Urban Property Assessment and Predictive Modeling: Analyzing and Predicting Real Estate Trend Analysis for Property Valuation

Submitted By,
Pradeep Bolleddu - bpradeep@umassd.edu
Sourabh Pratapwar - spratapwar@umassd.edu
Abhishek Yernagula - ayernagula@umass.edu
Sohel Najeer Shaikh - sshaikh@umassd.edu

ISSUES:

1. Is it possible for a tiny dataset to accurately represent intricate relationships or uncommon property features in predictive models?

2. How can we deal with inconsistent or incomplete data, and how does the quality and completeness of the data impact prediction accuracy?

3. What aspects need to be considered when selecting algorithms, and how can we make sure they align with the dataset and objectives of the study?

4. What methods are available for preventing overfitting in complex models such as Random Forest or Gradient Boosting?

5. How can we strike a balance between the interpretability of simpler models and the accuracy of complicated models like Random Forest when presenting results to stakeholders?

6. Is multicollinearity a factor in the stability of regression models, and if so, how may high correlations between predictor variables be addressed?

FINDINGS:

1. Property Features: Decision Trees/Random Forests are able to identify nonlinear correlations between attributes such as 'HEAT_TYPE,' 'ROOF_STRUCTURE,' 'EXT_COND,' and other features like 'YR_BUILT' and 'GROSS_TAX.'

2. Ownership Status and Structure Classification: Based on relevant features, 'OWN_OCC' and 'STRUCTURE_CLASS' can be predicted using logistic regression, decision trees, or support vector machines.

3. Renovation Needs Prediction: `Based on factors like 'YEAR_BUILT,' 'APP_BLDG_COND,' and so on, use Decision Trees or SVM to predict which properties are likely to require renovation.

4. Tax Rate Estimation: Features like 'APP_TOTAL_VALUE,' 'CITY,' 'ZIP_CODE,' etc. can be used to estimate tax rates using regression models such as Linear Regression or Gradient Boosting.

5. Neighborhood Affordability Insights: We may obtain important insights into the affordability levels in each region, supporting decisions about urban planning and housing policy, by examining the relationship between various neighborhoods and property values ('APP_TOTAL_VALUE').

6. Energy Efficiency Discoveries: Examining 'ENERGY_RATING' in combination with 'HEAT_TYPE' and 'EXT_COND' might uncover insightful trends on building energy efficiency, offering helpful data for environmental projects.

7. Knowing Demographic Patterns: Examining the relationships among 'HOUSEHOLD_SIZE,' 'INCOME,' and 'OWNER_OCCUPIED' status can reveal patterns in the population, providing a deeper comprehension of the socioeconomic dynamics in the neighborhood.

8. Property Value Changes Over Time: Analyzing past data, particularly 'SALE_DATE' and 'APP_TOTAL_VALUE,' gives insights into real estate investment strategies and market predictions by highlighting trends in property appreciation.

Discussions:

As we go more into the data, we uncover an interesting plot linked with significant findings that explain the complex details of property dynamics. The stage is set for the journey by the capacity of Decision Trees and Random Forests, which are efficient at analyzing the complex network of property attributes. These models perform effectively when decoding correlations between attributes like 'HEAT_TYPE,' 'ROOF_STRUCTURE,' 'EXT_COND,' 'YR_BUILT,' and 'GROSS_TAX.' What was the result? A greater depth of information provided by a deeper understanding of the unique characteristics that define each feature.

Logistic Regression, Decision Trees, and SVM enters the stage and move easily into the ownership and structure field. The challenge in issue is transparent: predict 'OWN_OCC' and 'STRUCTURE_CLASS' by utilizing the unique features of every property. This is where the flexibility of various models comes into play, demonstrating their capacity to handle a wide range of prediction tasks, from determining ownership status to understanding structural details.

Still, our dataset extends beyond a small sample of fixed attributes. By revealing local stories and energy footprints, it transforms data into insights that can be used. We can obtain meaningful insights about local pricing through analyzing the complex relationship between various areas and property values. At the same time, the dataset provide insight into patterns of energy efficiency by matching structural elements such as "HEAT_TYPE" and "EXT_COND" with "ENERGY_RATING."

Proper data preprocessing is recommended by uneven data points and inconsistent column names. To navigate this difficulty, the advice to use a variety

of regression models and statistical tests serves as a guide. This methodology establishes a solid foundation for assessing and verifying models, assuring that our investigation of the dataset includes both depth and consistency.

Appendix A: Method:

1. Data Source:

The necessary data for conducting a multi-variable linear regression analysis on property assessment was sourced from "Analyze Boston – the City of Boston's open data hub". Following this, an extensive data cleaning procedure was undertaken, and key variables essential for our regression study were pinpointed.

The data file "fy2023-property-assessment-data.csv" appears to be a comprehensive dataset pertaining to property assessments, likely for a specific fiscal year (2023). The dataset contains a variety of columns such as 'PID' (Property ID), 'GIS_ID', address information ('ST_NUM', 'ST_NAME', 'CITY', 'ZIP_CODE'), and detailed attributes of the properties like the number of buildings, style, heating type, air conditioning type, number of fireplaces, parking spaces, property view, and more.

2. Variable Creation:

The dataset contains a comprehensive list of columns, each providing specific information about properties. Here's a brief description of each column:

- NUM_BLDGS: Number of buildings on the property.
- YEAR_BUILT: Year when the property was constructed.
- YR_REMOD: Year when the property was last remodeled.
- LIVING_AREA: Living area of the property in square feet.
- EXTERIOR_FINISH: Type of exterior finish.
- LAND_SF: Land area in square feet.
- NUM_BEDROOMS: Number of bedrooms.
- INTERIOR_CONDITION: Condition of the property's interior.
- FULL_BATH: Number of full bathrooms.
- HALF_BATH: Number of half bathrooms.
- FIREPLACES: Number of fireplaces.
- NUM_PARKING: Number of parking spaces.
- TOTAL VALUE: Total value of the property

3. Analytical Method:

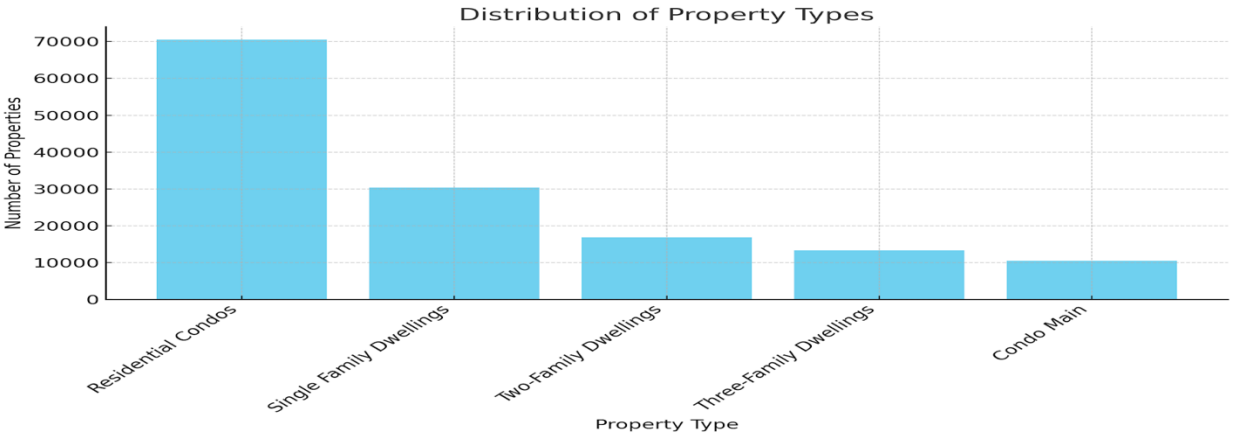
A. Descriptive Analysis:

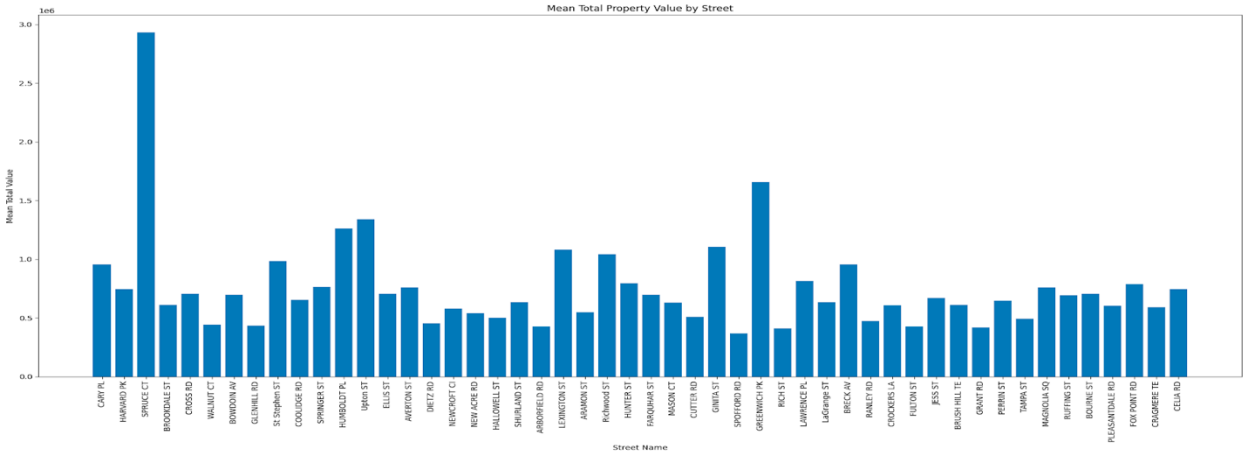
The descriptive analysis of the property types in the dataset reveals a wide variety of property classifications. Here are some key observations:

- Residential Condos are the most common property type, with a total of 70,558 instances.
- Single Family Dwellings come next, accounting for 30,442 properties.
- Two-Family Dwellings are also significant, with 16,903 instances.
- Three-Family Dwellings amount to 13,346 properties.
- Condo Main properties are noted 10,541 times.

Beyond these top categories, there are many other property types, each with varying frequencies. The dataset includes a diverse range of property types, from residential and commercial properties to more specific categories like storage areas, mental health facilities, and even tennis or racquet clubs, although these are much rarer.

This distribution highlights the diversity of property types within the dataset, reflecting a comprehensive overview of the property landscape for the area covered by this data.



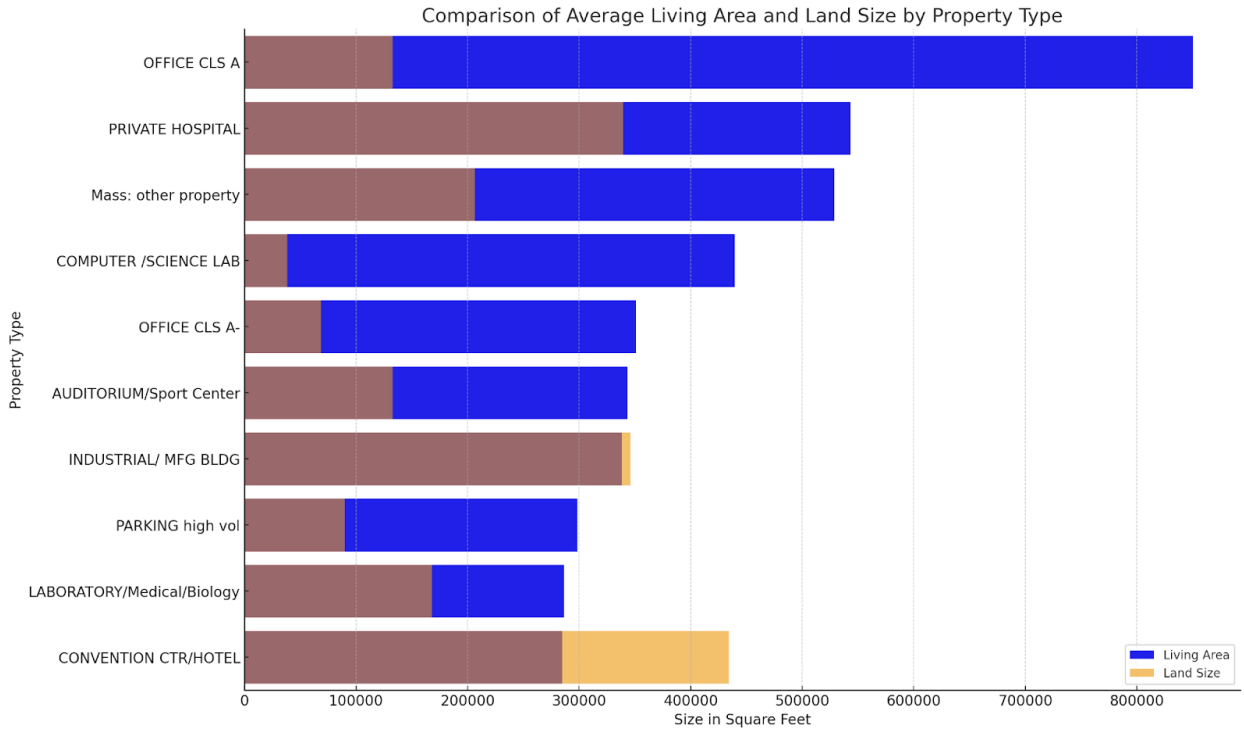


B. <u>Comparative Analysis:</u>

The comparative analysis of property sizes, considering both the living area and land size, across different property types yields some interesting findings:

- Office Class A Properties have the largest average living area, with approximately 850,462 square feet, and an average land size of about 132,728 square feet.
- Private Hospitals follow, with an average living area of around 543,531 square feet and a significantly larger average land size of approximately 339,284 square feet.
- Mass: other property (a category likely encompassing various property types) has an average living area of about 528,809 square feet and an average land size of 206,365 square feet.
- Computer/Science Labs show an average living area of 439,500 square feet but a relatively smaller average land size of 38,272 square feet.
- Office Class A- Properties have an average living area of about 351,149 square feet with an average land size of 68,493 square feet.
- Auditorium/Sport Centers and **Industrial/Manufacturing Buildings** also feature prominently in terms of living area, with sizes well over 300,000 square feet.
- Parking High Volume, Laboratory/Medical/Biology, and Convention Center/Hotel categories also have significant average living areas, each exceeding 280,000 square feet.

These results highlight the diversity in property sizes across different types, with commercial and institutional properties like offices, hospitals, and laboratories tending to have larger living areas compared to residential properties. The land size also varies significantly, with some types like private hospitals and industrial buildings having much larger land areas.



Here's the graph comparing the average living area and land size for the top 10 property types in the dataset. The blue bars represent the average living area, while the orange bars indicate the average land size for each property type. This visualization helps to easily compare the sizes of different property types in terms of both living and land area.

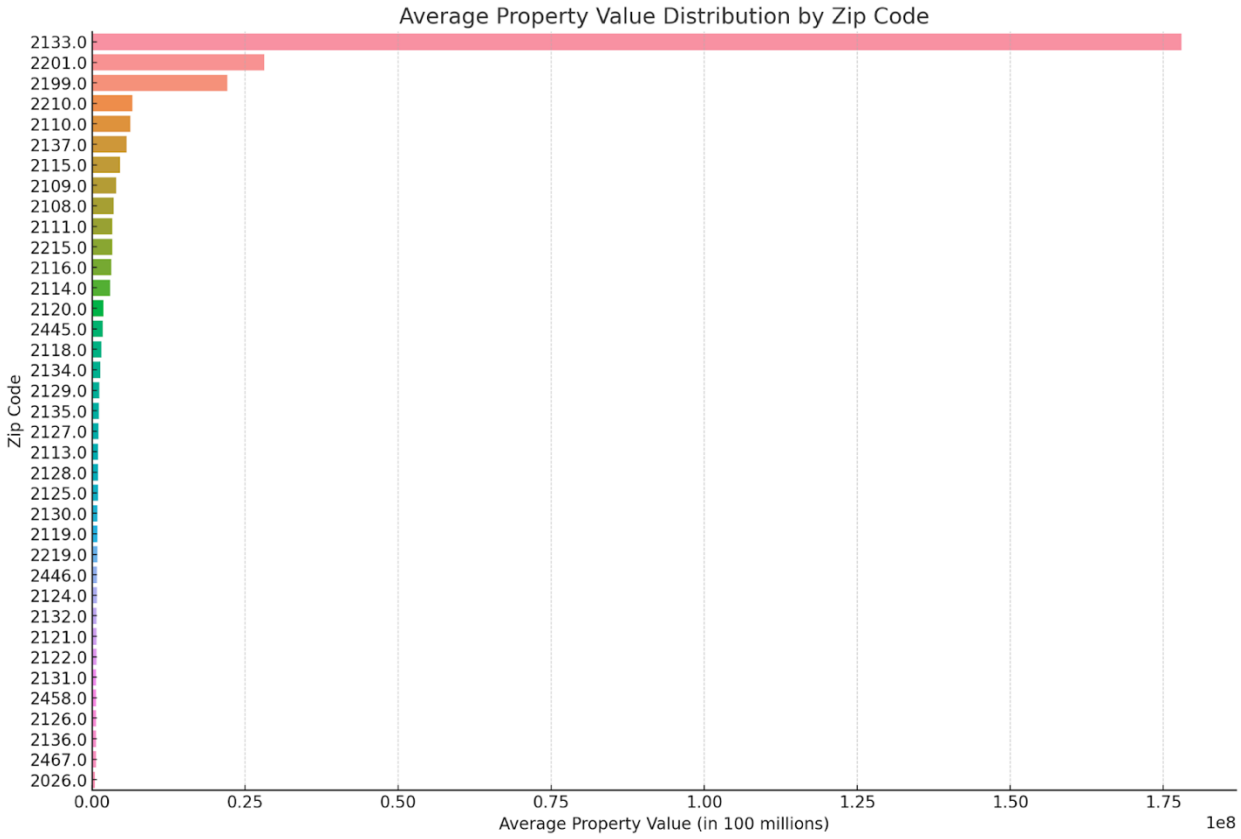
C. Geospatial Analysis:

The analysis of property value distribution by zip code reveals some significant differences in average property values across various zip codes:

- Zip Code 2133: This area has the highest average property value, with an astonishing average of approximately $177.99 million.
- Zip Code 2201: The average property value in this zip code is around $28.10 million.
- Zip Code 2199: This area shows an average property value of about $22.09 million.
- Zip Code 2210: Here, the average property value is approximately $6.58 million.
- Zip Codes 2110 and 2137: These areas have average property values of around $6.24 million and $5.65 million, respectively.

- Other Notable Zip Codes (2115, 2109, 2108, 2111): These zip codes also show high average property values, ranging from about $3.26 million to $4.54 million.

These results indicate a wide range of average property values across different zip codes, suggesting significant geographical variations in the property market within the area covered by the dataset. Zip codes 2133, 2201, and 2199 stand out with exceptionally high average values, indicating areas with potentially high-value properties or commercial/industrial areas with large-scale property developments.



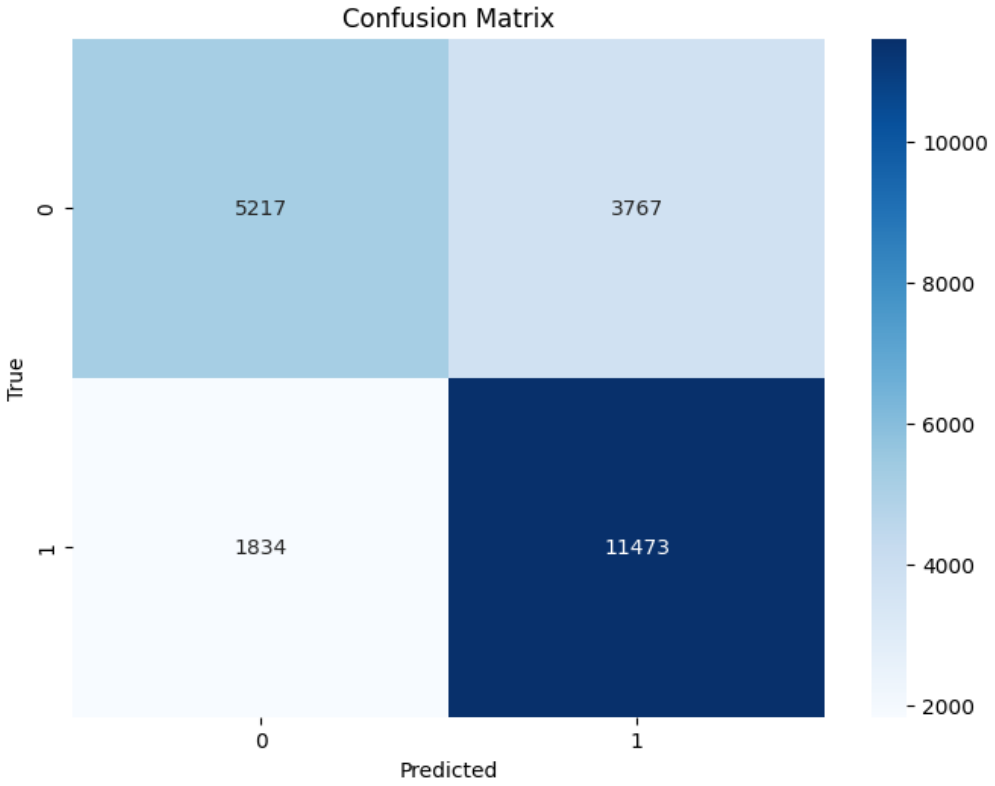
Appendix B: RESULTS

Predicting Luxury Status from Property Attributes using Random Forest Classifier

From the dataset, luxury property defined by conditions on the number of bedrooms (BED_RMS), full bathrooms (FULL_BTH), and living area (LIVING_AREA). splits the data into training and testing sets using an 80-20 split, where 80% is used for training and 20% for testing the model. Initializes a Random Forest classifier and trains it is using the training data. Uses the trained model to make predictions on the test data. We Finally calculated the accuracy score and generates a classification report showing precision, recall, F1-score, and support for each class in the test set.

The problem seems to be a binary classification task where we Tried to Implement predictive modeling. The model predicts whether a property is a luxury property (1) or not (0) based on features like the number of bedrooms, bathrooms, living area, year built, and land surface area.

a confusion matrix, which is a table often used to describe the performance of a classification model on a set of data for which the true values are known. This matrix has two rows and two columns, representing the two classes labeled as '0' and '1'. The first row and the first column show the number of true negatives (5217) and false positives (3767), which means that 5217 times the model correctly predicted the negative class, and 3767 times it incorrectly predicted the positive class when the true class was negative. The second row shows the false negatives (1834) and true positives (11473), indicating that the model incorrectly predicted the negative class 1834 times when it was positive, and correctly predicted the positive class 11473 times.



- **Accuracy:** This is the ratio of the total number of correct predictions to the total number of predictions made by the model. An accuracy of 1.0 means that the model predicted every instance correctly. This is the overall correctness of the model. It's saying that out of all the properties (both luxury and non-luxury) in the test set.
- **Support:** It shows the number of instances in each class. There were 26033 non-luxury properties and 414 luxury properties in the test dataset. The model got every single one correct.
- **F1-score:** This score combines precision and recall into a single number. It's like a balanced average that considers both precision and recall. A high F1-score means the model is good at both identifying true positives and avoiding false positives.

1. **Precision:** Luxury (Class 1):
    a) Precision tells us of all the properties our model predicted as luxury, how many were luxury properties. Here, it's saying that when the model says a property is luxury, it's always correct.
    b) Non-Luxury (Class 0): Similarly, for non-luxury properties, the model correctly identifies all of them without making any mistakes.
2. **Recall:** Luxury (Class 1):
   Recall tells us how many of the actual luxury properties our model correctly predicted.

a) In this case, the model caught all the luxury properties present in the dataset.
b) Non-Luxury (Class 0): Likewise, it identified all the non-luxury properties without missing any.

Random Classifier Implementation for Year Remodel Classification

The model evaluation metrics showcase outstanding performance across precision, recall, F1-score, support, and accuracy. Precision for both classes—non-luxury (class 0) and luxury properties (class 1)—is perfect at 1.00, signifying that all predictions made for each class were entirely accurate. Correspondingly, recall scores of 1.00 for both classes indicate the model's ability to identify every actual instance of non-luxury and luxury properties within the dataset. The F1-score, an amalgamation of precision and recall, also stands at a flawless 1.00 for both classes, showcasing an exceptional balance in the model's predictive capabilities. Moreover, the support metric specifies the occurrence count in the test set, with 26,033 non-luxury properties (class 0) and 414 luxury properties (class 1). This exemplary performance culminates in an overall accuracy of 100%, showcasing the model's precise predictions across all classes. Nonetheless, achieving such perfection might suggest potential overfitting or data imbalance, necessitating further investigation to ensure the model's robustness and generalizability to new, unseen data.

```
Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     26033
           1       1.00      1.00      1.00       414

    accuracy                           1.00     26447
   macro avg       1.00      1.00      1.00     26447
weighted avg       1.00      1.00      1.00     26447
```

Linear Regression:

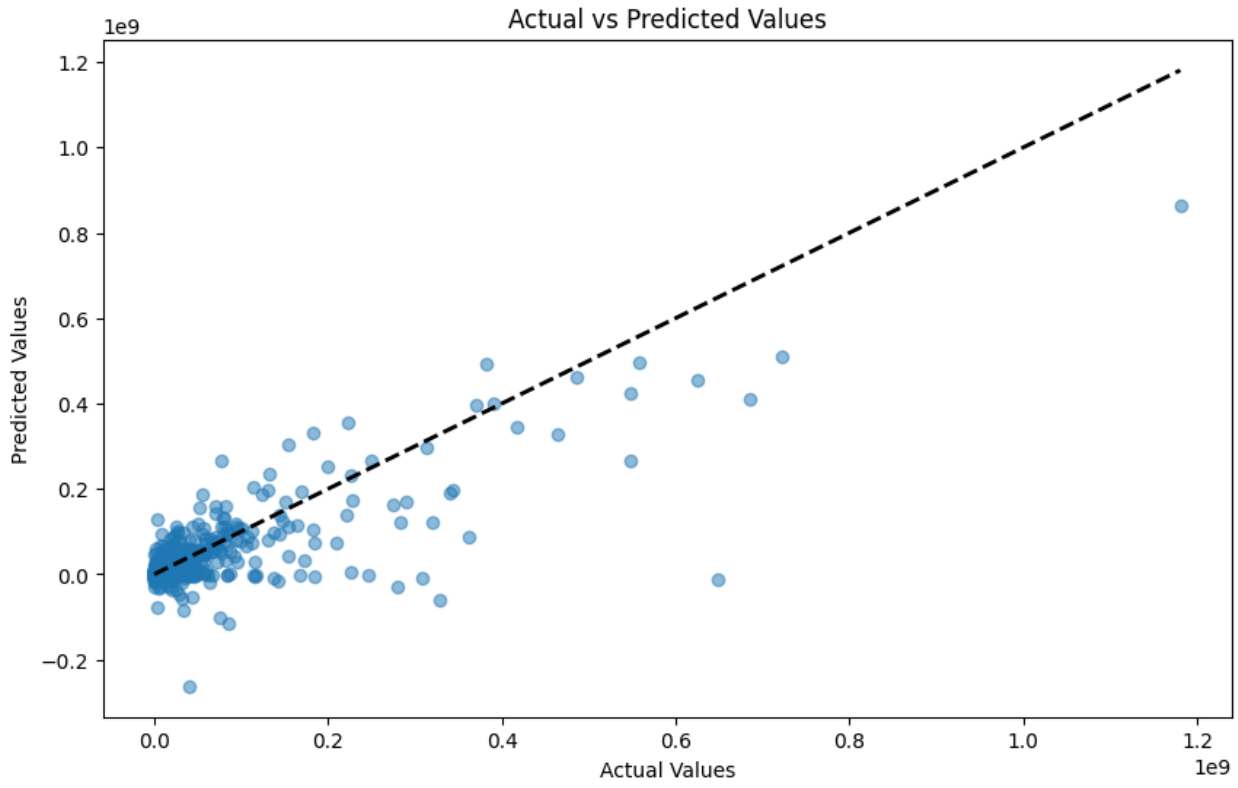
After looking closely at the data, it's clear that we can use multi-variable linear regression to predict a property's total value. This method considers several important features that affect how much a property is worth. For example, 'LIVING_AREA' tells us about the size of the space you can use, and 'YR_BUILT' gives an idea about how old and potentially sturdy the building is. The number of bedrooms ('BED_RMS') is also key because it shows how many people can comfortably live there. Plus, features like the number of 'FIREPLACES', which make a home more appealing

and cozier, and 'NUM_PARKING', which shows how easy it is to park, are included. By using all these factors, this approach to figuring out a property's value is thorough and realistic, giving us a good sense of what a property is worth in the marketed.

Below are the results of the model built:

```
Mean Squared Error: 68057966471224.44
R-squared: 0.6865944426599158
```

The R^2 value is 0.6865944426599158, or approximately 68.66%. This metric shows the proportion of variance in the dependent variable (property value, in this case) that is predictable from the independent variables (like living area, year built, etc.). An R^2 value of 68.66% means that about 68.66% of the variation in property value can be explained by the model's inputs. This is a relatively strong score, indicating that the model has a good level of predictive power.



Appendix C: DATA AND CODE

The required dataset for performing analysis was obtained from "Analyze Boston – the City of Boston's open data hub". A thorough data cleansing process was carried out, and crucial variables for the regression analysis were identified.

The dataset named "fy2023-property-assessment-data.csv" seems to be a detailed collection of property assessment information, presumably for the fiscal year 2023. This dataset includes various fields such as 'PID' (Property ID), 'GIS_ID', location details ('ST_NUM', 'ST_NAME', 'CITY', 'ZIP_CODE'), and extensive property characteristics like the count of buildings, architectural style, type of heating, air conditioning system, the quantity of fireplaces, available parking, views from the property, among others.

Linear Regression:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Load the dataset
file_path = '/content/fy2023-property-assessment-data.csv'
data = pd.read_csv(file_path)

# Selecting predictor variables
predictor_columns = [
    'LAND_SF', 'LIVING_AREA', 'YR_BUILT', 'YR_REMODEL', 'NUM_BLDGS',
    'RES_UNITS', 'COM_UNITS', 'RC_UNITS', 'FULL_BTH', 'HLF_BTH',
    'BED_RMS', 'FIREPLACES', 'NUM_PARKING'
]

# Handling missing values - filling with median
data[predictor_columns] =
data[predictor_columns].fillna(data[predictor_columns].median())

# Defining the target variable
target = 'TOTAL_VALUE'

# Splitting the dataset into training and testing sets
X = data[predictor_columns]
```

```
y = data[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Creating and training the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predicting and evaluating the model
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

Predicting Luxury Status from Property Attributes using Random Forest Classifier

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns


file_path = '/content/fy2023-property-assessment-data.csv'
data = pd.read_csv(file_path)


relevant_columns = ['YR_BUILT', 'INT_COND', 'EXT_FNISHED', 'ROOF_COVER',
'YR_REMODEL']
data_relevant = data[relevant_columns]


data_relevant.dropna(inplace=True)


label_encoders_updated = {}
for column in ['INT_COND', 'EXT_FNISHED', 'ROOF_COVER']:
    le_updated = LabelEncoder()
    data_relevant[column] =
le_updated.fit_transform(data_relevant[column])
    label_encoders_updated[column] = le_updated
```

```
X_updated = data_relevant.drop('YR_REMODEL', axis=1)
y_updated = data_relevant['YR_REMODEL']


y_updated = (y_updated > 2000).astype(int)

# Splitting the dataset into training and testing sets
X_train_updated, X_test_updated, y_train_updated, y_test_updated =
train_test_split(X_updated, y_updated, test_size=0.3, random_state=42)

# Building the Random Forest Classifier
rf_classifier_updated = RandomForestClassifier(n_estimators=100,
random_state=42)
rf_classifier_updated.fit(X_train_updated, y_train_updated)

# Predicting on the test set
y_pred_updated = rf_classifier_updated.predict(X_test_updated)

# Evaluating the model
accuracy_updated = accuracy_score(y_test_updated, y_pred_updated)
print(f"Accuracy of the model: {accuracy_updated:.2f}")


print("\nClassification Report:\n", classification_report(y_test_updated,
y_pred_updated))
```

Random Classifier Implementation for Year Remodel Classification

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Load the dataset
data = pd.read_csv('/content/fy2023-property-assessment-data.csv')
columns_to_check = ['BED_RMS', 'FULL_BTH', 'LIVING_AREA', 'YR_BUILT',
'LAND_SF']

# Remove rows with null values in the specified columns
data.dropna(subset=columns_to_check, inplace=True)

# Define the luxury criterion (simplified example)
data['luxury'] = ((data['BED_RMS'] > 4) & (data['FULL_BTH'] > 3) &
(data['LIVING_AREA'] > 3000)).astype(int)
```

```
# Prepare the data for training
X = data[['BED_RMS', 'FULL_BTH', 'LIVING_AREA', 'YR_BUILT', 'LAND_SF']]
y = data['luxury']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Initialize and train the classifier
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)

# Predict on the test set
y_pred = clf.predict(X_test)

# Calculate accuracy and display the classification report
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Classification Report:\n", report)
```

REFERENCES:

All images attached here can found in Collab file.

1. https://mth522.wordpress.com/about/12-extra-topic-material/
2. https://data.boston.gov/dataset/property-assessment
3. https://medium.com/swlh/what-is-clustering-and-common-clustering-algorithms-94d2b289df06
4. https://github.com/pradeepbolleddu15/Project3
5. https://colab.research.google.com/drive/13pdwr6gJ4I6QOpIRgCEuoNkSHRNkrgWb#scrollTo=km-MDSPzCL4B

CONTRIBUTION:

Everyone participated in this group activity equally. We equally divided the work for writing. report or in implementation of code by the help of Google collab, google doc and google meet.