Nicole Gallo
D600 – Statistical Data Mining
July 24, 2025
UGN1 Task 2: Logistic Regression Analysis

A. Create your subgroup and project in GitLab using the provided web link by doing the following:

- Clone the project to the IDE.
- Commit with a message and push when you complete each requirement listed in parts C2 through D4.
- Submit a copy of the GitLab repository URL in the "Comments to Evaluator" section when you submit this assessment.
- Submit a copy of the repository branch history retrieved from your repository, which must include the commit messages and dates.

B. Describe the purpose of this data analysis by doing the following:

1. Propose one research question that is relevant to a real-world organizational situation captured in the provided dataset that you will answer using logistic regression in the initial model.

   Can we predict whether a home is considered "luxury" based on square footage, bedroom count, school rating, and renovation quality?

2. Define one goal of the data analysis. Ensure that your goal is reasonable within the scope of the scenario and is represented in the available data.

   The goal of this analysis is to help real estate agents identify which characteristics of a home (square footage, number of bedrooms, school rating, renovation quality) are most likely to influence whether the home is categorized as “luxury”.
   This type of insight can inform marketing, design, and pricing strategies in the upscale housing market.

C. Summarize the data preparation process for logistic regression analysis by doing the following:

1. Identify the dependent and all independent variables that are required to answer the research question and justify your selection of variables.

   Dependent Variable
   `IsLuxury` (*Binary – 0 = not luxury, 1 = luxury*): We will use this as the dependent variable to predict, yes or no, whether a home is considered “luxury”.

   Independent Variables
   `SquareFootage`: Larger homes tend to be more luxurious
   `NumBedrooms`: More bedrooms contribute to the larger size, higher value perception
   `SchoolRating`: Higher-rated schools increase neighborhood value

`RenovationQuality`: Indicator of luxury finish or upgraded features

2. Describe the dependent variable and all independent variables from part C1 using descriptive statistics (counts, means, modes, ranges, min/max), including a screenshot of the descriptive statistics output for each of these variables.

	IsLuxury	**SquareFootage**	**NumBedrooms**	**SchoolRating**	**RenovationQuality**
count	7000.00000	7000.000000	7000.000000	7000.000000	7000.000000
mean	0.50400	1048.947459	3.008571	6.942923	5.003357
std	0.50002	426.010482	1.021940	1.888148	1.970428
min	0.00000	550.000000	1.000000	0.220000	0.010000
25%	0.00000	660.815000	2.000000	5.650000	3.660000
50%	1.00000	996.320000	3.000000	7.010000	5.020000
75%	1.00000	1342.292500	4.000000	8.360000	6.350000
max	1.00000	2874.700000	7.000000	10.000000	10.000000

```python
import pandas as pd

# Load dataset
df = pd.read_csv("D600_Task2_Dataset1_Housing_Information.csv")
```

```python
# Subset to selected variables
selected_vars = ['IsLuxury', 'SquareFootage', 'NumBedrooms', 'SchoolRating', 'RenovationQuality']
df_subset = df[selected_vars]
```

✓ 0.0s

```python
# Get descriptive statistics
descriptive_stats = df_subset.describe(include='all')

# Display the full output
print("Descriptive Statistics for Selected Variables:\n")
display(descriptive_stats)
```

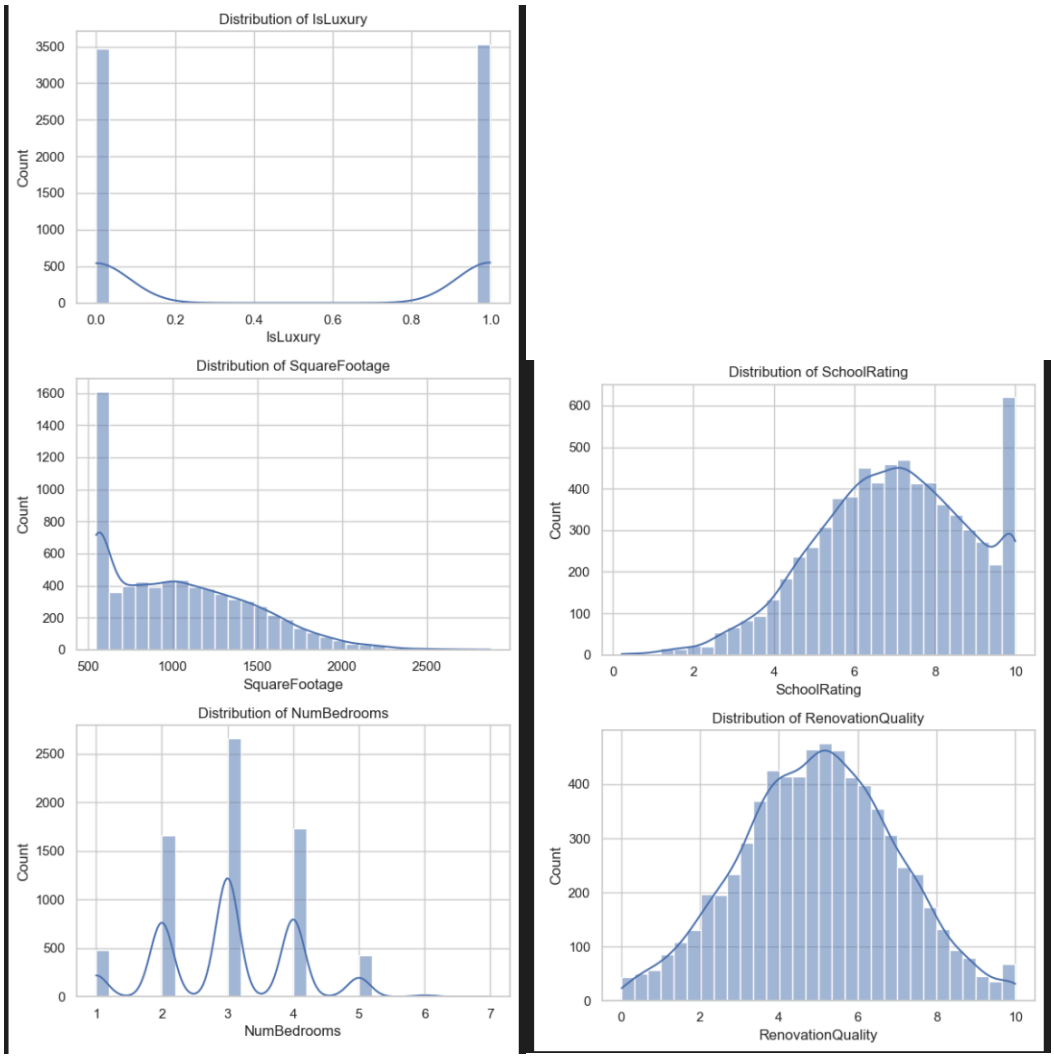
✓ 0.0s

```
Descriptive Statistics for Selected Variables:
```

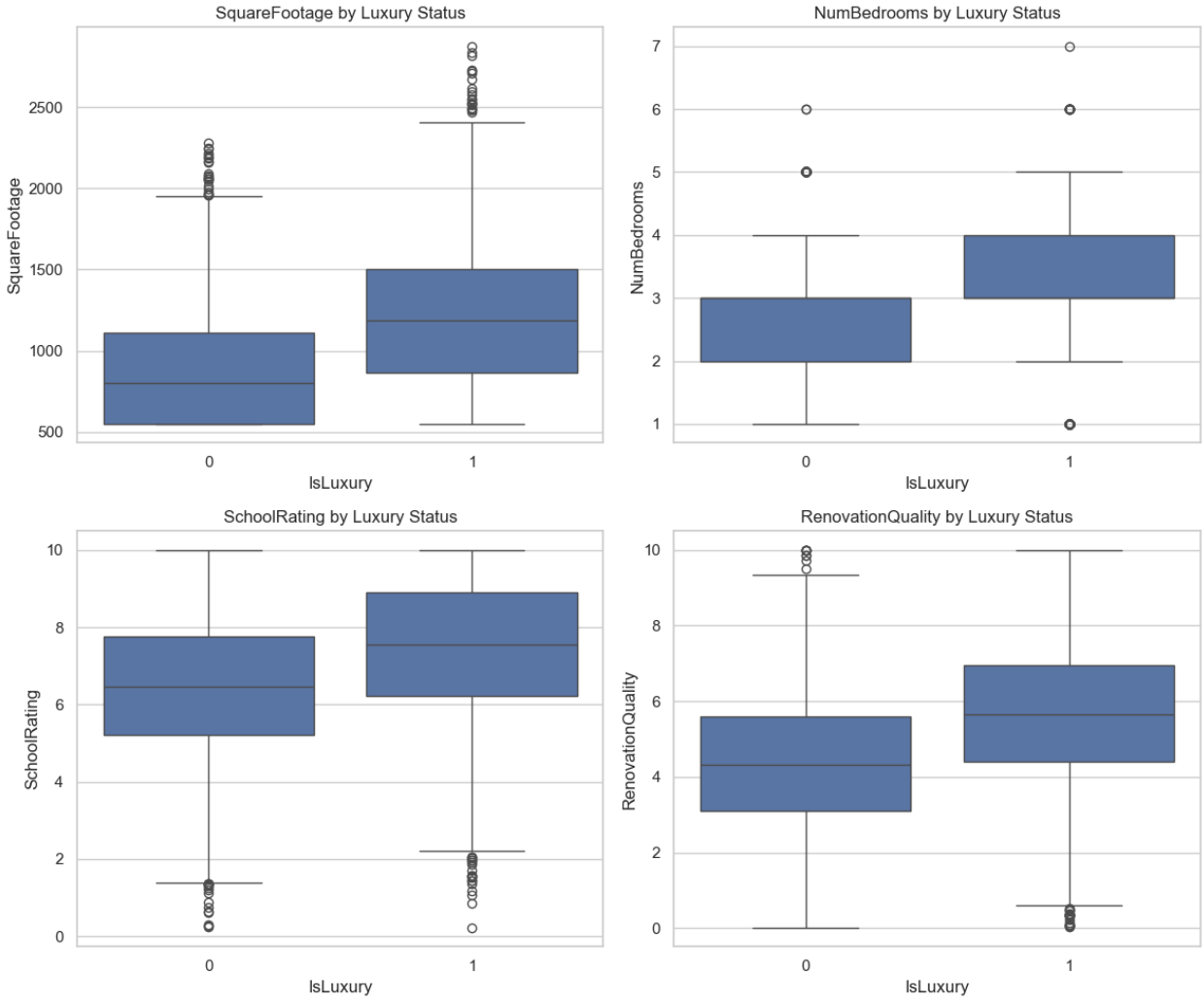
	IsLuxury	SquareFootage	NumBedrooms	SchoolRating	RenovationQuality
count	7000.00000	7000.000000	7000.000000	7000.000000	7000.000000
mean	0.50400	1048.947459	3.008571	6.942923	5.003357
std	0.50002	426.010482	1.021940	1.888148	1.970428
min	0.00000	550.000000	1.000000	0.220000	0.010000
25%	0.00000	660.815000	2.000000	5.650000	3.660000
50%	1.00000	996.320000	3.000000	7.010000	5.020000
75%	1.00000	1342.292500	4.000000	8.360000	6.350000
max	1.00000	2874.700000	7.000000	10.000000	10.000000

3. Generate univariate and bivariate visualizations of the distributions of the dependent and independent variables from part C1, including the dependent variable in the bivariate visualizations.

   Univariate (histograms) –



Bivariate (boxplots) –



D. Perform the data analysis and report on the results by doing the following:

1. Split the data into two datasets, with a larger percentage assigned to the training dataset and a smaller percentage assigned to the test dataset. Provide the file(s).

   D600_Task2_TrainSet.csv *(attached in submission)*

   D600_Task2_TestSet.csv *(attached in submission)*

2. Use the training dataset to create and perform a regression model using regression as a statistical method. Optimize the regression model using a process of your selection, including but not limited to, forward stepwise selection, backward stepwise elimination, and recursive selection. Provide a screenshot of the summary of the optimized model or the following extracted model parameters:
   - AIC: 6256.286118414961
   - BIC: 6289.438727798577
   - pseudo R2: 0.1954

- coefficient estimates:

	coef
const	-4.7812
SquareFootage	0.0018
NumBedrooms	0.6504
SchoolRating	0.0069
RenovationQuality	0.1910

- p-value of each independent variable

	P>|z|
const	0.000
SquareFootage	0.000
NumBedrooms	0.000
SchoolRating	0.725
RenovationQuality	0.000

```python
# Task 2 - D2

import statsmodels.api as sm

# Add constant to X_train
X_train_sm = sm.add_constant(X_train)

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

# Print model summary
print(result.summary())

print("AIC:", result.aic)
print("BIC:", result.bic)
```

✓ 0.0s

```
Optimization terminated successfully.
         Current function value: 0.557704
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:              IsLuxury   No. Observations:                 5600
Model:                         Logit   Df Residuals:                     5595
Method:                          MLE   Df Model:                            4
Date:                Fri, 25 Jul 2025   Pseudo R-squ.:                 0.1954
Time:                        10:21:52   Log-Likelihood:                -3123.1
converged:                       True   LL-Null:                       -3881.5
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================
                     coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const             -4.7812      0.163    -29.399      0.000      -5.100      -4.462
SquareFootage      0.0018   8.48e-05     20.929      0.000       0.002       0.002
NumBedrooms        0.6504      0.035     18.504      0.000       0.582       0.719
SchoolRating       0.0069      0.020      0.352      0.725      -0.032       0.045
RenovationQuality  0.1910      0.019      9.982      0.000       0.154       0.229
==============================================================================
AIC: 6256.286118414961
BIC: 6289.438727798577
```

3. Give the confusion matrix and accuracy of the optimized model used on the training set.

```
Confusion Matrix:
 [[2034  744]
 [ 778 2044]]

Training Set Accuracy: 0.7282
```

```python
# Task 2 - D3
from sklearn.metrics import confusion_matrix, accuracy_score

# Predict probabilities on training set
y_train_pred_prob = result.predict(X_train_sm)

# Convert probabilities to binary class predictions (threshold = 0.5)
y_train_pred = (y_train_pred_prob >= 0.5).astype(int)

# Generate confusion matrix and accuracy
conf_matrix = confusion_matrix(y_train, y_train_pred)
accuracy = accuracy_score(y_train, y_train_pred)

print("Confusion Matrix:\n", conf_matrix)
print("\nTraining Set Accuracy: {:.4f}".format(accuracy))
```

✓ 0.0s

```
Confusion Matrix:
 [[2034  744]
 [ 778 2044]]

Training Set Accuracy: 0.7282
```

4. Run the prediction on the test dataset using the optimized regression model from part D2 to evaluate the performance of the prediction model on the test data based on the confusion matrix and accuracy. Provide a screenshot of the results.
Note: The prediction run on the test dataset must use only the variables identified in the optimized regression model in part D2.

```
Confusion Matrix (Test Set):
 [[492 202]
 [211 495]]

Test Set Accuracy: 0.7050
```

```python
# Task 2 – D4
from sklearn.metrics import confusion_matrix, accuracy_score

# Add intercept to test set
X_test_sm = sm.add_constant(X_test)

# Predict probabilities on test set
y_test_pred_prob = result.predict(X_test_sm)

# Convert to binary predictions
y_test_pred = (y_test_pred_prob >= 0.5).astype(int)

# Confusion matrix and accuracy
conf_matrix_test = confusion_matrix(y_test, y_test_pred)
accuracy_test = accuracy_score(y_test, y_test_pred)

# Print results
print("Confusion Matrix (Test Set):\n", conf_matrix_test)
print("\nTest Set Accuracy: {:.4f}".format(accuracy_test))
```

✓ 0.0s

```
Confusion Matrix (Test Set):
 [[492 202]
 [211 495]]

Test Set Accuracy: 0.7050
```

E. Summarize your data analysis by doing the following:

1. List the packages or libraries you have chosen for Python or R and justify how each item on the list supports the analysis.

- `import pandas as pd` – used to load, clean, and explore the dataset as a dataframe (df), making it easier to manipulate variables and structure the data for modeling
- `import numpy as np` – used for numerical operations and array-based calculations, particularly behind the scenes in data manipulation and modeling
- `import matplotlib.pyplot as plt`
- `import seaborn as sns`
    - `matplotlib` and `seaborn` libraries were used to create univariate histograms and bivariate boxplots to visualize the distribution of variables and their relationships with the target (`IsLuxury`)
- `import statsmodels.api as sm` – used to build the logistic regression model, extract statistical outputs such as AIC, BIC, p-values, and pseudo R2
- `from sklearn.model_selection import train_test_split` – used to split the dataset into training and testing sets to assess how well the model generalizes
- `from sklearn.metrics import confusion_matrix, accuracy_score` – used to evaluate model performance by calculating accuracy and generating confusion matrices on both training and test data

2. Discuss the method used to optimize the model.

   To optimize the logistic regression model, I used **backward stepwise elimination**. This method involves starting with all selected predictors and then evaluating the statistical significance of each variable using p-values from the model output.

   After fitting the initial model with all four predictors (`SquareFootage`, `NumBedrooms`, `SchoolRating`, and `RenovationQuality`), I reviewed the p-values to assess which variables were statistically significant. While three predictors (`SquareFootage`, `NumBedrooms`, and `RenovationQuality`) had p-values less than 0.05, indicating they were statistically significant, the variable `SchoolRating` had a p-value of 0.725, suggesting it was not a meaningful contributor to the model.

3. Justify the approach discussed in part E2 that was used to optimize the model.

   I chose **backward stepwise elimination** because it is an effective and interpretable approach for identifying the most statistically significant predictors in a logistic regression model. It allows the model to start with all variables and remove any that do not contribute meaningfully, based on their p-values.

   `SchoolRating` was the only variable with a p-value > 0.05, indicating it was not statistically significant for our model. I kept this variable within the model to show transparency, completeness, and highlight the strength of the remaining variables in the model.

4. Summarize at least four assumptions of logistic regression.

   1. **Binary Dependent Variable** – Logistic regression requires that the dependent variable is binary. In this analysis, the dependent variable (`IsLuxury`) meets this assumption as it classifies homes as luxury (1) or not luxury (0).
   2. **Independence of Observations** – Each observation in the dataset should be independent of the others. The housing records should not be repeated or related to each other in a way that would make the model biased.
   3. **Linearity of the Logit** – We use linearity of the logit to predict the probability of something happening, using "logit" or log-odds. There should be a direct relationship between each input variable and the log-odds of the outcome, not the raw probability.
   4. **No Multicollinearity** – Independent variables should not be highly correlated with each other. Having a higher multicollinearity can make it hard to isolate the individual effect of the predictor. As explained on DataCamp, we can run a VIF (variance inflation factor) to test the multicollinearity.

5. Provide evidence that the assumptions from part E4 were verified by providing either a code snippet or a screen shot.

Binary Dependent Variable – `IsLuxury` was confirmed to be binary (1 or 0) using `.value_counts()`

```
df['IsLuxury'].value_counts()
```

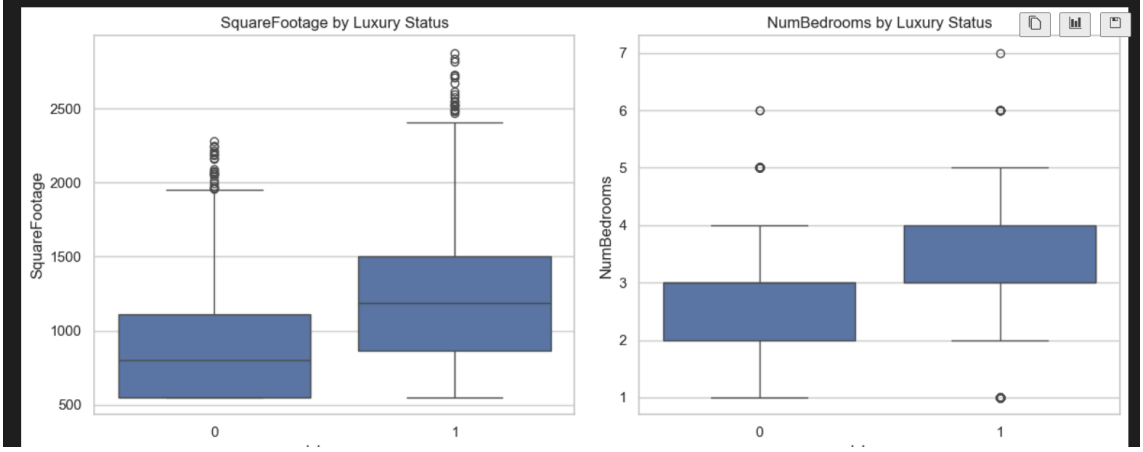
Independence of Observations – Each row in the dataset has a unique record and this was confirmed by checking for duplicates.

```
duplicate_rows = df.duplicated().sum()
print(f"Number of duplicate rows: {duplicate_rows}")
```

Linearity of the Logit – We confirmed linearity using boxplots which showed a consistent increase in the median values of `SquareFootage`, `NumBedrooms`, and `RenovationQuality` for luxury homes.

```
# Task 2 - C4

# Boxplots grouped by IsLuxury
fig, axes = plt.subplots(2, 2, figsize=(12, 10))
axes = axes.flatten()

predictors = ['SquareFootage', 'NumBedrooms', 'SchoolRating', 'RenovationQuality']

for i, var in enumerate(predictors):
    sns.boxplot(x='IsLuxury', y=var, data=df, ax=axes[i])
    axes[i].set_title(f'{var} by Luxury Status')

plt.tight_layout()
plt.show()
```

0.2s



No Multicollinearity – We used Variance Inflation Factor (VIF) to calculate each predictor for multicollinearity.

```
from statsmodels.stats.outliers_influence import
variance_inflation_factor
X_vif = sm.add_constant(X_train)
vif_data = pd.DataFrame()
vif_data["Variable"] = X_vif.columns
vif_data["VIF"] = [variance_inflation_factor(X_vif.values, i) for i
in range(X_vif.shape[1])]
print(vif_data)
```

6. Provide the regression equation and discuss the coefficient estimates

   $\text{logit}(P) = \beta_0 + \beta_1 * \text{SquareFootage} + \beta_2 * \text{NumBedrooms} - \beta_3 * \text{SchoolRating} + \beta_4 * \text{RenovationQuality}$

   Price = -4.7812 + 0.0018*SquareFootage + 0.6504*NumBedrooms – 0.0069*SchoolRating + 0.1910*RenovationQuality

   The interpretation of the coefficient estimates are as follows:
   - `SquareFootage (0.0018)` – for every additional square foot, the log-odds of a home being luxury increases by 0.0018, holding all other variables constant. With the p-value being < 0.05, we can conclude this is a statistically significant predictor.
   - `NumBedrooms (0.6504)`- for every additional bedroom, the log-odds of a home being luxury increases by 0.6504. With the p-value being < 0.05, we can conclude this is a statistically significant predictor.
   - `SchoolRating (0.0069)` – With a very small coefficient and a p-value > 0.05 (0.725), we can conclude this variable is not statistically significant in predicting luxury status.
   - `RenovationQuality (0.1910)` – for every additional point in renovation quality, the log-odds of the home being luxury increases by 0.1910. With the p-value being < 0.05, we can conclude this is a statistically significant predictor.

7. Discuss the model metrics by addressing each of the following:
   - **the accuracy for the test set**
     - The model achieved an accuracy of **70.5%** on the test set, meaning approximately 7 out of 10 homes were correctly classified as either luxury or not luxury by the model when evaluated on unseen data.
   - **the comparison of the accuracy of the training set to the accuracy of the test set**
     - The model had an accuracy of **72.8%** on the training set and **70.5%** on the test set. This slight drop in accuracy on the test set is expected and indicates that the model generalizes well to the dataset and any future dataset changes. The small difference suggests that the model is not overfitting, meaning it performs similarly on both datasets and consistently on new data.
   - **the comparison of the confusion matrix for the training set to the confusion matrix of the test set**
     - Training Set Confusion Matrix:
       - True Negatives (not luxury, correctly predicted): 2,034
       - False Positives (not luxury, incorrectly predicted as luxury): 744
       - False Negatives (luxury, incorrectly predicted as not luxury): 778
       - True Positives (luxury, correctly predicted): 2,044
     - Test Set Confusion Matrix:
       - True Negatives: 492

- False Positives: 202
- False Negatives: 211
- True Positives: 495

8. Discuss the results and implications of your prediction analysis.

   The logistic regression model was able to predict whether a home is classified as luxury with a test set accuracy of 70.5%. This performance is close to the training set accuracy of 72.8%, suggesting that the model generalizes well to new data and is not overfitting.

   The results show that 3 of the 4 predictor variables (SquareFootage, NumBedrooms, and RenovationQuality) had a statistically significant positive impact on the likelihood of a home being classified as "luxury". This means that homes with more square footage, more bedrooms, and higher-quality renovations are much more likely to fall into the luxury category. These findings do align with real-world expectations in the housing market.

   The variable SchoolRating was not statistically significant in this model, indicating that school quality did not meaningfully contribute to whether a home was considered luxury.

   These insights could help real estate professionals, property developers, or appraisers identify key features that increase the likelihood of a home being marketed or priced as luxury. Any marketing strategies could focus on promoting size and renovation quality, and developers could prioritize these features in high-end builds.

9. Recommend a course of action for the real-world organizational situation from part B1 based on your results and implications discussed in part E8.

   Based on the results of our logistic regression analysis, it is recommended that real estate professionals and property developers prioritize square footage, number of bedrooms, and renovation quality when assessing, pricing, or marketing homes as luxury properties.

   Since these 3 features were shown to be statistically significant predictors of luxury classification, marketing campaigns should make it a priority to highlight these characteristics.

   A characteristic like "School Rating" can be assumed to make an impact on home value; however, it did not significantly influence luxury classification. Therefore, it should not be a priority when trying to target luxury buyers.

   This model provides a clear, data-driven foundation for guiding business strategy in the "luxury" housing marketing.

F. Panopto recording

https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=174834f5-9de2-47f8-b1b3-b32601199ea1

G. Sources

https://www.datacamp.com/tutorial/understanding-logistic-regression-python
https://www.datacamp.com/tutorial/variance-inflation-factor