Robert Pavlik

D600 Statistical Data Mining

UGN1 Task 2: Logistic Regression Analysis

A. GitLab

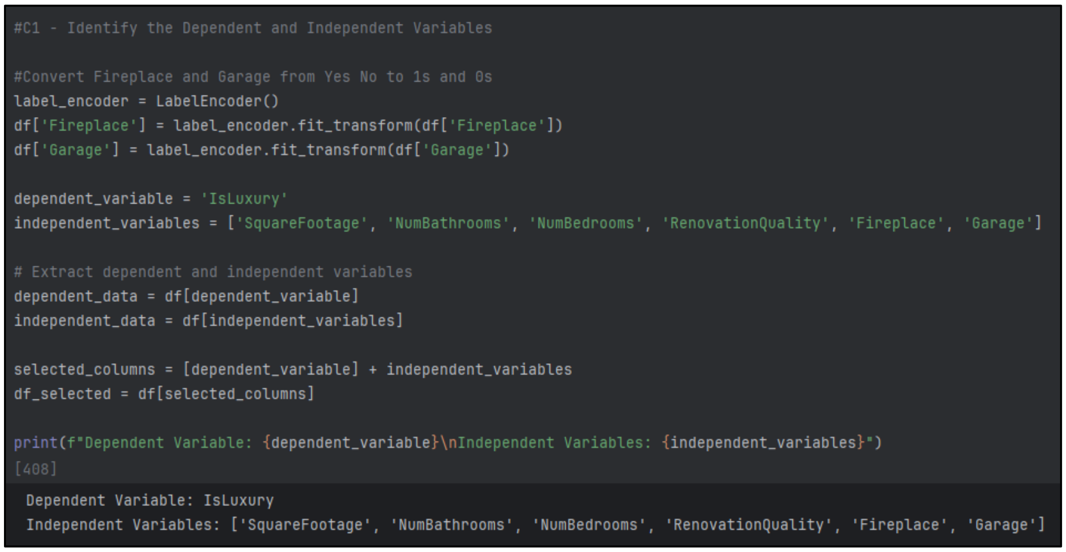
B. Describe the purpose of this Data Analysis

1. Research Question
   Can property features such as square footage, number of bathrooms, number of bedrooms, renovation quality, presence of a fireplace, and presence of a garage predict whether a house is classified as a luxury home?

2. Goal of Analysis
   This analysis aims to develop a logistic regression model to predict the likelihood of a house being classified as "Luxury" based on its physical features. This model will identify significant predictors among the selected features and provide actionable insights for real estate businesses to better understand what drives the luxury classification of properties

C. Summarize the Data Preparation Process for Logistic Regression Analysis

1. Identify Dependent and Independent Variables



```
#C1 - Identify the Dependent and Independent Variables

#Convert Fireplace and Garage from Yes No to 1s and 0s
label_encoder = LabelEncoder()
df['Fireplace'] = label_encoder.fit_transform(df['Fireplace'])
df['Garage'] = label_encoder.fit_transform(df['Garage'])

dependent_variable = 'IsLuxury'
independent_variables = ['SquareFootage', 'NumBathrooms', 'NumBedrooms', 'RenovationQuality', 'Fireplace', 'Garage']

# Extract dependent and independent variables
dependent_data = df[dependent_variable]
independent_data = df[independent_variables]

selected_columns = [dependent_variable] + independent_variables
df_selected = df[selected_columns]

print(f"Dependent Variable: {dependent_variable}\nIndependent Variables: {independent_variables}")
[408]
```

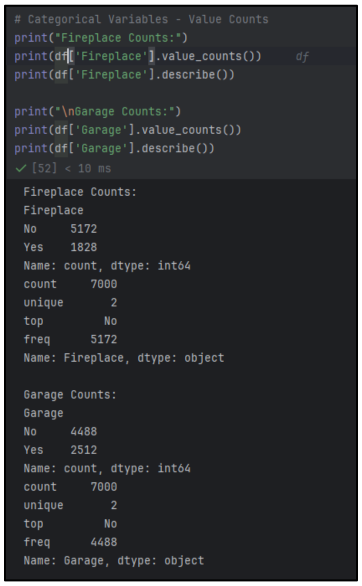
```
Dependent Variable: IsLuxury
Independent Variables: ['SquareFootage', 'NumBathrooms', 'NumBedrooms', 'RenovationQuality', 'Fireplace', 'Garage']
```

- Dependent Variable:
    - IsLuxury: A binary variable (0 = Non-Luxury, 1 = Luxury)

- Independent Variables:
    - SquareFootage: Continuous variable representing the size of the house in square feet. Larger square footage is typically associated with luxury homes.
    - NumBathrooms: Continuous variable representing the number of bathrooms in the house. Luxury homes often feature more bathrooms.
    - NumBedrooms: Continuous variable representing the number of bedrooms in the house. A higher number of bedrooms may indicate a luxury home.
    - RenovationQuality: Continuous variable representing the quality of renovations(scaled). Higher renovation quality is a characteristic of luxury homes.
    - Fireplace: Categorical variable indicating if the house has a fireplace (Yes / No). *Transformed to 1 for yes and 0 for No to make the variable continuous
    - Garage: Categorical variable indicating if the house has a garage (Yes / No). *Transformed to 1 for yes and 0 for No to make the variable continuous

The selection of these variables is justified based on their direct relevance to the physical attributes that contribute to a property's classification as luxury. These features provide a comprehensive assessment of size, amenities, and quality, which are key indicators in luxury real estate markets.

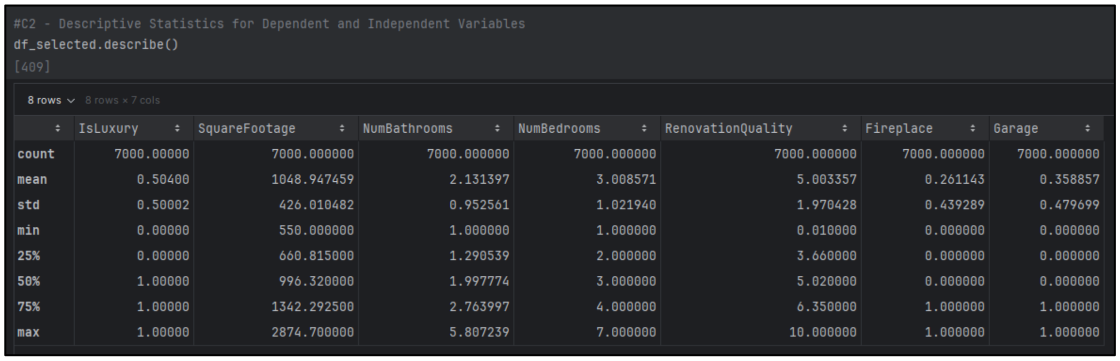
2. Descriptive Statistics of Dependent and Independent Variables



```
# Categorical Variables - Value Counts
print("Fireplace Counts:")
print(df['Fireplace'].value_counts())
print(df['Fireplace'].describe())

print("\nGarage Counts:")
print(df['Garage'].value_counts())
print(df['Garage'].describe())
```

```
[52] < 10 ms
```

```
Fireplace Counts:
Fireplace
No     5172
Yes    1828
Name: count, dtype: int64
count     7000
unique       2
top         No
freq      5172
Name: Fireplace, dtype: object

Garage Counts:
Garage
No     4488
Yes    2512
Name: count, dtype: int64
count     7000
unique       2
top         No
freq      4488
Name: Garage, dtype: object
```



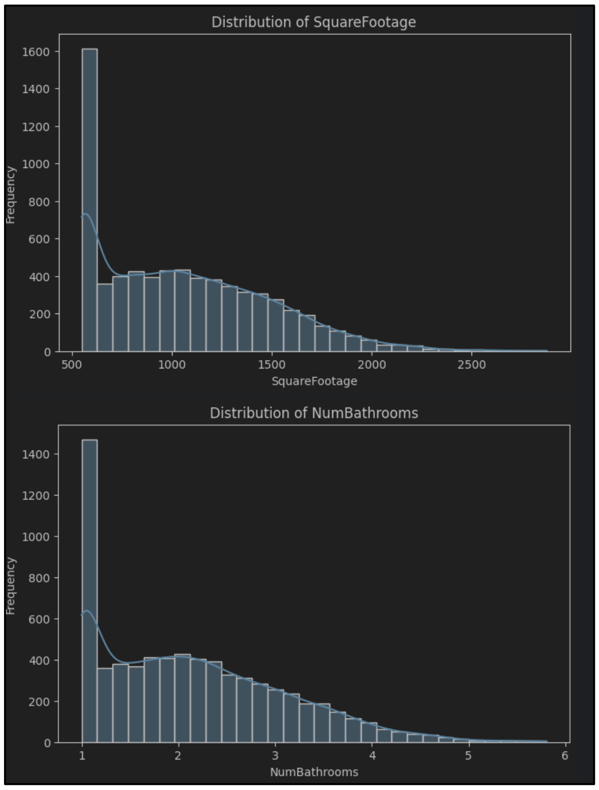
```
#C2 - Descriptive Statistics for Dependent and Independent Variables
df_selected.describe()
[409]
```

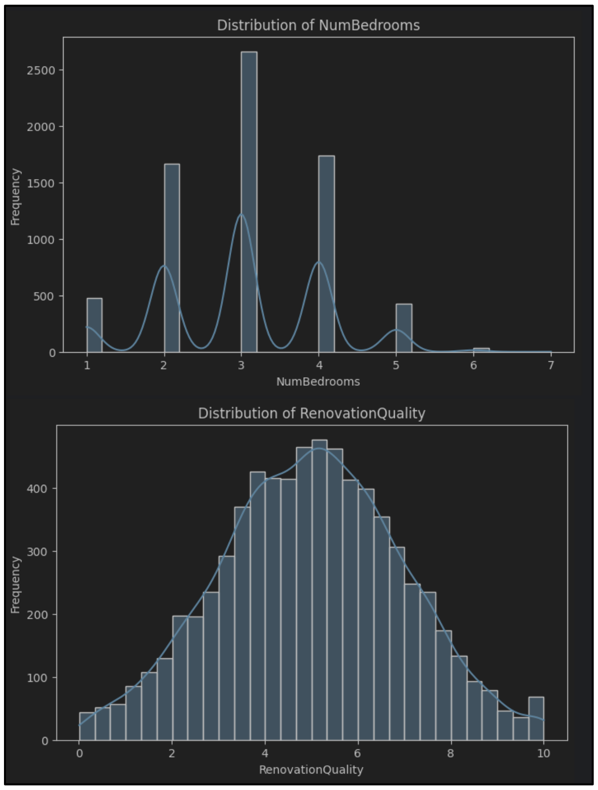
8 rows × 7 cols

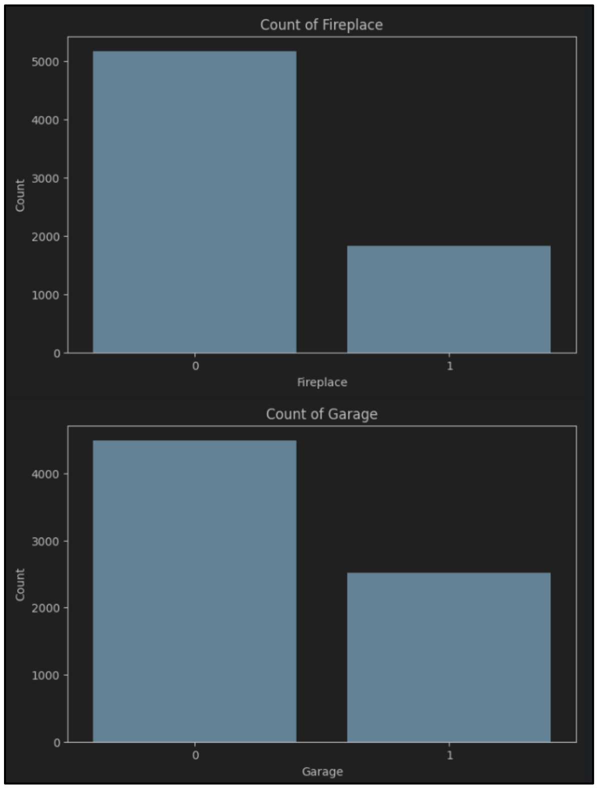
	IsLuxury	SquareFootage	NumBathrooms	NumBedrooms	RenovationQuality	Fireplace	Garage
count	7000.00000	7000.000000	7000.000000	7000.000000	7000.000000	7000.000000	7000.000000
mean	0.50400	1048.947459	2.131397	3.008571	5.003357	0.261143	0.358857
std	0.50002	426.010482	0.952561	1.021940	1.970428	0.439289	0.479699
min	0.00000	550.000000	1.000000	1.000000	0.010000	0.000000	0.000000
25%	0.00000	660.815000	1.290539	2.000000	3.660000	0.000000	0.000000
50%	1.00000	996.320000	1.997774	3.000000	5.020000	0.000000	0.000000
75%	1.00000	1342.292500	2.763997	4.000000	6.350000	1.000000	1.000000
max	1.00000	2874.700000	5.807239	7.000000	10.000000	1.000000	1.000000

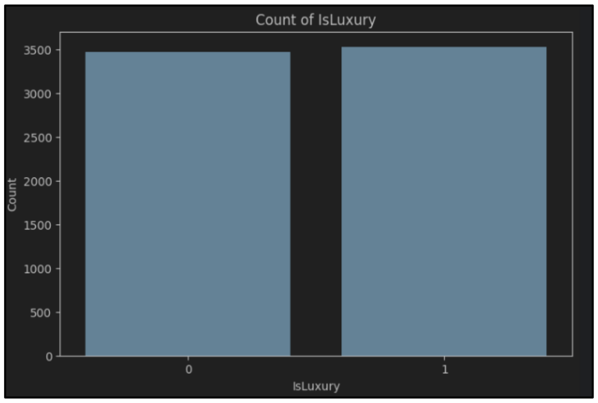
3. Univariate and Bivariate Statistical Visualization

- Univariate Visualization

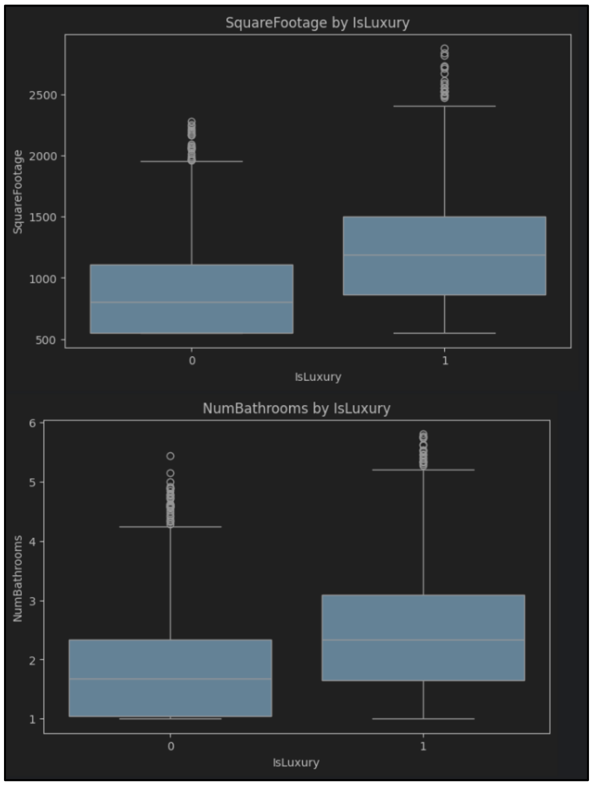


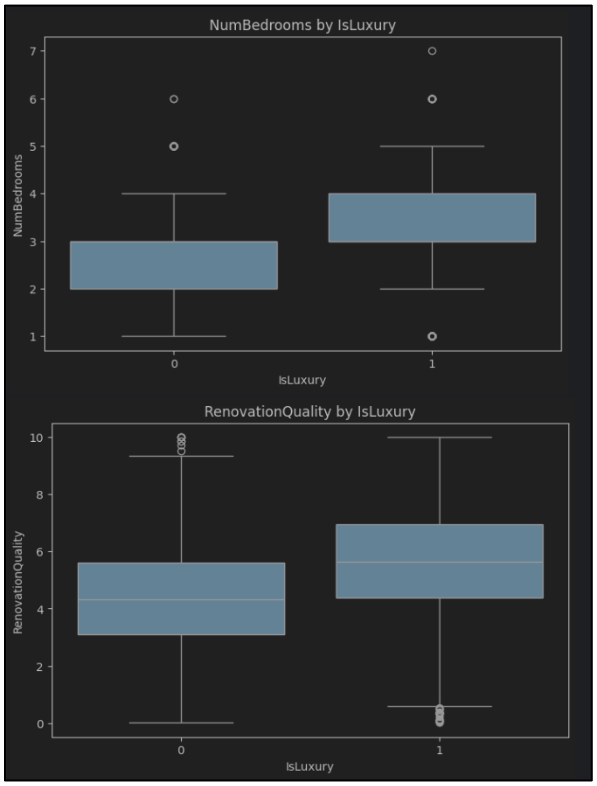


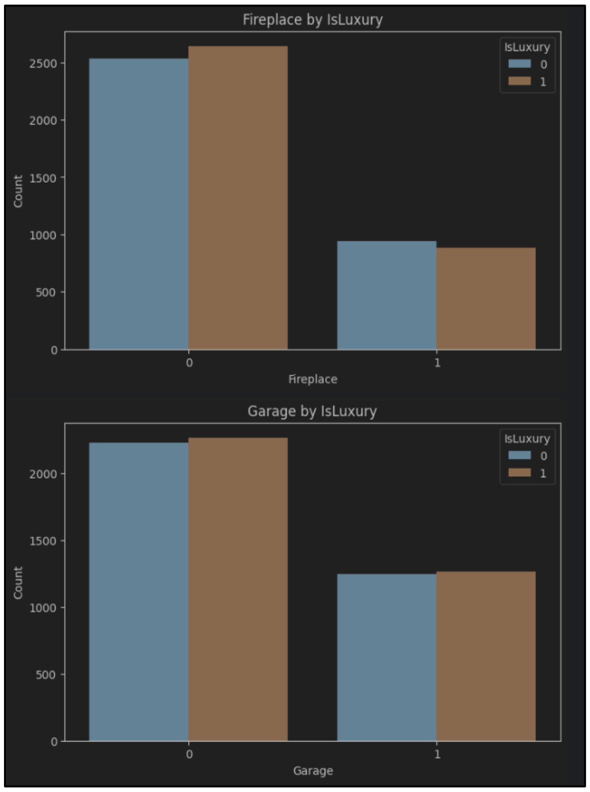




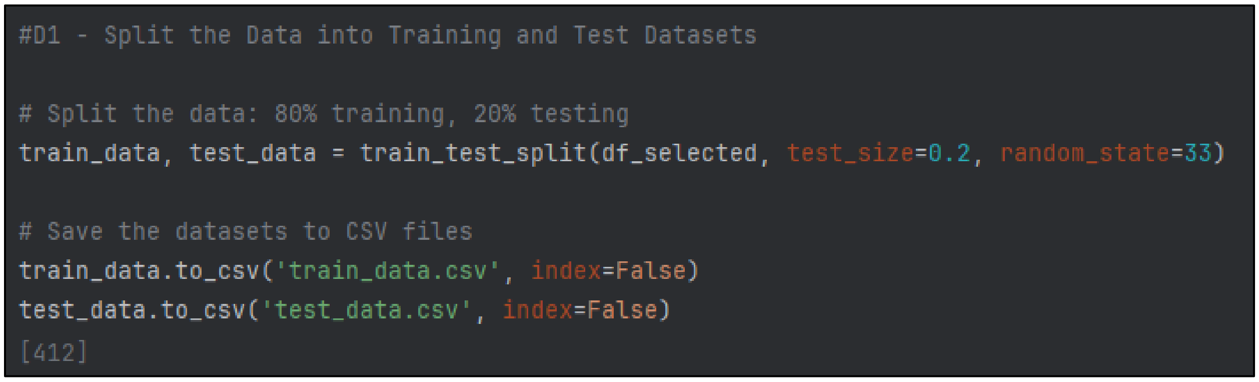
- Bivariate Visualization







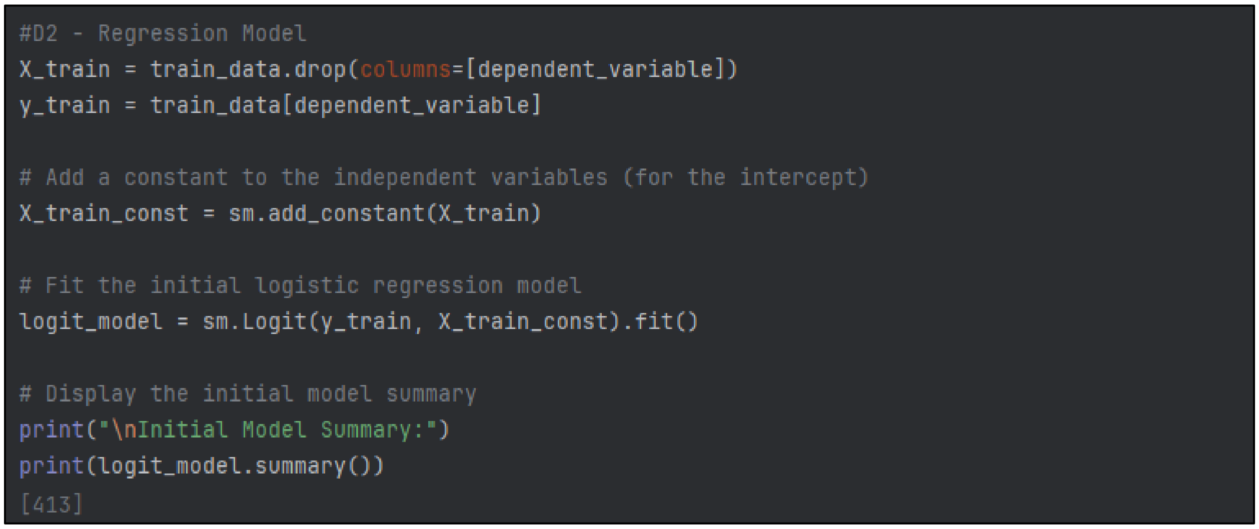
D. Perform the Data Analysis and Report on the Results



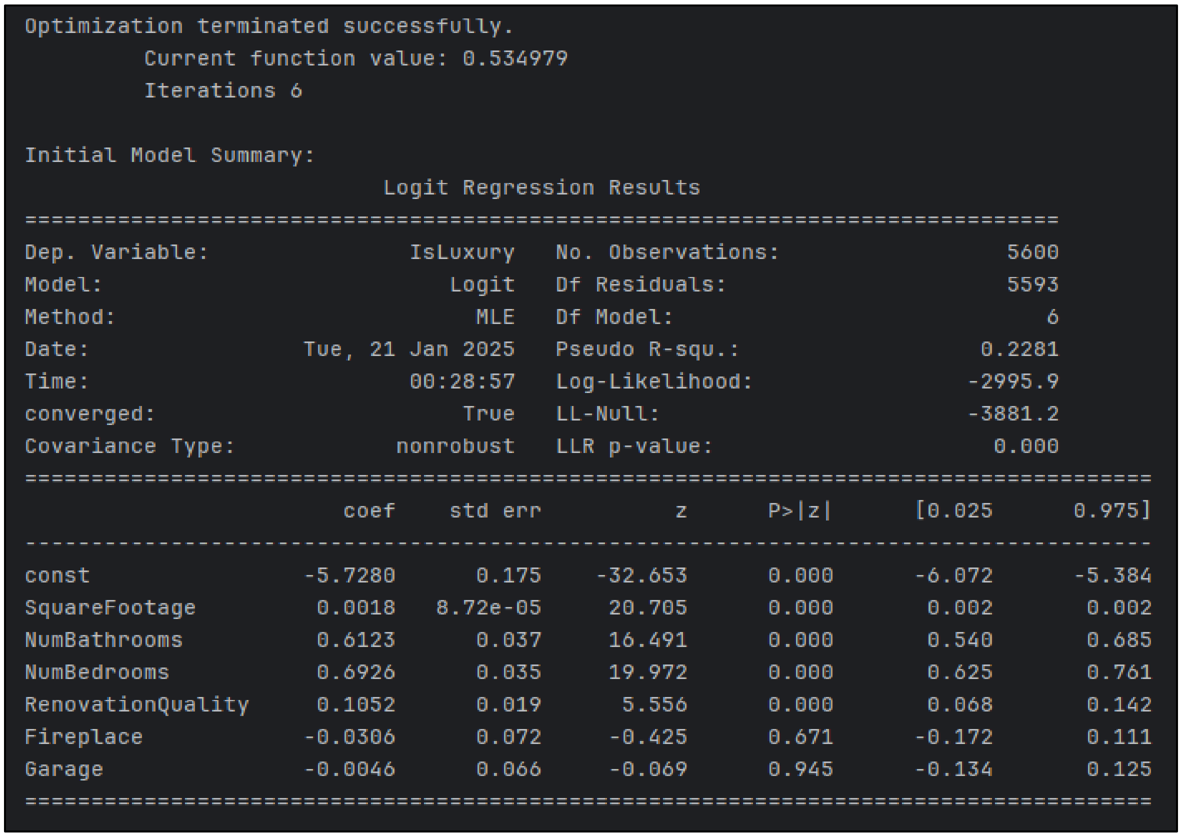
```
#D1 - Split the Data into Training and Test Datasets

# Split the data: 80% training, 20% testing
train_data, test_data = train_test_split(df_selected, test_size=0.2, random_state=33)

# Save the datasets to CSV files
train_data.to_csv('train_data.csv', index=False)
test_data.to_csv('test_data.csv', index=False)
[412]
```

1. Split Data into Two Datasets
   - The dataset was split into two sub-datasets
     - Training Set: 80% of the data, used to train the model
     - Test Set: 20% of the data, used to evaluate the model

2. Create and Perform a Regression Model / Optimize the Regression Model

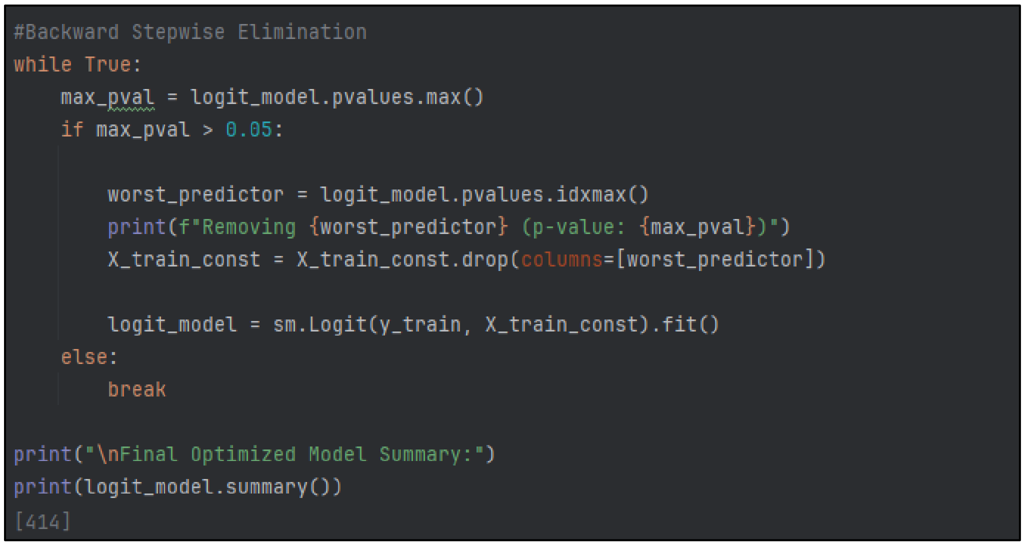


```
#D2 - Regression Model
X_train = train_data.drop(columns=[dependent_variable])
y_train = train_data[dependent_variable]

# Add a constant to the independent variables (for the intercept)
X_train_const = sm.add_constant(X_train)

# Fit the initial logistic regression model
logit_model = sm.Logit(y_train, X_train_const).fit()

# Display the initial model summary
print("\nInitial Model Summary:")
print(logit_model.summary())
[413]
```

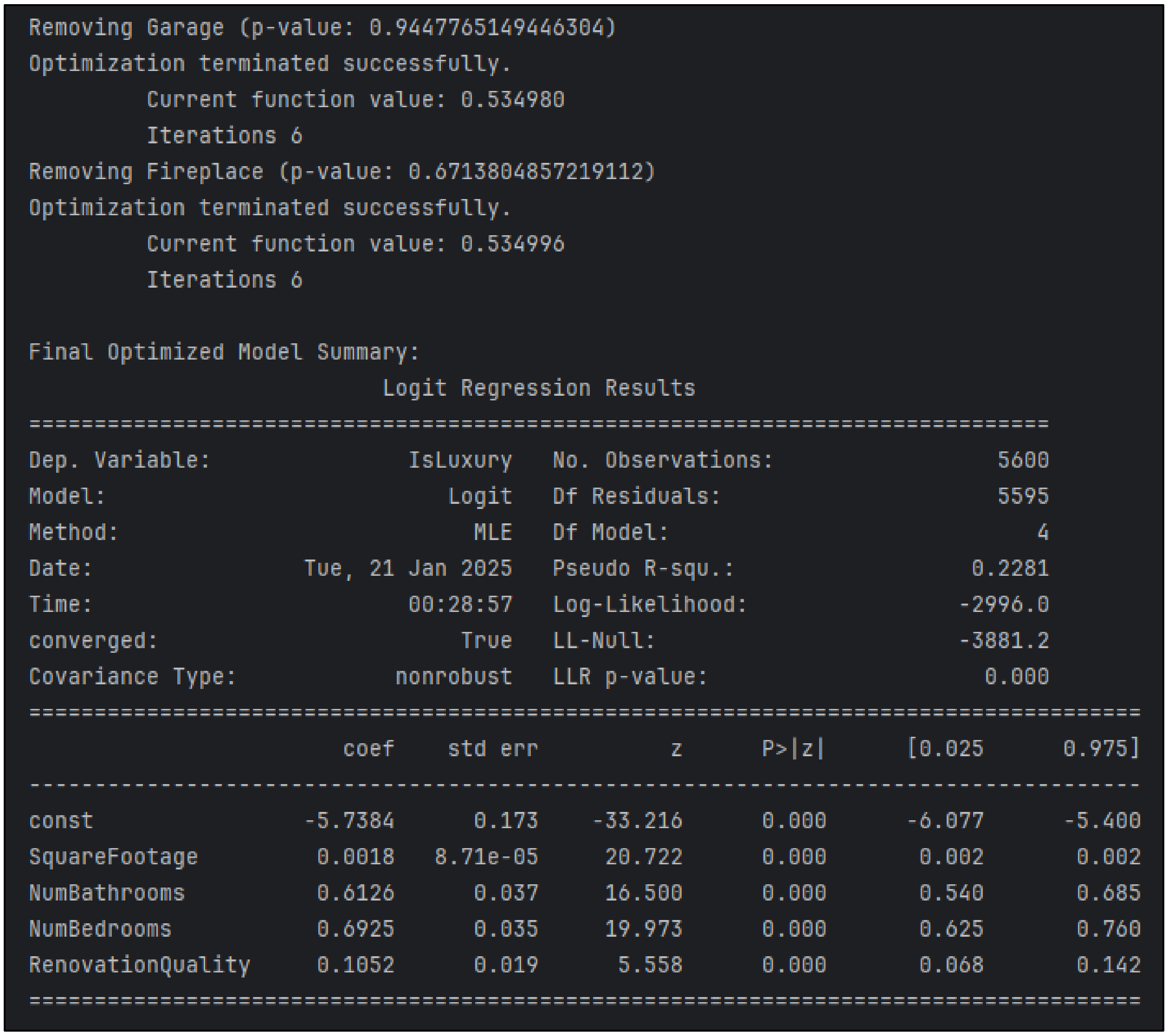


```
Optimization terminated successfully.
         Current function value: 0.534979
         Iterations 6

Initial Model Summary:
                           Logit Regression Results
==============================================================================
Dep. Variable:               IsLuxury   No. Observations:                 5600
Model:                          Logit   Df Residuals:                     5593
Method:                           MLE   Df Model:                            6
Date:                Tue, 21 Jan 2025   Pseudo R-squ.:                  0.2281
Time:                        00:28:57   Log-Likelihood:                -2995.9
converged:                       True   LL-Null:                       -3881.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
=====================================================================================
                        coef    std err          z      P>|z|      [0.025      0.975]
-------------------------------------------------------------------------------------
const                -5.7280      0.175    -32.653      0.000      -6.072      -5.384
SquareFootage         0.0018   8.72e-05     20.705      0.000       0.002       0.002
NumBathrooms          0.6123      0.037     16.491      0.000       0.540       0.685
NumBedrooms           0.6926      0.035     19.972      0.000       0.625       0.761
RenovationQuality     0.1052      0.019      5.556      0.000       0.068       0.142
Fireplace            -0.0306      0.072     -0.425      0.671      -0.172       0.111
Garage               -0.0046      0.066     -0.069      0.945      -0.134       0.125
=====================================================================================
```

- An initial logistic regression model was developed using the training dataset with all selected variables
- Backward Stepwise Elimination was used to optimize the model by removing variables with high p-values (>0.05)
- Initial Model Summary:
  - Pseudo R^2: 0.2281, indicating that the model explains about 22.81% of the variation in the dependent variable
  - Significant predictors: SquareFootage, NumBathrooms, NumBedrooms, and RenovationQuality
  - Non-Significant predictors: Fireplace (p=0.671) and Garage (p=0.945)

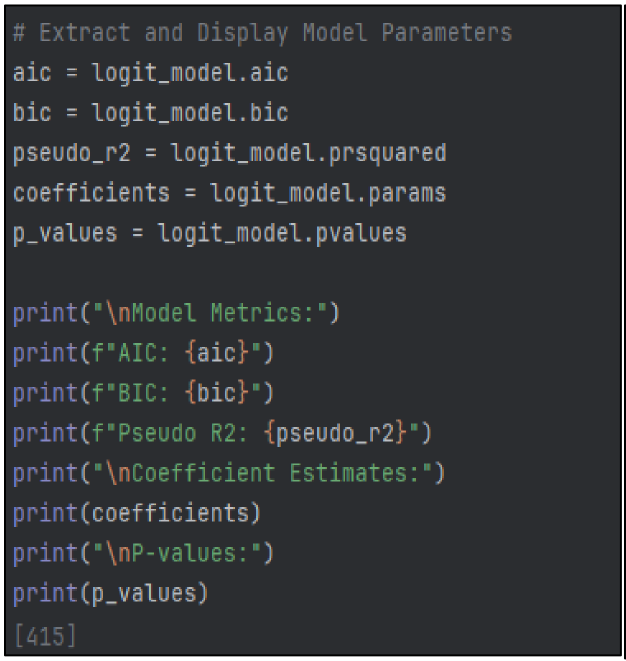


```python
#Backward Stepwise Elimination
while True:
    max_pval = logit_model.pvalues.max()
    if max_pval > 0.05:

        worst_predictor = logit_model.pvalues.idxmax()
        print(f"Removing {worst_predictor} (p-value: {max_pval})")
        X_train_const = X_train_const.drop(columns=[worst_predictor])

        logit_model = sm.Logit(y_train, X_train_const).fit()
    else:
        break

print("\nFinal Optimized Model Summary:")
print(logit_model.summary())
[414]
```

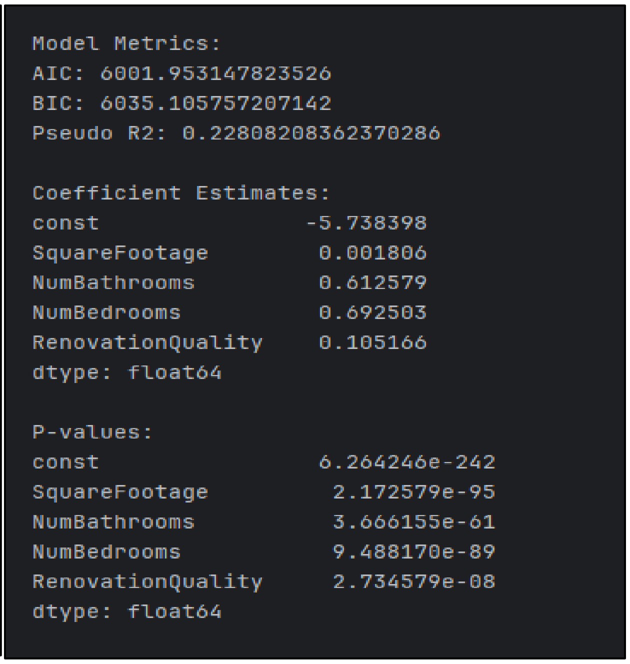


```
Removing Garage (p-value: 0.9447765149446304)
Optimization terminated successfully.
         Current function value: 0.534980
         Iterations 6
Removing Fireplace (p-value: 0.6713804857219112)
Optimization terminated successfully.
         Current function value: 0.534996
         Iterations 6

Final Optimized Model Summary:
                           Logit Regression Results
==============================================================================
Dep. Variable:               IsLuxury   No. Observations:                 5600
Model:                          Logit   Df Residuals:                     5595
Method:                           MLE   Df Model:                            4
Date:                Tue, 21 Jan 2025   Pseudo R-squ.:                  0.2281
Time:                        00:28:57   Log-Likelihood:                -2996.0
converged:                       True   LL-Null:                       -3881.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
=====================================================================================
                        coef    std err          z      P>|z|      [0.025      0.975]
-------------------------------------------------------------------------------------
const              -5.7384      0.173    -33.216      0.000      -6.077      -5.400
SquareFootage       0.0018   8.71e-05     20.722      0.000       0.002       0.002
NumBathrooms        0.6126      0.037     16.500      0.000       0.540       0.685
NumBedrooms         0.6925      0.035     19.973      0.000       0.625       0.760
RenovationQuality   0.1052      0.019      5.558      0.000       0.068       0.142
=====================================================================================
```

- Final Optimized Model Summary:
    - Variables retained: SquareFootage, NumBathrooms, NumBedrooms, and RenovationQuality
    - Variables removed: Fireplace and Garage
    - Pseudo R^2: 0.2281 (same as the initial model)
    - The coefficients for the retained variables are statistically significant, and their positive values suggest that increases in these features enhance the likelihood of being classified as luxury

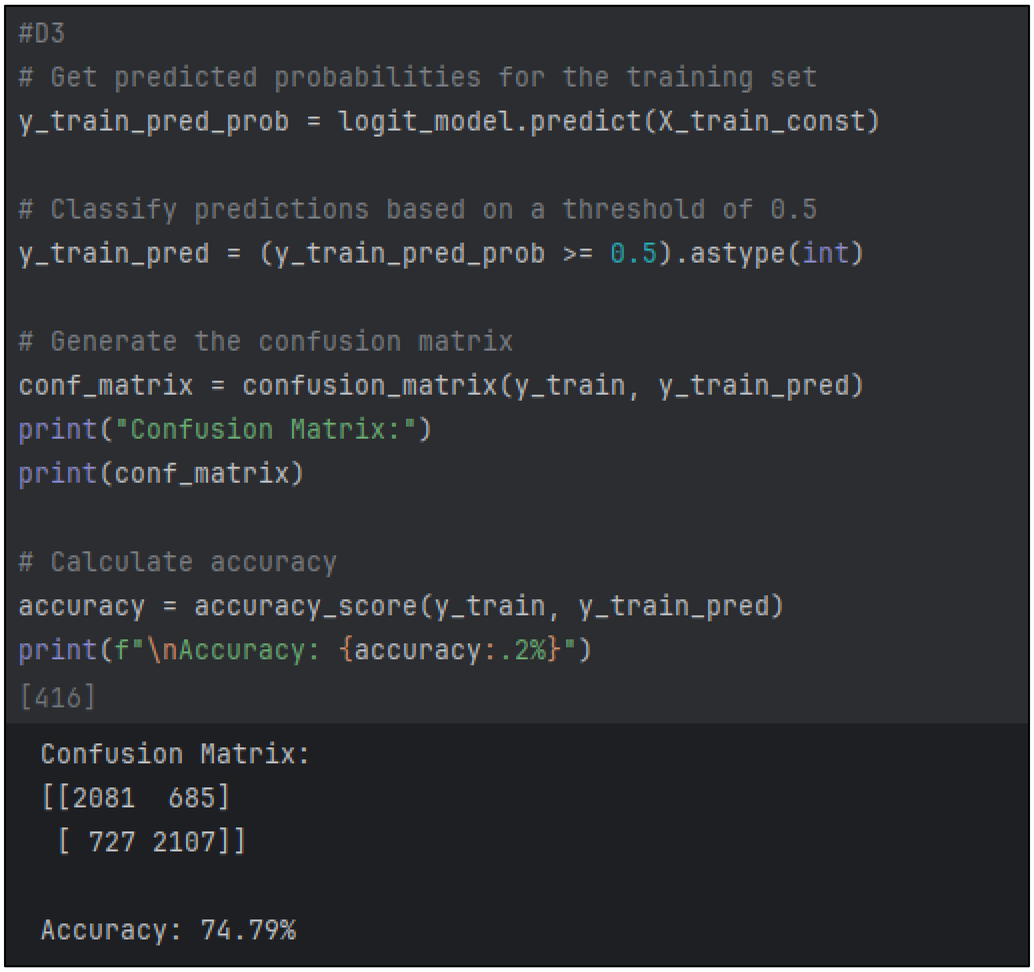


```
# Extract and Display Model Parameters
aic = logit_model.aic
bic = logit_model.bic
pseudo_r2 = logit_model.prsquared
coefficients = logit_model.params
p_values = logit_model.pvalues

print("\nModel Metrics:")
print(f"AIC: {aic}")
print(f"BIC: {bic}")
print(f"Pseudo R2: {pseudo_r2}")
print("\nCoefficient Estimates:")
print(coefficients)
print("\nP-values:")
print(p_values)
[415]
```



```
Model Metrics:
AIC: 6001.953147823526
BIC: 6035.105757207142
Pseudo R2: 0.22808208362370286

Coefficient Estimates:
const                  -5.738398
SquareFootage           0.001806
NumBathrooms            0.612579
NumBedrooms             0.692503
RenovationQuality       0.105166
dtype: float64

P-values:
const                   6.264246e-242
SquareFootage            2.172579e-95
NumBathrooms             3.666155e-61
NumBedrooms              9.488170e-89
RenovationQuality        2.734579e-08
dtype: float64
```

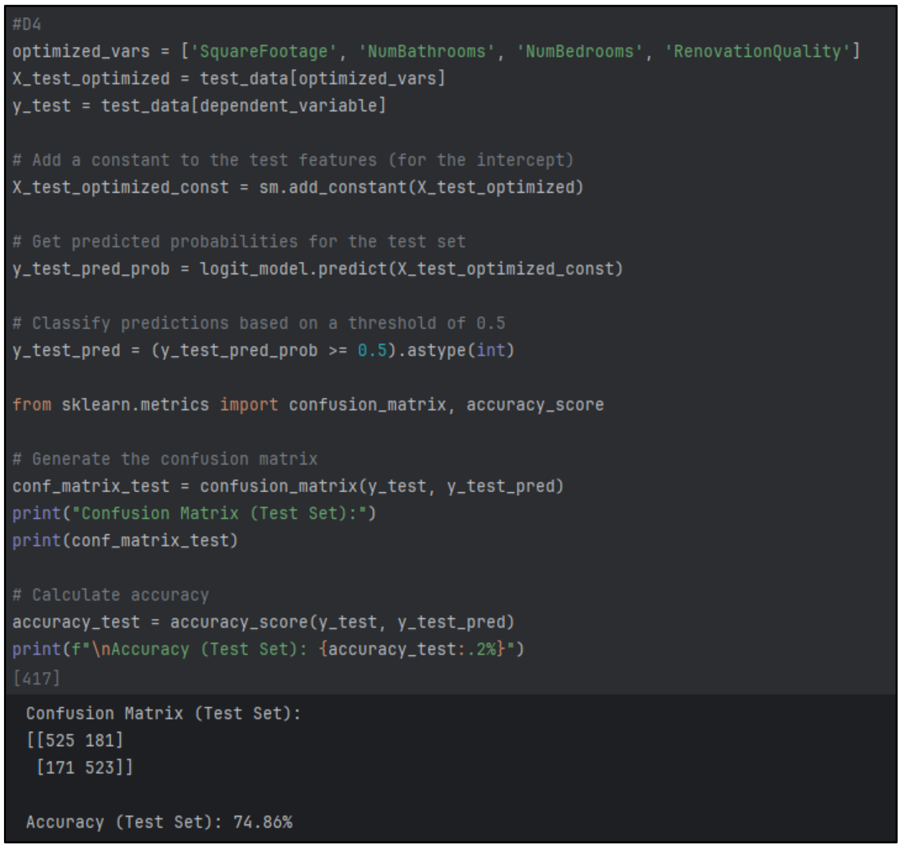
3. Confusion Matrix and Accuracy of Optimized Model
    - Confusion Matrix: [[2081 685] [727 2107]]
    - Accuracy: 74.79%



```
#D3
# Get predicted probabilities for the training set
y_train_pred_prob = logit_model.predict(X_train_const)

# Classify predictions based on a threshold of 0.5
y_train_pred = (y_train_pred_prob >= 0.5).astype(int)

# Generate the confusion matrix
conf_matrix = confusion_matrix(y_train, y_train_pred)
print("Confusion Matrix:")
print(conf_matrix)

# Calculate accuracy
accuracy = accuracy_score(y_train, y_train_pred)
print(f"\nAccuracy: {accuracy:.2%}")
[416]
```

```
Confusion Matrix:
[[2081  685]
 [ 727 2107]]

Accuracy: 74.79%
```

4. Run Prediction on Test Dataset using Optimized Regression Model
   - Confusion Matrix (Test Set): [[525 181] [171 523]]
   - Accuracy (Test Set): 74.86%



```
#D4
optimized_vars = ['SquareFootage', 'NumBathrooms', 'NumBedrooms', 'RenovationQuality']
X_test_optimized = test_data[optimized_vars]
y_test = test_data[dependent_variable]

# Add a constant to the test features (for the intercept)
X_test_optimized_const = sm.add_constant(X_test_optimized)

# Get predicted probabilities for the test set
y_test_pred_prob = logit_model.predict(X_test_optimized_const)

# Classify predictions based on a threshold of 0.5
y_test_pred = (y_test_pred_prob >= 0.5).astype(int)

from sklearn.metrics import confusion_matrix, accuracy_score

# Generate the confusion matrix
conf_matrix_test = confusion_matrix(y_test, y_test_pred)
print("Confusion Matrix (Test Set):")
print(conf_matrix_test)

# Calculate accuracy
accuracy_test = accuracy_score(y_test, y_test_pred)
print(f"\nAccuracy (Test Set): {accuracy_test:.2%}")
[417]
```

```
Confusion Matrix (Test Set):
[[525 181]
 [171 523]]

Accuracy (Test Set): 74.86%
```

E. Summarize Data Analysis

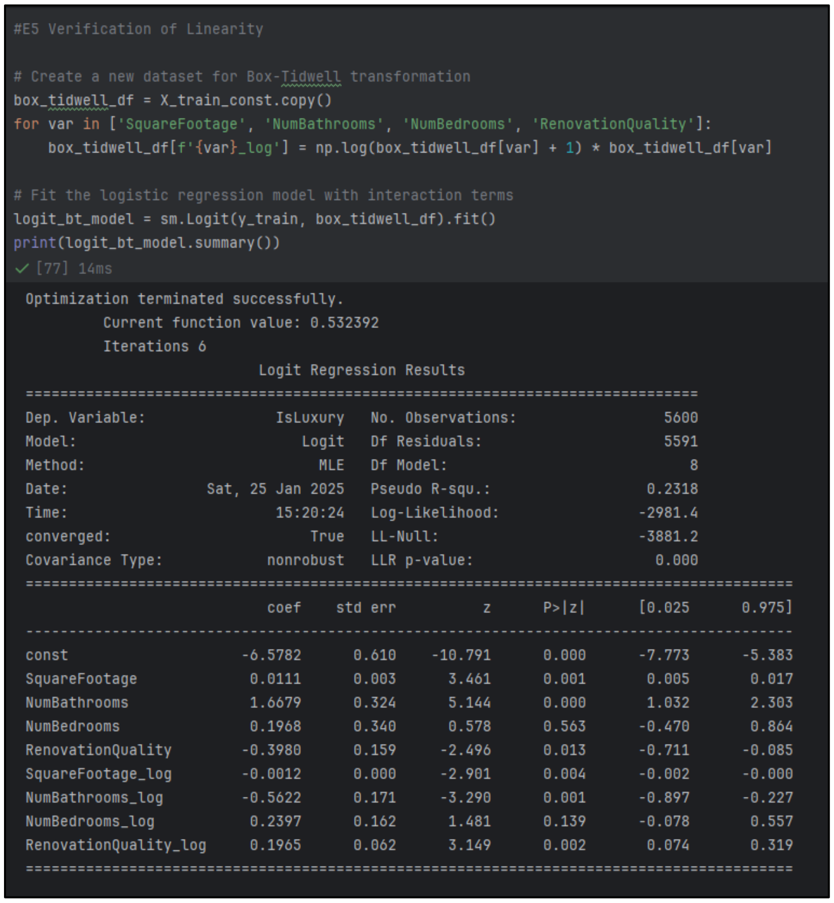
1. Packages and Libraries Used
   - Pandas: For data manipulation and preprocessing
   - Numpy: For numerical operations
   - Matplotlib & Seaborn: For visualizations
   - Sklearn: For data splitting, encoding, and metrics (confusion matrix, accuracy)
   - Statsmodels: For building and optimizing logistic regression models

2. Method used to Optimize
   - Backward Stepwise Elimination: Used to remove variables iteratively based on their p-values.

3. Justify Optimized Model
   - Backward stepwise elimination ensures that only statistically significant variables ($p<0.05$) are retained, improving model interpretability and reducing overfitting.

4. Summarize Logistic Regression (4 Assumptions)
   - Linearity of Logit: The relationship between independent and dependent variables and the log odds of dependent variables is linear
   - Independence of Observation: Observations are independent of each other
   - No Multicollinearity: Independent variables are not highly correlated
   - Sufficient Sample Size: The dataset is large enough to provide reliable estimates.

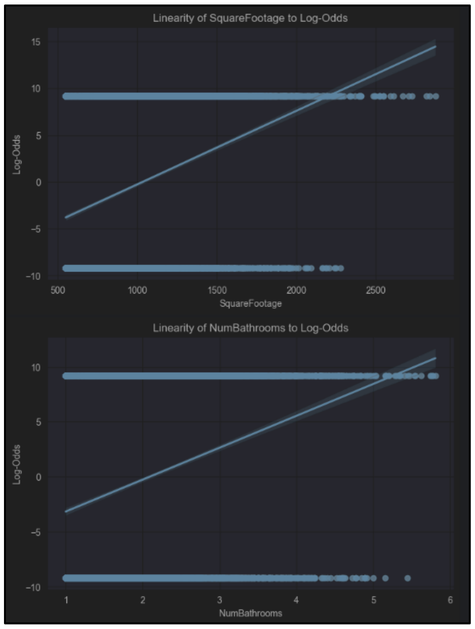
5. Evidence of Assumptions Verification
   - The logit regression results incorporating the Box-Tidwell interaction terms provide insights into the linearity assumption between the independent variables and the log odds of the dependent variable.

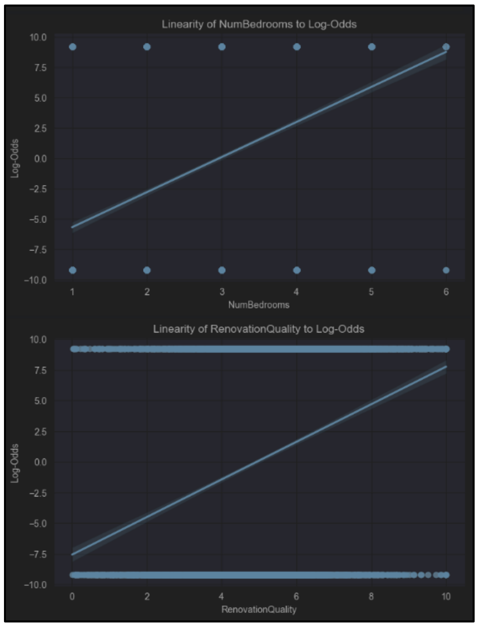


```
#E5 Verification of Linearity

# Create a new dataset for Box-Tidwell transformation
box_tidwell_df = X_train_const.copy()
for var in ['SquareFootage', 'NumBathrooms', 'NumBedrooms', 'RenovationQuality']:
    box_tidwell_df[f'{var}_log'] = np.log(box_tidwell_df[var] + 1) * box_tidwell_df[var]

# Fit the logistic regression model with interaction terms
logit_bt_model = sm.Logit(y_train, box_tidwell_df).fit()
print(logit_bt_model.summary())
✓ [77] 14ms
```

```
Optimization terminated successfully.
         Current function value: 0.532392
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:               IsLuxury   No. Observations:                 5600
Model:                          Logit   Df Residuals:                     5591
Method:                           MLE   Df Model:                            8
Date:                Sat, 25 Jan 2025   Pseudo R-squ.:                  0.2318
Time:                        15:20:24   Log-Likelihood:                -2981.4
converged:                       True   LL-Null:                       -3881.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
=========================================================================================
                            coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                    -6.5782      0.610    -10.791      0.000      -7.773      -5.383
SquareFootage             0.0111      0.003      3.461      0.001       0.005       0.017
NumBathrooms              1.6679      0.324      5.144      0.000       1.032       2.303
NumBedrooms               0.1968      0.340      0.578      0.563      -0.470       0.864
RenovationQuality        -0.3980      0.159     -2.496      0.013      -0.711      -0.085
SquareFootage_log        -0.0012      0.000     -2.901      0.004      -0.002      -0.000
NumBathrooms_log         -0.5622      0.171     -3.290      0.001      -0.897      -0.227
NumBedrooms_log           0.2397      0.162      1.481      0.139      -0.078       0.557
RenovationQuality_log     0.1965      0.062      3.149      0.002       0.074       0.319
=========================================================================================
```

- Pseudo R^2: 0.2318 indicates that the model explains 23.18% of the variability in the dependent variable. This is slightly higher than the initial model suggesting that incorporating Box-Tidwell terms marginally improves the fit.
- The significant coefficients for SquareFootage, NumBathrooms, and RenovationQuality confirm their strong relationship with the dependent variable. The corresponding interaction terms (*_log) adjust for non-linear effects, improving the model fit.
- The inclusion of the Box-Tidwell terms validates the assumptions of linearity for most variables by identifying and adjusting non-linear trends. This step ensures the reliability of the logistic regression results while retaining meaningful interpretations.





- Clear Linear Relationships: For SquareFootage, NumBathrooms, and RenovationQuality, the log odds show a consistent linear trend, confirming the assumption of linearity
- Less Impactful Trend: for NumBedrooms, the relationship appears weaker or less clear it may have a limited impact on the log odds, though it still contributes to the overall model

6. Discuss Model Metrics

   The final regression equation is:

   Log(odds) = -5.7384 + 0.0018(SquareFootage) + 0.6126(NumBathrooms) + 0.6925(NumBedrooms) + 0.1052(RenovationQuality)

   - Interpretation of Coefficients:
     - SquareFootage: For each additional square foot, the log odds of being luxury increase by 0.0018

        - NumBathrooms: Each additional bathroom increases the log odds of being luxury by 0.6126
        - NumBedrooms: Each additional bedroom increases the log odds of being luxury by 0.6925
        - RenovationQuality: Higher renovation quality increases the log odds of being luxury by 0.1052 per unit

7. Results of Prediction Analysis
    - Test Set Accuracy:
        - The test set accuracy of 74.86% indicated that the model correctly classified about 3 out of 4 properties as luxury or non-luxury.
    - Training vs Test Accuracy:
        - The similarity between training accuracy (74.79%) and test accuracy (74.86%) suggests the model generalizes well without overfitting
    - Confusion Matrices:
        - Training [[2081 685], [727 2107]]
            - Precision (Luxury): TP / (TP+FP) = 2107 / (2107 + 685) = 75.45%
            - Recall (Luxury): TP / (TP + FN) = 2107 / (2107 + 727) = 74.36%
        - Test [[525 181], [171 523]]
            - Precision (Luxury): 523 / (523 + 181) = 74.29%
            - Recall (Luxury): 523 / (523 + 171) = 75.35%
        - Formulas:
          Precision = True Positives (TP) / True Positives (TP) + False Negatives (FN)
          Recall = True Positives (TP) / True Positives (TP) + False Negatives (FN)
            - A higher precision indicates fewer false positives, meaning that when the model predicts a property as Luxury, it is more likely correct. A higher recall means the model captures most of the luxury properties, with fewer false negatives.

    - The consistent precision and recall between training and test sets indicate that the model is stable and generalizes well to unseen data.
    - These metrics balance the trade-off between false positives and false negatives, making the model reliable for real-world applications.

8. Results and Implications of Prediction Analysis
    - The model identifies key factors driving luxury classification, such as size, number of bathrooms and bedrooms, and renovation quality.

- Fireplace and garage, while common in luxury homes, were not significant predictors when accounting for other features

9. Recommended Course of Action
   - Focus marketing and pricing strategies on properties with large square footage, multiple bathrooms and bedrooms, and high-quality renovations.
   - Use the model to predict luxury status for new properties to assist with segmentation and target marketing
   - Consider collecting additional features (such as neighborhood attributes) to further refine predictions.