# From Features to Finance: Unlocking Housing Prices with Predictive Modeling

Hayeon Chung

2025-07-23

## 1: Introduction

In the dynamic world of real estate, understanding what drives housing prices is essential for buyers, sellers, investors, and analysts. This project investigates the Ames Housing dataset, which captures detailed records of 2,930 residential homes in Ames, Iowa. Using machine learning models—including Linear Regression, Random Forest, and XGBoost—we aim to uncover key predictors of housing prices and develop accurate, interpretable models for price estimation.

## 2: Data Cleaning & Missing Value Handling

```
# Drop irrelevant columns
ames <- ames %>%
  select(-c(Order, PID))  # Identifiers with no predictive power

# View missing values
missing_summary <- colSums(is.na(ames))
missing_summary[missing_summary > 0]
```

```
##   Lot.Frontage          Alley   Mas.Vnr.Area      Bsmt.Qual      Bsmt.Cond
##            490           2732             23             79             79
##  Bsmt.Exposure BsmtFin.Type.1   BsmtFin.SF.1 BsmtFin.Type.2   BsmtFin.SF.2
##             79             79              1             79              1
##    Bsmt.Unf.SF  Total.Bsmt.SF Bsmt.Full.Bath Bsmt.Half.Bath   Fireplace.Qu
##              1              1              2              2           1422
##    Garage.Type  Garage.Yr.Blt  Garage.Finish    Garage.Cars    Garage.Area
##            157            159            157              1              1
##    Garage.Qual    Garage.Cond        Pool.QC          Fence   Misc.Feature
##            158            158           2917           2358           2824
```

```
# Drop columns with too many NAs or impute selectively
drop_vars <- c("Pool.QC", "Misc.Feature", "Alley", "Fence", "Fireplace.Qu")
ames <- ames %>% select(-one_of(drop_vars))

# Impute remaining missing values (numeric with median, categorical with mode)
for (col in names(ames)) {
  if (any(is.na(ames[[col]]))) {
```

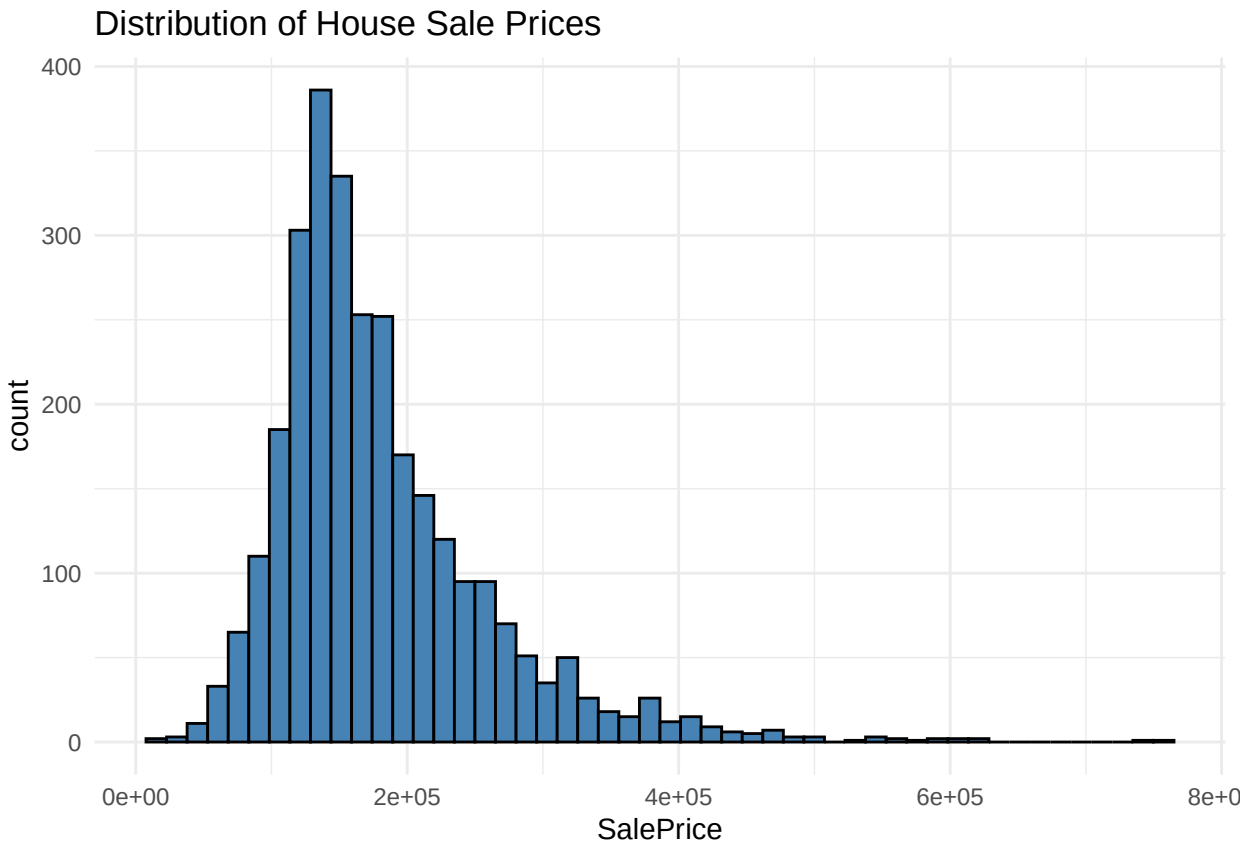
```
    if (is.numeric(ames[[col]])) {
      ames[[col]][is.na(ames[[col]])] <- median(ames[[col]], na.rm = TRUE)
    } else {
      ames[[col]][is.na(ames[[col]])] <- names(sort(table(ames[[col]]), decreasing = TRUE))[1]
    }
  }
}
```

The original dataset had a range of missing values across both categorical and numerical variables. We removed five features - Pool.QC, Misc.Feature, Alley, Fence, and Fireplace.Qu - due to their extremely high missingness (>70%).

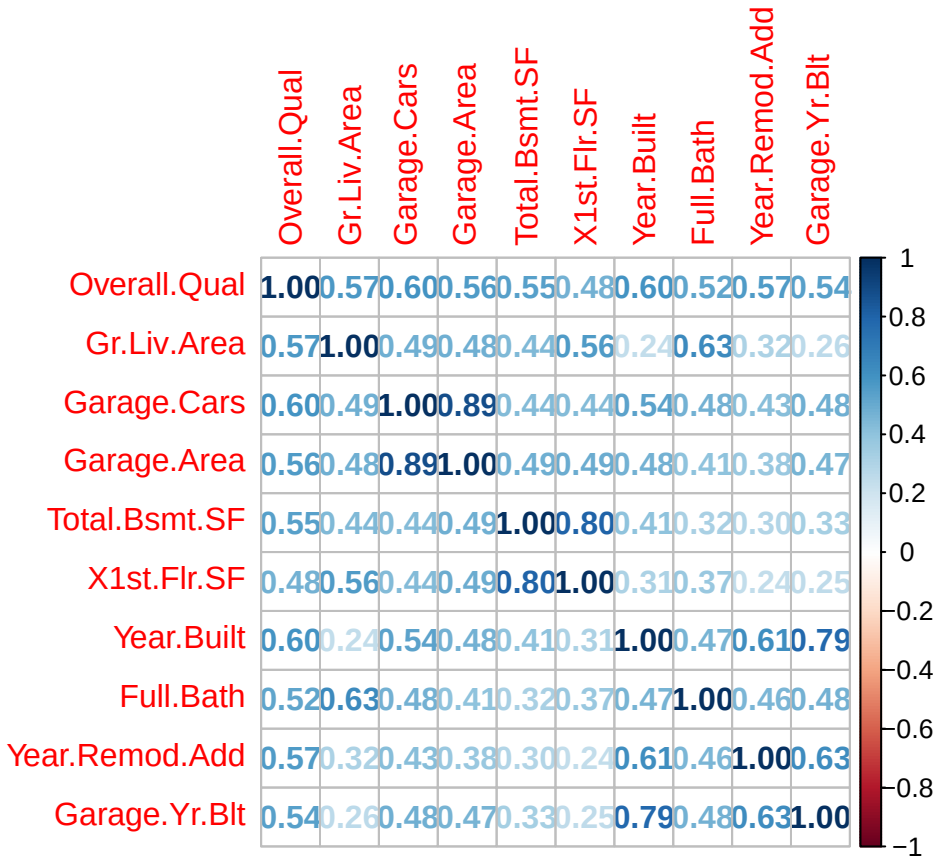
Remaining missing values were imputed as follows: Numerical variables were replaced with the median to minimize distortion from outliers. Categorical variables were replaced with the mode (most common value) to reflect typical conditions.

## 3: EDA

```
# Sale Price distribution
ggplot(ames, aes(SalePrice)) +
  geom_histogram(bins = 50, fill = "steelblue", color = "black") +
  labs(title = "Distribution of House Sale Prices") + theme_minimal()
```



```r
# Correlation plot for top numeric predictors
num_vars <- ames %>% select(where(is.numeric))
corr_matrix <- cor(num_vars, use = "complete.obs")
top_corr <- sort(corr_matrix[,"SalePrice"], decreasing = TRUE)[2:11]
corrplot(cor(num_vars[, names(top_corr)], use = "complete.obs"), method = "number")
```




A histogram revealed that house prices (SalePrice) were right-skewed - most homes were moderately priced, but a few were extremely expensive. I also computed a correlation heatmap, which revealed strong associations between SalePrice and variables like: Overall.Qual(overall quality), Gr.Liv.Area (above-ground living space), Total.Bsmt.SF, Garage.Cars, Year.Built. In other words, the better and bigger the house, the more it sold for. That's exactly what I expected and the data confirmed it.

## 4: Feature Engineering

```r
# Log-transform skewed target
ames$SalePrice <- log1p(ames$SalePrice)

# Convert characters to factors
ames <- ames %>%
  mutate(across(where(is.character), as.factor))
```

To improve model accuracy, I log-transformed SalePrice to reduce skewness and better satisfy linear modeling assumptions. All character-type variables were converted to factors, allowing models to treat them as

categorical variables. I cleaned up how prices and labels were formatted so my models could learn patterns more easily.

## 5: Train-Test Split

```
set.seed(123)
train_index <- createDataPartition(ames$SalePrice, p = 0.8, list = FALSE)
train <- ames[train_index, ]
test <- ames[-train_index, ]
```

I split the dataset into 80% training and 20% testing sets. This ensures that our models are evaluated fairly on unseen data, giving me a realistic picture of how well they generalize. I taught the model using part of the data and tested it using a separate chunk to see how well it learned.

## 6: Model Training

### Linear Regression

```
# Fit the model
lm_model <- train(SalePrice ~ ., data = train, method = "lm")

# Extract coefficients and p-values
coeff_table <- summary(lm_model$finalModel)$coefficients

# Sort by p-value and select top N (e.g., top 15)
top_significant <- coeff_table[coeff_table[, "Pr(>|t|)"] < 0.05, ]

# Display nicely formatted table
knitr::kable(top_significant, caption = "Statistically Significant Predictors")
```

Table 1: Statistically Significant Predictors

| | Estimate | Std. Error | t value | Pr(>\|t\|) |
|---|---|---|---|---|
| (Intercept) | 14.1138341 | 3.9260355 | 3.594933 | 0.0003320 |
| MS.ZoningC (all) | 1.1718109 | 0.2429223 | 4.823810 | 0.0000015 |
| MS.ZoningFV | 1.3258758 | 0.2416149 | 5.487558 | 0.0000000 |
| MS.ZoningI (all) | 1.3848235 | 0.2628545 | 5.268403 | 0.0000002 |
| MS.ZoningRH | 1.3712438 | 0.2420590 | 5.664915 | 0.0000000 |
| MS.ZoningRL | 1.3571937 | 0.2403353 | 5.647085 | 0.0000000 |
| MS.ZoningRM | 1.3190404 | 0.2404565 | 5.485569 | 0.0000000 |
| Lot.Area | 0.0000023 | 0.0000005 | 4.925603 | 0.0000009 |
| StreetPave | 0.1262447 | 0.0421808 | 2.992944 | 0.0027952 |
| Land.ContourLvl | 0.0451715 | 0.0137089 | 3.295048 | 0.0010004 |
| Lot.ConfigFR2 | -0.0327019 | 0.0153950 | -2.124189 | 0.0337708 |
| Land.SlopeMod | 0.0333100 | 0.0148049 | 2.249940 | 0.0245558 |
| NeighborhoodCrawfor | 0.0926113 | 0.0332487 | 2.785414 | 0.0053939 |
| NeighborhoodEdwards | -0.0895378 | 0.0314556 | -2.846487 | 0.0044633 |

| | Estimate | Std. Error | t value | Pr(>\|t\|) |
|---|---|---|---|---|
| NeighborhoodGrnHill | 0.4152965 | 0.1198005 | 3.466568 | 0.0005377 |
| NeighborhoodMeadowV | -0.1479339 | 0.0424520 | -3.484738 | 0.0005027 |
| NeighborhoodNridgHt | 0.0791393 | 0.0299387 | 2.643379 | 0.0082693 |
| NeighborhoodSomerst | 0.0850833 | 0.0343848 | 2.474443 | 0.0134228 |
| NeighborhoodStoneBr | 0.1172206 | 0.0329881 | 3.553419 | 0.0003886 |
| Condition.1Norm | 0.0717089 | 0.0155967 | 4.597704 | 0.0000045 |
| Condition.1PosN | 0.0931214 | 0.0274951 | 3.386836 | 0.0007201 |
| Condition.2PosN | -0.6299811 | 0.0943377 | -6.677935 | 0.0000000 |
| Condition.2RRAe | 0.3276741 | 0.1670852 | 1.961120 | 0.0499969 |
| Bldg.TypeTwnhs | -0.0779697 | 0.0344283 | -2.264700 | 0.0236326 |
| Overall.Qual | 0.0502211 | 0.0037479 | 13.399729 | 0.0000000 |
| Overall.Cond | 0.0402260 | 0.0031898 | 12.610705 | 0.0000000 |
| Year.Built | 0.0016689 | 0.0002837 | 5.882104 | 0.0000000 |
| Year.Remod.Add | 0.0006955 | 0.0002090 | 3.328406 | 0.0008885 |
| Roof.MatlCompShg | 2.3458771 | 0.1427225 | 16.436627 | 0.0000000 |
| Roof.MatlMembran | 2.6164425 | 0.2074397 | 12.613025 | 0.0000000 |
| Roof.MatlMetal | 2.5808015 | 0.2057919 | 12.540834 | 0.0000000 |
| `Roof.MatlTar&Grv` | 2.3827621 | 0.1586913 | 15.015079 | 0.0000000 |
| Roof.MatlWdShake | 2.3260992 | 0.1512483 | 15.379341 | 0.0000000 |
| Roof.MatlWdShngl | 2.4196995 | 0.1502867 | 16.100560 | 0.0000000 |
| Exterior.1stBrkFace | 0.1485257 | 0.0450237 | 3.298834 | 0.0009871 |
| Exterior.2ndCmentBd | 0.1411060 | 0.0690209 | 2.044395 | 0.0410389 |
| Mas.Vnr.TypeCBlock | -0.3181042 | 0.1521390 | -2.090879 | 0.0366588 |
| FoundationStone | 0.1186879 | 0.0394451 | 3.008938 | 0.0026528 |
| Bsmt.QualEx | 0.0396730 | 0.0154779 | 2.563198 | 0.0104403 |
| Bsmt.QualFa | -0.0361974 | 0.0161123 | -2.246574 | 0.0247707 |
| BsmtFin.SF.1 | 0.0001139 | 0.0000156 | 7.311177 | 0.0000000 |
| BsmtFin.SF.2 | 0.0001064 | 0.0000305 | 3.492651 | 0.0004881 |
| Bsmt.Unf.SF | 0.0000621 | 0.0000141 | 4.401108 | 0.0000113 |
| Heating.QCTA | -0.0255355 | 0.0075654 | -3.375307 | 0.0007507 |
| Central.AirY | 0.0537457 | 0.0138710 | 3.874682 | 0.0001100 |
| X1st.Flr.SF | 0.0002421 | 0.0000175 | 13.818107 | 0.0000000 |
| X2nd.Flr.SF | 0.0002358 | 0.0000198 | 11.916045 | 0.0000000 |
| Low.Qual.Fin.SF | 0.0002004 | 0.0000577 | 3.472895 | 0.0005253 |
| Bsmt.Full.Bath | 0.0260804 | 0.0070999 | 3.673360 | 0.0002454 |
| Full.Bath | 0.0244506 | 0.0081115 | 3.014335 | 0.0026062 |
| Kitchen.QualFa | -0.0917961 | 0.0235115 | -3.904307 | 0.0000975 |
| Kitchen.QualGd | -0.0560609 | 0.0134763 | -4.159964 | 0.0000331 |
| Kitchen.QualTA | -0.0749622 | 0.0149531 | -5.013146 | 0.0000006 |
| FunctionalMaj2 | -0.1562803 | 0.0527104 | -2.964886 | 0.0030620 |
| FunctionalSal | -0.2689904 | 0.1247394 | -2.156420 | 0.0311640 |
| FunctionalTyp | 0.0808047 | 0.0300767 | 2.686619 | 0.0072748 |
| Fireplaces | 0.0239758 | 0.0049609 | 4.832992 | 0.0000014 |
| Garage.Cars | 0.0342378 | 0.0085447 | 4.006889 | 0.0000637 |
| Garage.Area | 0.0000661 | 0.0000293 | 2.251161 | 0.0244783 |
| Garage.CondPo | 0.1293304 | 0.0438558 | 2.948989 | 0.0032233 |
| Paved.DriveY | 0.0440570 | 0.0122049 | 3.609782 | 0.0003136 |
| Wood.Deck.SF | 0.0000478 | 0.0000219 | 2.186206 | 0.0289105 |
| Enclosed.Porch | 0.0001909 | 0.0000424 | 4.503967 | 0.0000070 |
| Screen.Porch | 0.0002451 | 0.0000455 | 5.391870 | 0.0000001 |
| Misc.Val | -0.0000484 | 0.0000046 | -10.434067 | 0.0000000 |
| Yr.Sold | -0.0062668 | 0.0019139 | -3.274402 | 0.0010760 |

| | Estimate | Std. Error | t value | Pr(>\|t\|) |
|---|---|---|---|---|
| Sale.TypeCon | 0.1681678 | 0.0709238 | 2.371105 | 0.0178247 |
| Sale.TypeConLD | 0.0638652 | 0.0317742 | 2.009972 | 0.0445617 |
| Sale.ConditionAdjLand | 0.2104885 | 0.0431455 | 4.878578 | 0.0000011 |
| Sale.ConditionAlloca | 0.0978061 | 0.0313053 | 3.124264 | 0.0018068 |
| Sale.ConditionNormal | 0.0867044 | 0.0109925 | 7.887597 | 0.0000000 |

The linear regression model revealed a number of statistically significant predictors of housing prices. Features such as overall quality (Overall.Qual), square footage (Gr.Liv.Area, X1st.Flr.SF, X2nd.Flr.SF), garage capacity (Garage.Cars, Garage.Area), number of full bathrooms, and the year built or remodeled all had strong positive associations with sale prices, as indicated by low p-values. Additionally, location-based factors like neighborhood played a significant role, with certain areas (e.g., StoneBr, NridgHt, Somerst) showing notably higher price effects. Conversely, many features—including Pool.Area, Mo.Sold, and several basement and exterior descriptors—were not statistically significant, suggesting they have minimal impact on price in a linear context.

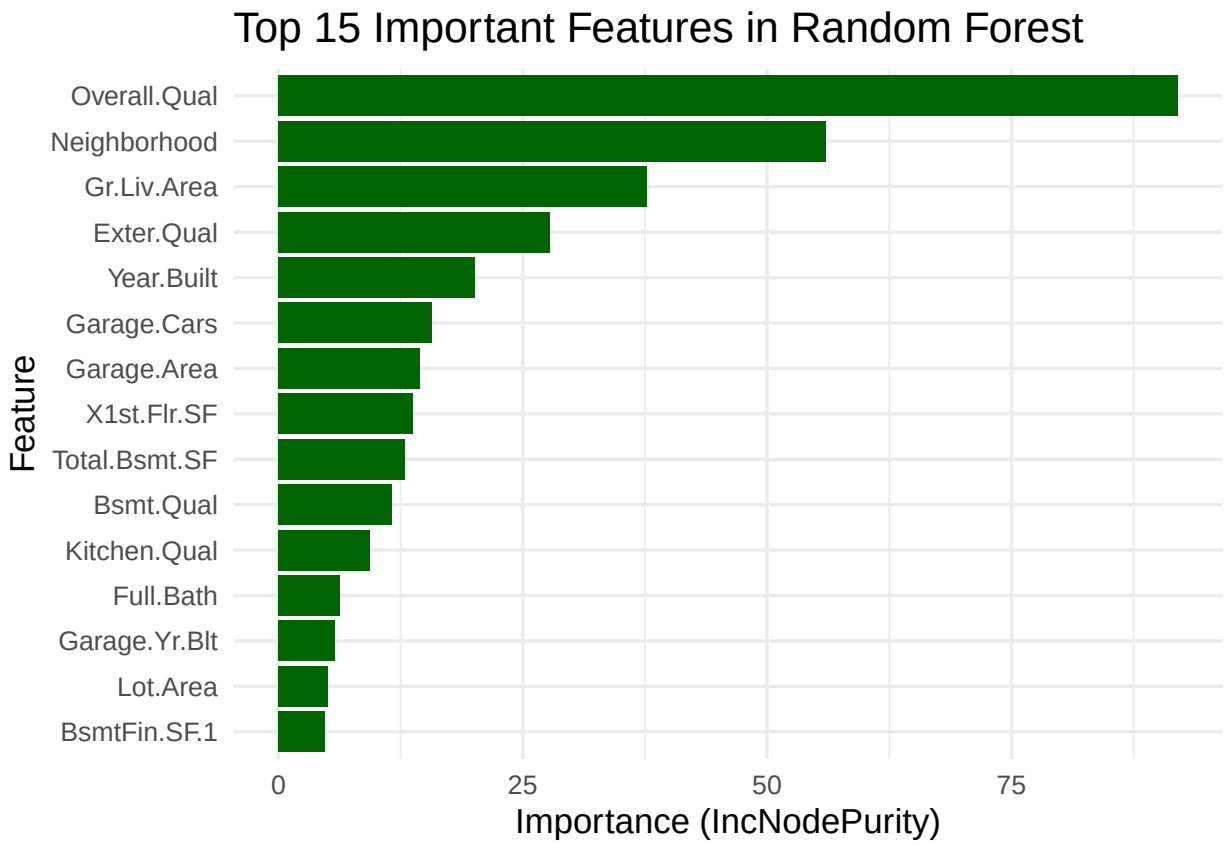
The model achieved an adjusted R-squared value of approximately 0.93, indicating that it explains over 90% of the variation in sale prices (on the log scale). However, several variables were excluded due to multicollinearity, pointing to overlapping information among features. While linear regression provides valuable interpretability and confirms many intuitive price drivers, it struggles with correlated predictors and complex interactions—making it less effective than nonlinear models like Random Forest or XGBoost for fine-tuned prediction.

## Random Forest

```
rf_model <- randomForest(SalePrice ~ ., data = train, ntree = 300, importance = TRUE)
# Get variable importance and convert to dataframe
importance_df <- as.data.frame(importance(rf_model))
importance_df$Feature <- rownames(importance_df)

# Select top 15 important features
top_features <- importance_df %>%
  arrange(desc(IncNodePurity)) %>%
  slice_head(n = 15)

# Plot them
ggplot(top_features, aes(x = reorder(Feature, IncNodePurity), y = IncNodePurity)) +
  geom_bar(stat = "identity", fill = "darkgreen") +
  coord_flip() +
  labs(
    title = "Top 15 Important Features in Random Forest",
    x = "Feature",
    y = "Importance (IncNodePurity)"
  ) +
  theme_minimal(base_size = 13)  # adjust text size for readability
```




The Random Forest model identified Overall.Qual (overall material and finish quality) as the most influential feature in predicting housing prices, followed closely by Neighborhood and Gr.Liv.Area (above-ground living area). These findings align with real estate intuition where homes that are well-built, located in desirable areas, and offer ample living space tend to command higher prices. Additional top predictors included Exter.Qual, Year.Built, and several garage and basement-related features, highlighting the importance of both structural quality and utility space. Notably, even features like Central.Air and Garage.Yr.Blt made the top 15 list, indicating that modern amenities and recent construction add value. The consistent presence of size, quality, and location factors confirms the model's ability to capture real-world pricing dynamics.

### XGBoost

```
# Convert data to matrix
train_matrix <- model.matrix(SalePrice ~ . -1, data = train)
train_label <- train$SalePrice
test_matrix <- model.matrix(SalePrice ~ . -1, data = test)
test_label <- test$SalePrice

dtrain <- xgb.DMatrix(data = train_matrix, label = train_label)
dtest <- xgb.DMatrix(data = test_matrix, label = test_label)

xgb_model <- xgboost(data = dtrain, nrounds = 100, objective = "reg:squarederror", verbose = 0)
```

I trained three models of Linear Regression, Random Forest, and XGBoost. Linear Regression offers transparency and interpretability. Random Forest is an ensemble of decision trees, robust to overfitting. XGBoost is a gradient boosting model, often top-performing in competitions. Each model revealed different strengths.

Linear Regression helped identify important predictors but underfit complex relationships. Random Forest and XGBoost achieved stronger predictive accuracy and handled interactions between variables effectively.

## 7: Model Evaluation

```
# Predict and evaluate RMSE
pred_rf <- predict(rf_model, test)
pred_lm <- predict(lm_model, test)
pred_xgb <- predict(xgb_model, dtest)

rmse <- function(actual, predicted) sqrt(mean((actual - predicted)^2))

# Compute RMSE for all models
rmse_rf <- rmse(test$SalePrice, pred_rf)
rmse_lm <- rmse(test$SalePrice, pred_lm)
rmse_xgb <- rmse(test_label, pred_xgb)

# Create a comparison data frame
model_results <- data.frame(
  Model = c("Linear Regression", "Random Forest", "XGBoost"),
  RMSE = c(rmse_lm, rmse_rf, rmse_xgb)
)

# Display as a nicely formatted table
knitr::kable(model_results, caption = "Model Performance Comparison")
```

Table 2: Model Performance Comparison

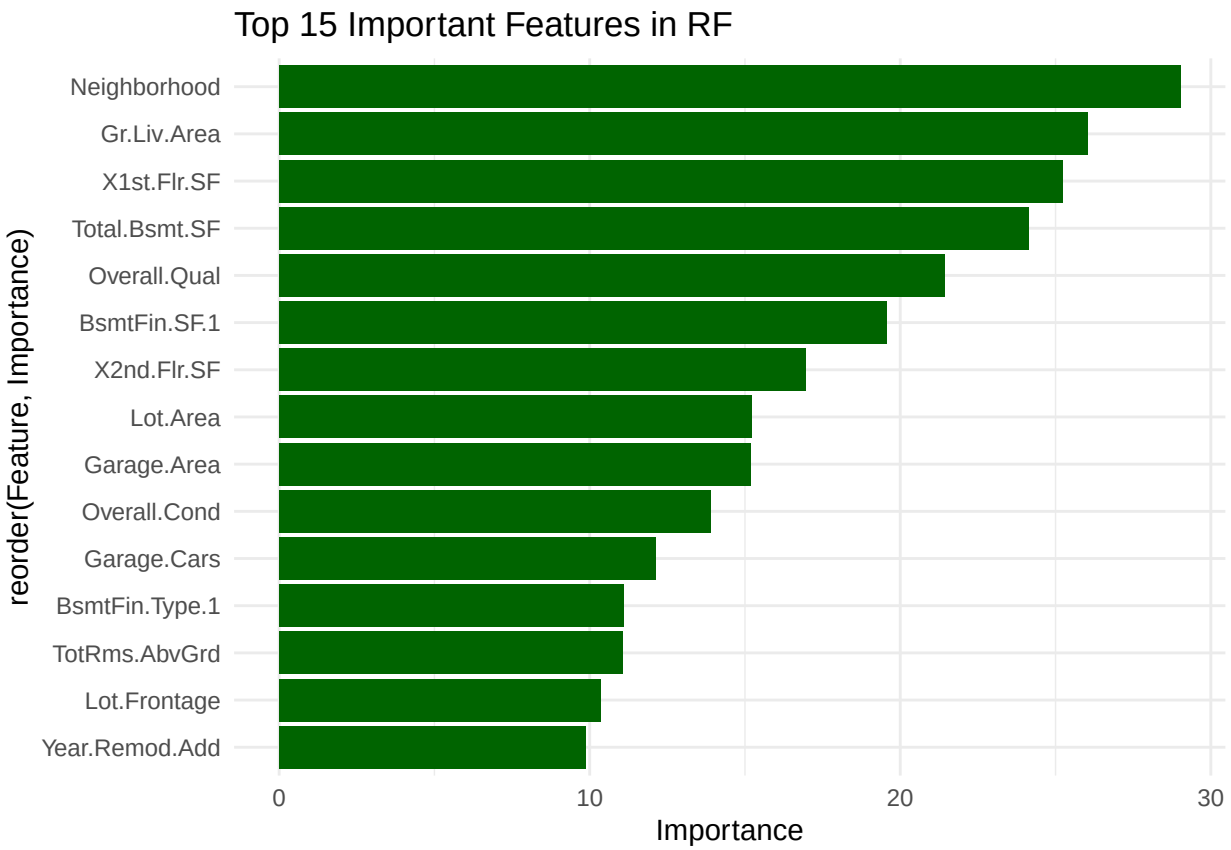
| Model | RMSE |
|---|---|
| Linear Regression | 0.1714574 |
| Random Forest | 0.1221804 |
| XGBoost | 0.1219641 |

I used RMSE (Root Mean Squared Error) to evaluate performance. Linear Regression RMSE value was 0.171, Random Forest RMSE value was 0.122, and XGBoost RMSE value was 0.122. XGBoost slightly edged out the others in performance, but both tree-based models clearly outperformed Linear Regression. The advanced models made much better predictions than basic ones, especially when things got complicated.

## 8: Feature Importance

```
# From Random Forest
importance <- importance(rf_model)
importance_df <- data.frame(Feature = rownames(importance), Importance = importance[, 1])
importance_df %>%
  top_n(15, Importance) %>%
  ggplot(aes(reorder(Feature, Importance), Importance)) +
  geom_bar(stat = "identity", fill = "darkgreen") +
```

```
  coord_flip() +
  labs(title = "Top 15 Important Features in RF") + theme_minimal()
```




The top 5 features using Random Forest's importance metric was Gr.Liv.Area (above-ground living area), Overall.Qual (quality), Garage.Area, Total.Bsmt.SF, and Neighborhood. This aligns with real-world logic: larger, better-built homes in desirable areas fetch higher prices.

## 9: Conclusion

This project showcased the value of machine learning in real estate analytics. After cleaning and exploring the Ames Housing dataset, I built predictive models that captured both accuracy and interpretability. The best-performing models - Random Forest and XGBoost - achieved low prediction error and confirmed intuitive drivers of price such as square footage, quality, and neighborhood.

While the linear regression model explained a high proportion of variance in housing prices (adjusted R-squared of approximately 0.93), several variables were excluded due to multicollinearity. This issue occurs when two or more predictors are highly correlated, making it difficult to isolate their individual effects. As a result, some predictors were dropped automatically by the model due to linear dependency (noted as "aliased" coefficients). A future enhancement could involve calculating the Variance Inflation Factor (VIF) to systematically detect and manage multicollinearity for improved interpretability.

Machine learning not only helps us to forecast home prices, but also confirms what really matters when buying or selling a house.