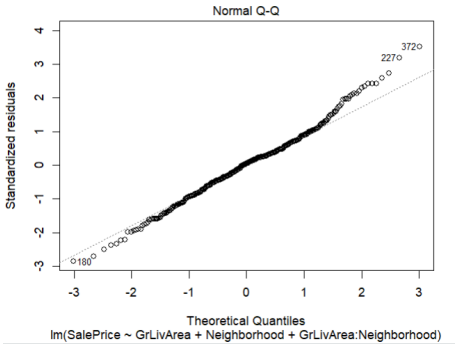


Equation: *Sale Price = Bo + B1(GrLivArea)+B2(Edwards(Neighborhood)) + B3(NAmes(Neighborhood))+ B4(GrLivArea: Edwards) + B5(GrLiveArea: NAmes)*

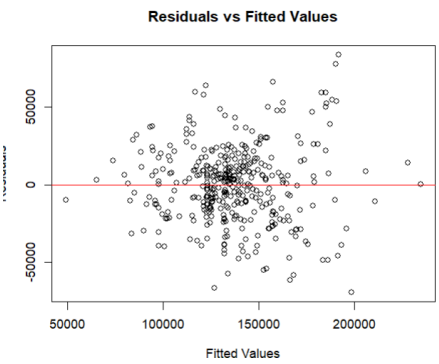
** The Brookside Neighborhood was set as the reference for the model*

Once we fit the model we generated plots to visually check the assumptions of linear regression. Upon examining the first q-q plot, four observations (131,169,190,339) stood out as potentially influential. On further inspection observations 131 and 339 are both outliers in the GrLivArea variable of homes in Edwards. While the other outliers(169,190) are the Sale Prices of homes residing in North Ames. The four observation values are much greater than the mean of their respective variables (Mean Sale Price: 138,062 Mean GrLivArea:1300 Sqft).





After checking Cook's distance depicted in the first model on the right only observation 339 appeared to be influential, however after removing the observation, the other outliers became more influential. So we removed all four observations and refit our model concluding that they were data entry errors. The q-q plot and plot of residuals vs fitted values on the right were generated after removing the outliers. Looking at the plots there is not enough visual evidence to suggest against homoscedasticity and equal variance of the residuals. Therefore we proceeded with the linear regression model under the assumption of normality, equal variance, and independence of observations in and between variables.

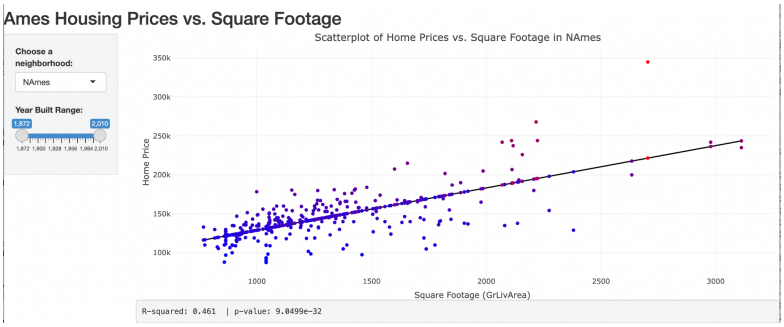


The refit model suggests that for every 100 sqft increase in the general living area, the estimated Sale Price of homes will increase between 7,052 and 10,400 dollars at a 95% confidence level. Our best estimate is an increase of 8,716 dollars per 100 sqft increase(p-value: <0.0001). This is with respect to the reference neighborhood Brook Side. It is estimated at a 95% confidence level that in North Ames the Sale Price per 100 sqft will be between 5,609 and 1,911 dollars lower than the Sale Price in Brookside(p-value: <0.0001). There is not enough statistical evidence however to indicate the relationship between Sale Price and GrLivArea is different in Edwards compared to Brookside(Intercept p-value: 0.23, Adjustment p-value:0.12).

It is important to note that sqft cannot be equal to zero, as all properties have a value above zero for square footage. Therefore, interpreting the Sale Price when GrLivArea is equal to zero sqft is not meaningful in this context. The adjusted r^2 of the model is 0.52 meaning 52% of the variability in sales price can be explained by its relationship with general living area sqft, and neighborhood. Finally addressing Century 21 Ames question, there is statistically significant evidence to suggest that the neighborhood has an effect on the relationship between Sale Price and General Living Area sqft (p-value:< 0.001). As this was an observational study no causal relationship can be inferred however, an association between the variables(Sale Price, GrLivArea, Neighborhood) can be made on the population of neighborhoods in Ames Iowa in which Century 21 Ames sells homes.

R Shiny: Ames Housing Prices Vs. Square Footage

We created an R-Shiney application to have an interactive visualization for exploring the relationship between house prices and square footage in our housing dataset. Our application allows the user to select any neighborhood in the data set and the range of years for houses built in that neighborhood. We leveraged the Plotly library to generate a scatterplot that shows the relationship between house prices (y-axis) and square footage (GRLivArea, x-axis), where each point represents a house from our dataset. The point's color represents the sales price, with a gradient of blue(low) and red(high). We also implemented a linear regression line that shows the relationship between square footage and house prices within the selected neighborhood.



The application's user interface has a sidebar containing a drop-down menu to select the neighborhood and a slider to choose a range of years built. The application output's a scatterplot and a text output of the linear regression model's R-squared value and p-value, which indicate the model's fit and the statistical significance of the relationship between square footage and house prices. We built this application so the user can interact with it by changing the neighborhood or year range, which will update the scatterplot and the model statistics.

Predicting Ames Home Prices: Comparing Regression Techniques

The objective of question two is to construct a highly predictive model for the sales price of homes in Ames, Iowa that encompasses all neighborhoods. The study was limited to techniques that we have covered in our course which are specifically forward selection, backward elimination, stepwise selection, and our custom model. To evaluate the performance of each model, we generated the adjusted R2, CV Press, and Kaggle Score metrics. Ultimately, our goal was to identify the model that can best predict future sale prices of homes in Ames, Iowa, and provide a clean explanation of our selection criteria.

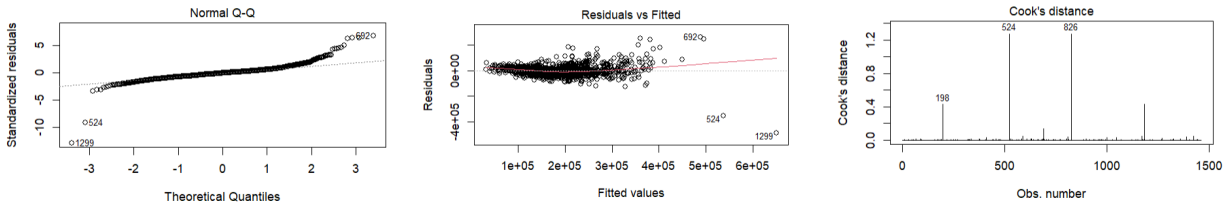
To begin our statistical analysis, we constructed four models, including forward selection, backward elimination, stepwise selection, and a custom model. We began by importing our data from the CSV file "train.csv". To address missing values, we adopted a strategy informed by the data_description.txt file, reclassifying columns with missing values such as "FireplaceQU" with "no_fireplace" and similarly for other relevant columns. We utilized the is.na() function to identify columns with missing values and then applied indexing to assign specific values to those missing values. We then imputed missing values for "Lot Frontage" and "Electrical". "Lot Frontage" originally had 259 missing values, and we inputted those missing values using the median of the available values. As for "Electrical", only one value was missing, and we imputed it with the most frequent category. Next, we converted all character variables to factors using the as.factor() function, and dropped any unused levels using the droplevels() function. In order to handle categorical variables, we generated dummy variables through the dummyVars() and predict() functions, before subsequently combining them with the original data using the Cbind() function. Finally, our final output consisted of cleaned and formatted data, with SalePrice as the first column, primed for our statistical analysis.

Our team generated four different regression models using the OLSRR (Ordinary Least Squares Regression Residuals and Diagnostics) package in R. These models were developed to predict the sale price of houses in the US housing market. The first model we developed was the Forward model, which included 35 predictors. This model had an adjusted R2 of .9173, which means that the model explains 91.73% of the variability in the response variable (sale price) after adjusting for the number of predictors in the model. Additionally, the CV Press for this model was 43168.59, which indicates the mean squared prediction error when using leave-one-out cross-validation. Finally, this model had a Kaggle Score of .18626, a metric used to evaluate the accuracy of a model in the Kaggle competition.

On the other hand, the Backward model used 37 predictors and had the same adjusted R2 of .9173 as the Forward model. However, this model had a higher CV Press of 44072.33 and a lower Kaggle Score of .17681, indicating a higher prediction error and lower accuracy than the Forward model. The Stepwise model was our most complex model, using 53 predictors, but it also had the highest adjusted R2 of .9195. However, this model also had the highest CV Press of 44752.62 and a higher Kaggle score of .22085, indicating a higher prediction error and lower accuracy than the other models. Our Custom model, developed with guidance from a US Housing Specialist in Texas, was the model we ultimately chose to use. It only used seven predictors and had an adjusted R2 of .8013, a CV Press of 36537.93, and a Kaggle Score of .18461. Despite having a lower adjusted R2 than the other models, the Custom model had the lowest CV Press and Kaggle score, indicating the lowest prediction error and highest accuracy among the models we developed. Below is a table of the key statistics for each model.

Predictive Models	Adjusted R2	Cv Press	Kaggle Score
Forward	0.9173	43168.59	0.18626
Backward	0.9173	44072.33	0.17681
Stepwise	0.9195	44752.62	0.22087
Custom	0.8013	36537.93	0.18461

To ensure the accuracy and reliability of the Custom model we generated the QQ and Residual plots below to visually check the assumptions of linear regression. However, upon examining the q-q plot and plot of residuals versus fitted points, we observed three outliers (524, 692, 1299) that could be potentially influential. To assess their impact, we checked Cook's distance, as depicted in the last model on the right, and found that none of the outliers shown in the q-q plot and plot of the residuals were overly influential. The most influential points (524, 826) had an estimated Cook's distance of 1.3, which is below our threshold of 2.0. Therefore, we concluded that the outliers had low influence and did not impact the assumptions. Based on our examination of the plots, we found no evidence to suggest against homoscedasticity and equal variance of the residuals. However, because we imputed missing values with the average of their respective variable, we acknowledge that we may have violated the assumption of independence in our analysis. We noted this limitation and proceeded with our analysis exercising caution.



In response to Century 21 objective we built the most predictive model for the sales price of homes in Ames, Iowa. We generated four models in response using forward selection, backward elimination, stepwise selection, and our custom model. Ultimately we selected the Custom Model due to its lower CV Press and Kaggle Score, indicating higher accuracy and lower prediction error compared to the other models. Looking at the adjusted R2 of the Custom Model(.8013), 80% of the variability in Sale Price is explained by the custom model suggested by the US housing specialist.The Custom Model’s assumptions of linear regression were checked, and the outliers were deemed to have a negligible influence. As this was an observational study, no casual relationship can me inferred, only a association can be made between the variables in the neighborhoods that reside in Ames Iowa. Finally it is important to note that due to the violations of the independence assumption there may be limitations of our resulting model generated for Century 21 Ames.

Appendix

Analysis Question 1 Code

```
# Load Library's for analysis
library(tidyverse)
library(lmtest)
library(ggplot2)
library(stats)

# load in Train & Test set

train = read.csv(file.choose())
test = read.csv(file.choose())

# check data columns for missing values

cols_missing = colSums(is.na(train))
print(cols_missing)

# total missing values

total_missing = sum(is.na(train))

#filter data for neighborhoods

train2 = train %>% filter(Neighborhood %in% c("NAmes", "Edwards", "BrkSide"))

test2 = test %>% filter(Neighborhood %in% c("NAmes", "Edwards", "BrkSide"))


# graphic of regression model

plot = train2 %>% ggplot(aes(x = GrLivArea, y = SalePrice, color = Neighborhood)) +
 geom_point()+ geom_smooth(method = "lm") + ggtitle("Home Sale Price vs Square Footage of Living Area") +
  ylab("Sale Price") + xlab("Square Footage") + theme(legend.position = "bottom")

plot + scale_y_continuous(limits = c(50000, 300000), expand = c(0,0),
    labels = function(x) format(x, big.mark = ",", scientific = FALSE, trim = TRUE, drop0trailing = TRUE, decimal.m

# fit the model

#lm = SalePrice ~ GrLivArea + Neighborhood + GrLivArea:Neighborhood
model = lm(SalePrice ~ GrLivArea + Neighborhood + GrLivArea:Neighborhood, data= train2)
summary(model)
confint(model)

#check assumption

residuals = model$residuals
fitted_values = model$fitted.values
plot(fitted_values, residuals, main = "Residuals vs Fitted Values", xlab = "Fitted Values", ylab = "Res
abline(h=0, col = "red")

#independence
acf(residuals, main = "Autocorrelation of Residuals")
#the autocorrelations are very low, thus the independence assumptions is met


#indentify outliers
plot(model, which = 1:4)

# remove outleirs from data

train_clean =  train2[-339, ]

# refit the model

model2 = lm(SalePrice ~ GrLivArea + Neighborhood + GrLivArea:Neighborhood, data= train_clean)
summary(model2)
confint(model2)

#identify outliers of refit model
plot(model2, which = 1:4)

# remove other outliers including 169, 190, 131

train_clean2 = train2[-c(169,339,190,131), ]

# refit model without outliers

model3 = lm(SalePrice ~ GrLivArea + Neighborhood + GrLivArea:Neighborhood, data= train_clean2)
summary(model3)
confint(model3)

# check assumptions/ Cook's distance of final model
plot(model3, which = 1:4)
# plot residuals

residuals2 = model3$residuals
fitted_values2 = model3$fitted.values
plot(fitted_values2, residuals2, main = "Residuals vs Fitted Values", xlab = "Fitted Values", ylab = "R
abline(h=0, col = "red")
```

Rshiny App Code

```
library(plotly)
library(shiny)

train_data = read.csv("/Users/databbese/Desktop/SMU/Statistical Foundations/train.csv")
train_data = as.data.frame(train_data)

ui = fillPage(
  titlePanel("Ames Housing Prices vs. Square Footage"),
  fluidRow(
    column(2,
           wellPanel(
             selectInput("neighborhood",
                         "Choose a neighborhood:",
                         choices = sort(unique(train_data$Neighborhood)),
                         selected = "NAmes"),
             sliderInput("year_range", "Year Built Range:",
                         min(train_data$YearBuilt), max(train_data$YearBuilt),
                         range(train_data$YearBuilt), step = 1)
           )
    ),
    column(10,
           plotlyOutput("scatterplot", height = "100%"),
           verbatimTextOutput("model_stats")
    )
  ),
  tags$head(tags$style(HTML("
    .container-fluid {
      height: 100%;
      display: flex;
      flex-direction: column;
    }

    .fluidRow {
      flex: 1;
      display: flex;
    }

    .col-sm-10 {
      flex: 1;
      height: 100%;
    }
  ")))
)

server = function(input, output) {
  output$scatterplot = renderPlotly({
    selected_neighborhood = input$neighborhood
    year_range = input$year_range

    neighborhood_data = train_data[train_data$Neighborhood == selected_neighborhood & train_data$YearBuilt >= year_range[1] & train_data$YearBuilt <= year_range[2],]

    color_gradient = colorRampPalette(c("blue", "red"))(n = 100)
    point_colors = color_gradient[cut(neighborhood_data$SalePrice,
                                      breaks = 100
```

Analysis Question 2 Code

```
# analysis question 2
  # load libraries
library(tidyverse)
library(caret)
library(dplyr)
library(olsrr)

# Read in the data
train_data = read.csv(file.choose())
test_data = read.csv(file.choose())

# check data for missing values
na_counts = colSums(is.na(train_data))
na_counts

# data Cleaning
train_data$SalePrice = as.numeric(train_data$SalePrice)

#clean up Alley Coloumn
train_data$Alley[is.na(train_data$Alley)] = "no_pool"
test_data$Alley[is.na(test_data$Alley)] = "no_pool"

#clean up fireplace coloumn
train_data$FireplaceQu[is.na(train_data$FireplaceQu)] = "no_fireplace"
test_data$FireplaceQu[is.na(test_data$FireplaceQu)] = "no_fireplace"

#clean up PoolQC
train_data$PoolQC = as.character(train_data$PoolQC)
train_data$PoolQC[is.na(train_data$PoolQC)] = "no_pool"
test_data$PoolQC = as.character(test_data$PoolQC)
test_data$PoolQC[is.na(test_data$PoolQC)] = "no_pool"
#Clean up fence
train_data$Fence[is.na(train_data$Fence)] = "no_fence"
test_data$Fence[is.na(test_data$Fence)] = "no_fence"
#Clean up Misc Features
train_data$MiscFeature[is.na(train_data$MiscFeature)] = "none"
test_data$MiscFeature[is.na(test_data$MiscFeature)] = "none"
#Clean up BsmtQual
train_data$BsmtQual[is.na(train_data$BsmtQual)] = "no_basement"
test_data$BsmtQual[is.na(test_data$BsmtQual)] = "no_basement"
#Clean up BsmtCond
train_data$BsmtCond[is.na(train_data$BsmtCond)] = "no_basement"
test_data$BsmtCond[is.na(test_data$BsmtCond)] = "no_basement"
#Clean up BsmtExposure
train_data$BsmtExposure[is.na(train_data$BsmtExposure)] = "no_basement"
test_data$BsmtExposure[is.na(test_data$BsmtExposure)] = "no_basement"

#Clean up BsmtFinType1
train_data$BsmtFinType1[is.na(train_data$BsmtFinType1)] = "no_basement"
test_data$BsmtFinType1[is.na(test_data$BsmtFinType1)] = "no_basement"
#Clean up BsmtFinType1
train_data$BsmtFinType2[is.na(train_data$BsmtFinType2)] = "no_basement"
test_data$BsmtFinType2[is.na(test_data$BsmtFinType2)] = "no_basement"
#Clean up GarageType
train_data$GarageType[is.na(train_data$GarageType)] = "no_garage"
test_data$GarageType[is.na(test_data$GarageType)] = "no_garage"
#Clean up GarageFinish
train_data$GarageFinish[is.na(train_data$GarageFinish)] = "no_garage"
test_data$GarageFinish[is.na(test_data$GarageFinish)] = "no_garage"

#Clean up GarageQual
train_data$GarageQual[is.na(train_data$GarageQual)] = "no_garage"
test_data$GarageQual[is.na(test_data$GarageQual)] = "no_garage"
#Clean up GarageYrBlt
train_data$GarageYrBlt[is.na(train_data$GarageYrBlt)] = "no_garage"
test_data$GarageYrBlt[is.na(test_data$GarageYrBlt)] = "no_garage"
#Clean up GarageCond
train_data$GarageCond[is.na(train_data$GarageCond)] = "no_garage"
test_data$GarageCond[is.na(test_data$GarageCond)] = "no_garage"
#Impute the missing values of LotFrontage with the median of the available values
train_data$LotFrontage[is.na(train_data$LotFrontage)] = median(train_data$LotFrontage, na.rm = TRUE)
test_data$LotFrontage[is.na(test_data$LotFrontage)] = median(test_data$LotFrontage, na.rm = TRUE)
#Replace the NA in electrl with the most frequent catagory
electrical_mode = names(which.max(table(train_data$Electrical)))
train_data$Electrical[is.na(train_data$Electrical)] = electrical_mode
electrical_mode = names(which.max(table(test_data$Electrical)))
test_data$Electrical[is.na(test_data$Electrical)] = electrical_mode
# Convert variables to factors and drop unused levels
train_data_clean = train_data %>% mutate_if(is.character, as.factor) %>% droplevels()
test_data_clean = test_data %>% mutate_if(is.character, as.factor) %>% droplevels()
# Update the NA count - train set
na_counts = colSums(is.na(train_data))
na_counts

# Update the NA count - test set
na_counts = colSums(is.na(test_data))
na_counts
#convert variables to factors
train_data = train_data %>% mutate_if(is.character, as.factor)
test_data = test_data %>% mutate_if(is.character, as.factor)


# Convert categorical variables to dummy variables - Training Set
train_data_dummy = train_data[, -which(names(train_data) == "SalePrice")]
dummy_vars = as.data.frame(predict(dummyVars(~., data = train_data_dummy), train_data_dummy))
cat("Dummy variables dimensions after model.matrix():", dim(dummy_vars), "\n")
print(colnames(train_data_clean))
write.csv(train_data_clean, file = "train_data_clean.csv")

# Convert categorical variables to dummy variables - test Set
test_data_dummy = test_data[, -which(names(test_data) == "SalePrice")]
dummy_vars = as.data.frame(predict(dummyVars(~., data = test_data_dummy), test_data_dummy))
cat("Dummy variables dimensions after model.matrix():", dim(dummy_vars), "\n")
print(colnames(test_data_clean))


# Generate forward, Backward, and stepwise model

fit_home = lm(SalePrice~., data = train_data)

# forward model & Cv press

ols_step_forward_p(fit_home,penter = .05,details = TRUE)
# generated forward model
forward_model = lm(SalePrice~ OverallQual+ GrLivArea+ Neighborhood+ BsmtQual+RoofMatl+
                     BsmtFinSF1+ MSSubClass+ BsmtExposure+ KitchenQual+ Condition2+ OverallCond+
                     YearBuilt+ LotArea+ SaleCondition+ PoolQC+ GarageCars+ ExterQual+ TotalBsmtSF+
                     BldgType+ Functional+ BedroomAbvGr+ PoolArea+ ScreenPorch+ LowQualFinSF+ Condition1+
                     MasVnrArea+ LandSlope+ Exterior1st+ GarageQual+ LandContour+ Street+ LotConfig+ GarageCond+
                     MSZoning+ BsmtFullBath , data = train_data,trControl= train_control, method = "lm")

# cv press of forward model

#define training control
train_control= trainControl(method = "LOOCV")

#train the forward model
forward_modelT = train(SalePrice ~ OverallQual+ GrLivArea+ Neighborhood+ BsmtQual+RoofMatl+
                         BsmtFinSF1+ MSSubClass+ BsmtExposure+ KitchenQual+ Condition2+ OverallCond+
                         YearBuilt+ LotArea+ SaleCondition+ PoolQC+ GarageCars+ ExterQual+ TotalBsmtSF+
                         BldgType+ Functional+ BedroomAbvGr+ PoolArea+ ScreenPorch+ LowQualFinSF+ Condition1+
                         MasVnrArea+ LandSlope+ Exterior1st+ GarageQual+ LandContour+ Street+ LotConfig+ GarageCond+
                         MSZoning+ BsmtFullBath , data = train_data,trControl= train_control, method = "lm")
forward_modelT
```

```
    color_gradient = colorRampPalette(c("blue", "red"))(n = 100)
    point_colors = color_gradient[cut(neighborhood_data$SalePrice,
                                      breaks = 100,
                                      labels = FALSE,
                                      include.lowest = TRUE)]

    model = lm(SalePrice ~ GrLivArea, data = neighborhood_data)

    plot_ly(neighborhood_data,
            x = ~GrLivArea,
            y = ~SalePrice,
            type = "scatter",
            mode = "markers",
            marker = list(color = point_colors),
            text = ~paste("Price:", SalePrice, "<br>", "Square Footage:", GrLivArea, "<br>", "Year Built:", YearBuilt),
            hoverinfo = "text") %>%
      add_trace(
        x = ~GrLivArea,
        y = fitted(model),
        type = "scatter",
        mode = "lines",
        line = list(color = 'black'),
        hoverinfo = "none",
        showlegend = FALSE
      ) %>%
      layout(title = paste("Scatterplot of Home Prices vs. Square Footage in", selected_neighborhood),
             xaxis = list(title = "Square Footage (GrLivArea)"),
             yaxis = list(title = "Home Price"))
  })

  output$model_stats <- renderText({
    selected_neighborhood = input$neighborhood
    year_range = input$year_range

    neighborhood_data = train_data[train_data$Neighborhood == selected_neighborhood & train_data$YearBuilt >= year_range[1] & train_data$YearBuilt <= year_range[2],]

    model = lm(SalePrice ~ GrLivArea, data = neighborhood_data)

    model_summary <- summary(model)

    r_squared <- model_summary$r.squared
    p_value <- model_summary$coefficients[2, 4]

    paste("R-squared:", round(r_squared, 3), " | p-value:", formatC(p_value, format = "e", digits = 4))
  })

}

shinyApp(ui = ui, server = server)
```

```
#### Backward Model & CV press

ols_step_backward_p(fit_home,prem = .05, details = TRUE)

#Final model
Backward_model = lm(SalePrice~LotArea+ Street+ LandContour+  LotConfig+ LandSlope+
                      Neighborhood+ Condition1+ Condition2+ BldgType+ OverallQual+ OverallCond+
                      YearBuilt+ YearRemodAdd+ RoofMatl+ ExteriorIst+ ExterQual+ BsmtQual+  BsmtExposure+
                      BsmtFinSF1+ BsmtFinSF2+ BsmtUnfSF+ TotalBsmtSF+ X1stFlrSF+ X2ndFlrSF+ LowQualFinSF+ GrLivArea+
                      BedroomAbvGr+ KitchenQual+  Functional+ GarageCars+ GarageArea+ GarageQual+ GarageCond+ WoodDeckSF+
                      ScreenPorch+ PoolArea+ PoolQC+SaleCondition, data= train_data)
summary(Backward_model)

#train the Backward model
Backward_modelT = train(SalePrice ~LotArea+ Street+ LandContour+  LotConfig+ LandSlope+ Neighborhood+ Condition1+ Condition2+
                          BldgType+ OverallQual+ OverallCond+ YearBuilt+ YearRemodAdd+ RoofMatl+ ExteriorIst+ ExterQual+ BsmtQual+
                          BsmtExposure+  BsmtFinSF1+ BsmtFinSF2+ BsmtUnfSF+ TotalBsmtSF+ X1stFlrSF+ X2ndFlrSF+ LowQualFinSF+ GrLivArea+
                          BedroomAbvGr+ KitchenQual+  Functional+ GarageCars+ GarageArea+ GarageQual+ GarageCond+ WoodDeckSF+ ScreenPorch+
                          PoolArea+ PoolQC+SaleCondition, data = train_data,trControl= train_control, method = "lm")

Backward_modelT

# stepwise MODEL & CV Press

ols_step_both_p(fit_home,pent = .05, prem = .05, details = TRUE)

#Final model
stepwise_model = lm(SalePrice ~ MSSubClass + MSZoning + LotArea + Street + LandContour +
                      Utilities + LotConfig + LandSlope + Neighborhood + Condition1 + Condition2 +
                      BldgType + HouseStyle + OverallQual + OverallCond + YearBuilt + YearRemodAdd +
                      RoofStyle + RoofMatl + Exterior1st + Exterior2nd + ExterQual + BsmtQual + BsmtCond +
                      BsmtExposure + BsmtFinType1 + BsmtFinSF1 + BsmtFinType2 + BsmtFinSF2 + BsmtUnfSF +
                      Heating + Electrical + X1stFlrSF + X2ndFlrSF + FullBath + BedroomAbvGr + KitchenAbvGr +
                      KitchenQual + TotRmsAbvGrd + Functional + Fireplaces + GarageType + GarageFinish + GarageCars +
                      GarageArea + GarageQual + GarageCond + WoodDeckSF + X3SsnPorch + ScreenPorch + PoolArea + PoolQC +
                      Fence + MoSold + YrSold + SaleType + SaleCondition + Electrical +
                      BsmtCond + Foundation + ExteriorIst, data = train_data)
summary(stepwise_model)

#define training control
train_control= trainControl(method = "LOOCV")

#train stepwise model
Stepwise_modelT = train(SalePrice ~ MSSubClass + MSZoning + LotArea + Street + LandContour +
                          Utilities + LotConfig + LandSlope + Neighborhood + Condition1 + Condition2 +
                          BldgType + HouseStyle + OverallQual + OverallCond + YearBuilt + YearRemodAdd +
                          RoofStyle + RoofMatl + Exterior1st + Exterior2nd + ExterQual + BsmtQual + BsmtCond +
                          BsmtExposure + BsmtFinType1 + BsmtFinSF1 + BsmtFinType2 + BsmtFinSF2 + BsmtUnfSF +
                          Heating + Electrical + X1stFlrSF + X2ndFlrSF + FullBath + BedroomAbvGr + KitchenAbvGr +
                          KitchenQual + TotRmsAbvGrd + Functional + Fireplaces + GarageType + GarageFinish + GarageCars +
                          GarageArea + GarageQual + GarageCond + WoodDeckSF + X3SsnPorch + ScreenPorch + PoolArea + PoolQC +
                          Fence + MoSold + YrSold + SaleType + SaleCondition + Electrical +
                          BsmtCond + Foundation + ExteriorIst, data = train_data,trControl= train_control, method = "lm")

Stepwise_modelT

#use models to predict Saleprice of test dataset

#forward results
test_data1$SalePrice = predict(forward_model, newdata = test_data)

average_price = mean(test_data1$SalePrice, na.rm = TRUE)

# replace NAs in the SalePrice column with the average value
test_data1$SalePrice[is.na(test_data1$SalePrice)] <- average_price

resultsF <- test_data[, c("Id", "SalePrice")]
write.csv(resultsF, file = "resultsF.csv", row.names = FALSE)

#Backwards results

test_data1$SalePrice = predict(Backward_model, newdata = test_data)

average_price = mean(test_data1$SalePrice, na.rm = TRUE)

# replace NAs in the SalePrice column with the average value
test_data1$SalePrice[is.na(test_data1$SalePrice)] <- average_price

#convert results to csv
resultsB <- test_data[, c("Id", "SalePrice")]
write.csv(resultsB, file = "resultsB.csv", row.names = FALSE)


#Stepwise results
test_data1$SalePrice = predict(stepwise_model, newdata = test_data)

average_price= mean(test_data1$SalePrice, na.rm = TRUE)


# replace NAs in the SalePrice column with the average value
test_data1$SalePrice[is.na(test_data1$SalePrice)] <- average_price

#convert results to csv
resultsS <- test_data2[, c("Id", "SalePrice")]
write.csv(results5, file = "results5.csv", row.names = FALSE)


# custom model

custom_model = lm(SalePrice~GrLivArea+ Neighborhood+ LotArea+ CentralAir+ KitchenQual+GarageArea+ YearBuilt, data = train_data)
summary(custom_model)

# check cvpress for model
custom_modelT = train(SalePrice ~GrLivArea+ Neighborhood+ LotArea+ CentralAir+ KitchenQual+
                        GarageArea+ YearBuilt, data = train_data,trControl= train_control, method = "lm")
custom_modelT

test_data1$SalePrice = predict(custom_model, newdata = test_data)

average_price = mean(test_data1$SalePrice, na.rm = TRUE)


# replace NAs in the SalePrice column with the average value
test_data1$SalePrice[is.na(test_data1$SalePrice)] <- average_price

#convert results to csv
resultsC <- test_data[, c("Id", "SalePrice")]
write.csv(resultsC, file = "resultsC.csv", row.names = FALSE)

# check assumptions of custom model

plot(custom_model, which = 1:4)
```

References

Kaggle. House Prices: Advanced Regression Techniques. Retrieved April 10, 2023, from https://www.kaggle.com/c/house-prices-advanced-regression-techniques

R-Shiny App

https://8rj3po-oneal-gray.shinyapps.io/Housing/

GitHub Pages

O’neal: https://rogray85.github.io/

David: https://matthewdavid210.github.io/