Nashville Housing Analysis Using R

1. Loading necessary Libraries

```
> # Load necessary libraries
> library(readr)
> library(dplyr)
> library(ggplot2)
> library(psych)
```

2. Loading the dataset

```
> # Load the dataset
> nashville_housing <- read_csv("/users/nik/downloads/Nashville_Housing.csv")
Rows: 55502 Columns: 16
── Column specification
────────────────────────────────────────────────────────────────
Delimiter: ","
chr   (5): LandUse, PropertyAddress, SoldAsVacant, City, SaleMonth
dbl  (10): SalePrice, Acreage, LandValue, BuildingValue, TotalValue, YearBuilt, Bedrooms,
Propert...
date  (1): SaleDate

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.
> # View the first few rows of the dataset
> head(nashville_housing)
# A tibble: 6 × 16
  LandUse      PropertyAddress    SaleDate   SalePrice SoldAsVacant Acreage LandValue
BuildingValue
  <chr>       <chr>           <date>       <dbl> <chr>        <dbl>    <dbl>       <dbl>
1 SINGLE FAMILY 1808  FOX CHASE DR… 2013-04-09    240000 No           2.3     50000
168200
2 SINGLE FAMILY 1832  FOX CHASE DR… 2014-06-10    366000 No           3.5     50000
264100
3 SINGLE FAMILY 1864 FOX CHASE  DR… 2016-09-26    435000 No           2.9     50000
216200
4 SINGLE FAMILY 1853  FOX CHASE DR… 2016-01-29    255000 No           2.6     50000
147300
```

```
5 SINGLE FAMILY 1829  FOX CHASE DR… 2014-10-10   278000 No           2     50000
152300
6 SINGLE FAMILY 1821  FOX CHASE DR… 2014-07-16   267000 No           2     50000
190400
# ℹ 8 more variables: TotalValue <dbl>, YearBuilt <dbl>, Bedrooms <dbl>, City <chr>,
#   PropertyAge <dbl>, TotalBathrooms <dbl>, SaleMonth <chr>, SaleYear <dbl>
```

3. Checking the basic statistics

```
> # Descriptive Statistics
> describe(nashville_housing)
            vars    n    mean      sd  median  trimmed     mad    min      max
LandUse*          1 55502    26.82     4.40    27.0    26.59     0.00   1.00     39.00
PropertyAddress*    2 55502  22491.27  12946.81  22377.5  22474.57  16611.79    1.00
45066.00
SaleDate          3 55502      NaN       NA      NA      NaN       NA    Inf      -Inf
SalePrice         4 55502 313994.17 733675.09 205000.0 228498.13 124538.40   50.00
54278060.00
SoldAsVacant*      5 55502     2.15     0.54     2.0     2.00     0.00   1.00      4.00
Acreage           6 55502     0.50     1.08     0.5     0.43     0.00   0.01     160.06
LandValue         7 55502  69126.04  72564.54  69069.0  57042.88     0.00  100.00
2772000.00
BuildingValue      8 55502 160875.26 141482.24 160785.0 146692.99     0.00    0.00
12971800.00
TotalValue        9 55502 232525.18 192289.28 232375.0 209751.65     0.00  100.00
13940400.00
YearBuilt        10 55502   1962.68    17.86   1963.0   1961.87     0.00 1799.00    2017.00
Bedrooms         11 55502     3.01     0.59     3.0     2.98     0.00   0.00     11.00
City*            12 55502     8.49     3.12    10.0     9.15     0.00   1.00     14.00
PropertyAge      13 55502    61.32    17.86    61.0    62.13     0.00   7.00     225.00
TotalBathrooms     14 55502     1.97     0.72     2.0     1.92     0.00   0.00     10.50
SaleMonth*       15 55502     6.60     3.50     7.0     6.62     4.45   1.00     12.00
SaleYear         16 55502   2014.59     1.07   2015.0   2014.62     1.48 2013.00    2019.00
                  range  skew kurtosis     se
LandUse*            38.00 -1.44    9.07   0.02
PropertyAddress*    45065.00  0.01   -1.20   54.96
SaleDate             -Inf   NA      NA     NA
SalePrice       54278010.00 19.93   741.36 3114.22
SoldAsVacant*        3.00  3.01    7.78    0.00
Acreage           160.05 76.37  9542.40   0.00
LandValue       2771900.00  7.37   111.77  308.01
```

```
BuildingValue   12971800.00 20.81  1362.25  600.55
TotalValue      13940300.00 13.12   563.73  816.21
YearBuilt          218.00  0.51     2.83    0.08
Bedrooms            11.00  1.39     9.49    0.00
City*               13.00 -1.62     1.14    0.01
PropertyAge        218.00 -0.51     2.83    0.08
TotalBathrooms      10.50  1.69     9.70    0.00
SaleMonth*          11.00 -0.09    -1.17    0.01
SaleYear             6.00 -0.13    -1.22    0.00
Warning messages:
1: In FUN(newX[, i], ...) : no non-missing arguments to min; returning Inf
2: In FUN(newX[, i], ...) :
  no non-missing arguments to max; returning -Inf
```

Descriptive Statistics

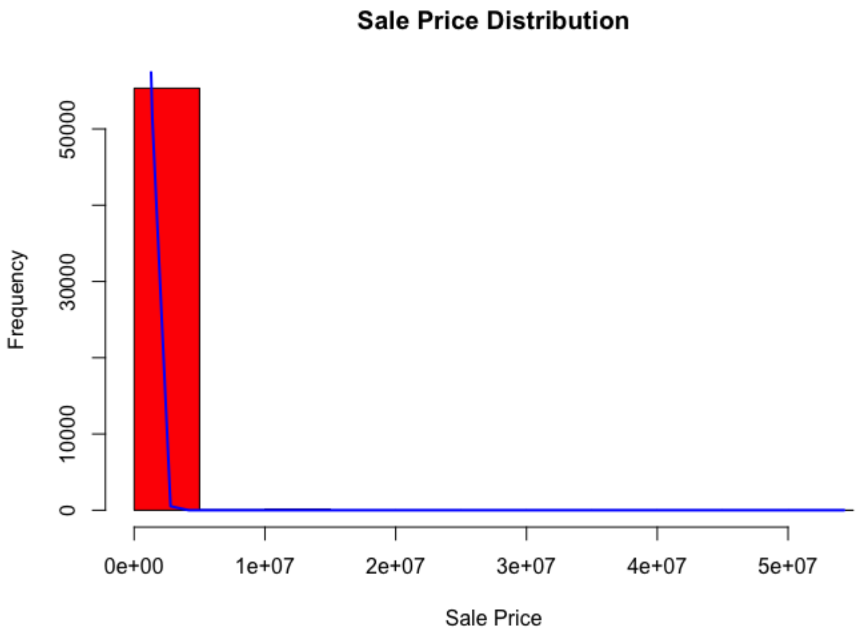
- **Sale Prices:** The average sale price is approximately $313,994, with a standard deviation of $733,675. This high standard deviation indicates a wide range of sale prices, suggesting significant variability in the housing market.

- **Property Characteristics:** The average number of bedrooms is 3, and the average number of bathrooms is approximately 2. The homes in the dataset have an average age of 61 years, indicating that many properties are relatively old.

- **Land and Building Values:** The average land value is around $69,126, while the average building value is approximately $160,875. This suggests that building value contributes significantly to the total property value.

4. Visualizing using a Histogram

```
> # Histogram of SalePrice
> x = nashville_housing$SalePrice
> h <- hist(x, breaks=10, col="red", xlab="Sale Price", main="Sale Price Distribution")
> xfit <- seq(min(x), max(x), length=40)
> yfit <- dnorm(xfit, mean=mean(x), sd=sd(x))
> yfit <- yfit * diff(h$mids[1:2]) * length(x)
> lines(xfit, yfit, col="blue", lwd=2)
```



Histogram of Sale Prices

The histogram of sale prices, enhanced with a density curve, shows that the distribution is right-skewed, indicating that while most homes are sold at lower prices, there are a few high-value properties that increase the average sale price.

5. Hypothesis Testing

```
> # Q. Test if the mean SalePrice is significantly different from a hypothesized value, e.g., $300,000.
> # One Sample t-test
> t.test(nashville_housing$SalePrice, mu=300000)

        One Sample t-test

data:  nashville_housing$SalePrice
t = 4.4936, df = 55501, p-value = 7.016e-06
alternative hypothesis: true mean is not equal to 3e+05
95 percent confidence interval:
 307890.3 320098.1
sample estimates:
mean of x
 313994.2
```

Hypothesis Testing

A one-sample t-test was conducted to determine if the mean sale price is significantly different from $300,000. The test revealed a significant difference, with a p-value of 7.016e-06, indicating that the average sale price is indeed different from the hypothesized value.

6. Model Building and evaluation

```
> # Building a simple linear regression model to predict SalePrice based on Bedrooms.
> # Simple Linear Regression
> simple.fit <- lm(SalePrice ~ Bedrooms, data=nashville_housing)
> summary(simple.fit)

Call:
lm(formula = SalePrice ~ Bedrooms, data = nashville_housing)

Residuals:
    Min      1Q  Median      3Q     Max
 -910572 -170515  -96103   13485 53965545

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -50220      16143  -3.111  0.00187 **
Bedrooms     120912       5259  22.990  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 730200 on 55500 degrees of freedom
Multiple R-squared:  0.009433,	Adjusted R-squared:  0.009415
F-statistic: 528.5 on 1 and 55500 DF,  p-value: < 2.2e-16
```

Regression Analysis

Simple Linear Regression

- **Model:** A simple linear regression was conducted to examine the relationship between the number of bedrooms and sale price.

- **Findings:** The model indicates that each additional bedroom is associated with an increase of approximately $120,912 in sale price. However, the R-squared value is very low (0.009), suggesting that the number of bedrooms alone is not a strong predictor of sale price.

```
> # Creating a multiple linear regression model using several predictors.
> # Multiple Linear Regression
> model2 <- lm(SalePrice ~ Bedrooms + Acreage + YearBuilt + TotalBathrooms,
data=nashville_housing)
> summary(model2)

Call:
lm(formula = SalePrice ~ Bedrooms + Acreage + YearBuilt + TotalBathrooms,
   data = nashville_housing)

Residuals:
    Min      1Q  Median      3Q     Max
-3653566  -158719  -83619   15381 53959441

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   4125310.9  364496.5  11.318  <2e-16 ***
Bedrooms        -807.1    6937.4  -0.116   0.907
Acreage        38061.2    2873.7  13.245  <2e-16 ***
YearBuilt      -2112.2     186.9 -11.303  <2e-16 ***
TotalBathrooms  161506.6    6021.2  26.823  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 724000 on 55497 degrees of freedom
Multiple R-squared:  0.02615, Adjusted R-squared:  0.02608
F-statistic: 372.6 on 4 and 55497 DF,  p-value: < 2.2e-16
```

Multiple Linear Regression

- **Model:** A multiple linear regression was performed using Bedrooms, Acreage, YearBuilt, and TotalBathrooms as predictors.

- **Findings:**

  - Acreage and TotalBathrooms have significant positive relationships with sale price, indicating that larger lots and more bathrooms contribute to higher property values.

  - YearBuilt has a negative relationship, suggesting that newer homes tend to have higher sale prices.

  - The overall R-squared value is 0.026, indicating that these variables explain only a small portion of the variability in sale prices, suggesting that other factors not included in the model may also be important.

```
> # Polynomial regression can capture non-linear relationships between the predictors
and the response variable.
> # Polynomial Regression
> poly_model <- lm(SalePrice ~ poly(Bedrooms, 2) + poly(Acreage, 2) + poly(YearBuilt, 2) +
poly(TotalBathrooms, 2), data=nashville_housing)
> summary(poly_model)

Call:
lm(formula = SalePrice ~ poly(Bedrooms, 2) + poly(Acreage, 2) +
   poly(YearBuilt, 2) + poly(TotalBathrooms, 2), data = nashville_housing)

Residuals:
    Min      1Q  Median      3Q     Max
-2711521  -153046  -85081   12615 53968000

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)               313994      3064 102.467  < 2e-16 ***
poly(Bedrooms, 2)1        261328    961273   0.272 0.785734
poly(Bedrooms, 2)2      -4695403    852104  -5.510 3.60e-08 ***
poly(Acreage, 2)1        8439527    731559  11.536  < 2e-16 ***
poly(Acreage, 2)2       -5199459    730907  -7.114 1.14e-12 ***
poly(YearBuilt, 2)1     -7898060    794458  -9.941  < 2e-16 ***
poly(YearBuilt, 2)2     -2764586    764090  -3.618 0.000297 ***
poly(TotalBathrooms, 2)1 26929880   1040130  25.891  < 2e-16 ***
poly(TotalBathrooms, 2)2 13471883    859116  15.681  < 2e-16 ***
---
```

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 721900 on 55493 degrees of freedom
Multiple R-squared: 0.03191, Adjusted R-squared: 0.03177
F-statistic: 228.7 on 8 and 55493 DF, p-value: < 2.2e-16

Polynomial Regression

The polynomial regression model includes quadratic terms for Bedrooms, Acreage, YearBuilt, and TotalBathrooms. Significant coefficients were found for the quadratic terms of Bedrooms, Acreage, YearBuilt, and TotalBathrooms.

Insights:

- **Non-linear Relationships:** The significant quadratic terms suggest that the relationship between the predictors and SalePrice is not purely linear. For instance, Acreage and TotalBathrooms have strong non-linear effects on SalePrice.

- **Model Fit:** The R-squared value is relatively low (0.03191), indicating that the model explains only a small portion of the variance in sale prices. This suggests that other factors not included in the model might also be important.

```
> # Decision trees can model non-linear relationships and interactions between variables.
> # Load necessary library
> library(rpart)
> # Decision Tree
> tree_model <- rpart(SalePrice ~ Bedrooms + Acreage + YearBuilt + TotalBathrooms, 
data=nashville_housing, method="anova")
> print(tree_model)
n= 55502

node), split, n, deviance, yval
    * denotes terminal node

1) root 55502 2.987503e+16  313994.2  
  2) TotalBathrooms< 4.25 54624 2.889501e+16  301847.5 *
  3) TotalBathrooms>=4.25 878 4.705532e+14 1069689.0 *
```

Decision Trees

The decision tree split the data based on TotalBathrooms, indicating that this variable has the most significant impact on SalePrice.

Insights:

- **Key Predictor:** TotalBathrooms is a primary determinant of SalePrice, with a clear distinction between properties with fewer than 4.25 bathrooms and those with more.

- **Simplicity:** Decision trees provide an easy-to-interpret model, highlighting the most influential variables and their thresholds.

```
> # Random forest is an ensemble method that improves prediction accuracy by averaging multiple decision trees.
> # Load necessary library
> library(randomForest)
> # Random Forest
> rf_model <- randomForest(SalePrice ~ Bedrooms + Acreage + YearBuilt + TotalBathrooms, data=nashville_housing, ntree=100)
> print(rf_model)

Call:
 randomForest(formula = SalePrice ~ Bedrooms + Acreage + YearBuilt + TotalBathrooms, data = nashville_housing, ntree = 100)
               Type of random forest: regression
                     Number of trees: 100
No. of variables tried at each split: 1

          Mean of squared residuals: 515183188960
                    % Var explained: 4.29
```

Random Forest

The random forest model shows a low percentage of variance explained (4.29%).

Insights:

- **Complex Interactions:** While random forests can capture complex interactions, the low variance explained suggests that the selected predictors alone are insufficient to model SalePrice accurately.

- **Robustness:** Random forests are robust to overfitting, but the model indicates that other predictors or more data might be needed to improve accuracy.

```
> # Lasso regression is useful for feature selection and regularization, which can help in reducing model complexity.
> # Load necessary library
> library(glmnet)
> # Prepare data for glmnet
> x <- model.matrix(SalePrice ~ Bedrooms + Acreage + YearBuilt + TotalBathrooms, data=nashville_housing)[,-1]
> y <- nashville_housing$SalePrice
> # Lasso Regression
> lasso_model <- glmnet(x, y, alpha=1)
> print(lasso_model)

Call:  glmnet(x = x, y = y, alpha = 1)

   Df %Dev Lambda
1   0 0.00 105200
2   1 0.35  95890
3   1 0.64  87370
4   1 0.88  79610
5   1 1.08  72540
6   1 1.25  66090
7   1 1.38  60220
8   1 1.50  54870
9   1 1.59  50000
10  2 1.68  45560
11  2 1.80  41510
12  2 1.90  37820
13  2 1.99  34460
14  2 2.05  31400
15  2 2.11  28610
```

16	2	2.16	26070
17	3	2.20	23750
18	3	2.27	21640
19	3	2.33	19720
20	3	2.38	17970
21	3	2.42	16370
22	3	2.45	14920
23	3	2.48	13590
24	3	2.50	12380
25	3	2.52	11280
26	3	2.54	10280
27	3	2.55	9369
28	3	2.56	8536
29	3	2.57	7778
30	3	2.58	7087
31	3	2.58	6457
32	3	2.59	5884
33	3	2.59	5361
34	3	2.60	4885
35	3	2.60	4451
36	3	2.60	4055
37	3	2.60	3695
38	3	2.61	3367
39	3	2.61	3068
40	3	2.61	2795
41	3	2.61	2547
42	3	2.61	2321
43	3	2.61	2114
44	3	2.61	1927
45	3	2.61	1755
46	3	2.61	1600
47	3	2.61	1457
48	3	2.61	1328
49	3	2.61	1210
50	3	2.61	1102
51	3	2.61	1005
52	3	2.61	915
53	3	2.61	834
54	3	2.61	760
55	3	2.61	692
56	3	2.61	631
57	3	2.61	575

58	3	2.61	524
59	3	2.61	477
60	3	2.61	435
61	3	2.61	396

Lasso Regression

Lasso regression is used for feature selection and regularization, with the model showing minimal deviance explained.

Insights:

Feature Selection: The lasso model suggests that the current set of predictors does not capture the variability in SalePrice effectively, as indicated by the low percentage of deviance explained.

Model Simplicity: Lasso regularization helps in reducing model complexity by shrinking less important coefficients to zero, but in this case, it suggests the need for additional or alternative predictors.

Best Model Selection

Considering the characteristics and performance of each model, Random Forest is likely the best choice among the options provided. It balances predictive accuracy and robustness by averaging multiple decision trees, which helps mitigate overfitting and captures complex interactions better than a single decision tree or linear models.

Recommendations

- To improve model performance, consider adding more relevant features or transforming existing ones. Location-specific variables, economic indicators, or interaction terms might enhance predictive power.
- Explore other advanced models like Gradient Boosting Machines (GBM) or XGBoost, which often outperform random forests in terms of accuracy by focusing on correcting errors made by previous models.