Applied Machine Learning

Real Estate Predictive Analytics using Decision Trees and Feature Importance Models

Chukwunonso Nnodum

Contents

Background and Context

Domain

The problem statement we are addressing is of the **Real Estate** domain. The dataset undertaken for this project deals with 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa. This document shall further provide the analysis done by me on the data and the insights derived. This shall further help the home buyer to fully understand the pricing of houses based on specific features.

Brief Description of the scenario

Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But this playground competition's dataset proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

Decision(s) of interest

Identification: Given the insights from the house price data file, the critical business decision is to identify the target home buyers group by implementing a house vetting support program to predict house prices. This program would be designed to address the disparities in houses based on various features of the house.
Visualization: Once we do the analytics, we need to make the data presentable to the stake holders and the decision makers. Clean data tells a good story, i.e. visualizations will make it easier to understand the large data set so that the decisions can be implemented later by the real estate company.

Situation Assessment

Brentford Real Estate is a prestigious institution known for its commitment to giving customers the best their money can buy. It stands at a pivotal juncture for the 3rd quarter of 2024. The firm has made a resolute decision to intensify its focus on improving the quality of potential homeowners that they attract. Brentford Real Estate owning acres of residential homes in Ames, Iowa, has a long-standing reputation for real estate excellence. However, recent concerns regarding the quality of its customers have prompted the firm to reevaluate its quality of homes available. Currently, the house features data of 2024 is provided by the firm. Based on this data, certain conclusions are to be made.

The dataset is to be used for determining the price of homes based on certain features that a homeowner may find attractive.

To conclude, the comprehensive dataset would serve as the foundation for the efforts, enabling Brentford Real Estate to identify buyers and investors who will contribute to the rich cultural tapestry of the institution while ensuring that every applicant has a fair chance to enjoy their new home.

Business Objective

The stakeholders at Brentford Real Estate are expecting to gain multiple inputs from the analysis that would be provided. Key objectives we would work towards are as mentioned below:

1. ***Attract Quality Home Buyers:*** One of the main objectives for stakeholders in the firm is to attract high-quality home buyers. This includes buyers who are financially stable, pre-approved for mortgage and willing to invest in property.

2. ***Increase Real Estate Ranking***: The firm aims to improve their rankings. They want to do so by maintaining quality buyers and investors.
3. ***Assessing Support Systems***: The firm wants to know about their customers to gauge the kind of houses they are interested in and guide them to make the perfect choice.

Data Mining goals

The goal is to filter the data for Grade (above) Living Area, total basement square feet, wood deck square feet, Garage Area, Year built, 2nd floor square feet and Year built. These filters are done to identify and create patterns that make the dataset clearer and understandable. I came about these features by applying a feature importance model in my coding.

Data requirements

Data sorting tools were applied to the raw data. Rearranging the data set from largest to smallest to find that the most expensive houses don't really prioritize having big open porches and wood decks. Also, notice that as the houses get cheaper, they tend to not come with paved drive ways, hence, tend to have lesser garage space for cars. This would make the dataset easier to understand and make us able to derive problems that we can solve in the future.

Describe data.

The data provided is an excel file with 1461 rows and 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa. The data consists of house features. This data is a mix of numeric and categorical variables. The column name "Id" is the primary identifier for the houses.

Sources

The main data file was provided by the professor and is assumed to be a **PRIMARY DATA SET.**

Quality

The Data has some hidden references and patterns, when we sort and dive deep into the data, we find that the data has much more to tell than observing to at a superficial level. We need to data mine into the file to get insights and apply analytical tools such as sub-setting, sorting, filtering along with mathematical functions such as average and min, max. i.e., clean the data before we visualize the same.

Data Preparation

Data Selection

Out of the 79 variables, we selected:

1. Garage Area,
2. Total Basement SF,
3. Wood Deck SF,
4. Overall qual
5. Gr Liv Area ("Grade (above) Living Area"),
6. 2nd Floor SF,
7. Year built.

‘Sale Price’ attribute plays an important role and acts as a target variable. The other variables are used as input parameters for the model to execute the result.

Data Cleaning

As we proceeded to the next part of building a model, it was necessary to focus on the filling missing values of the dataset, splitting them into training and test sets at a 0.7 and 0.3 measurement respectively.

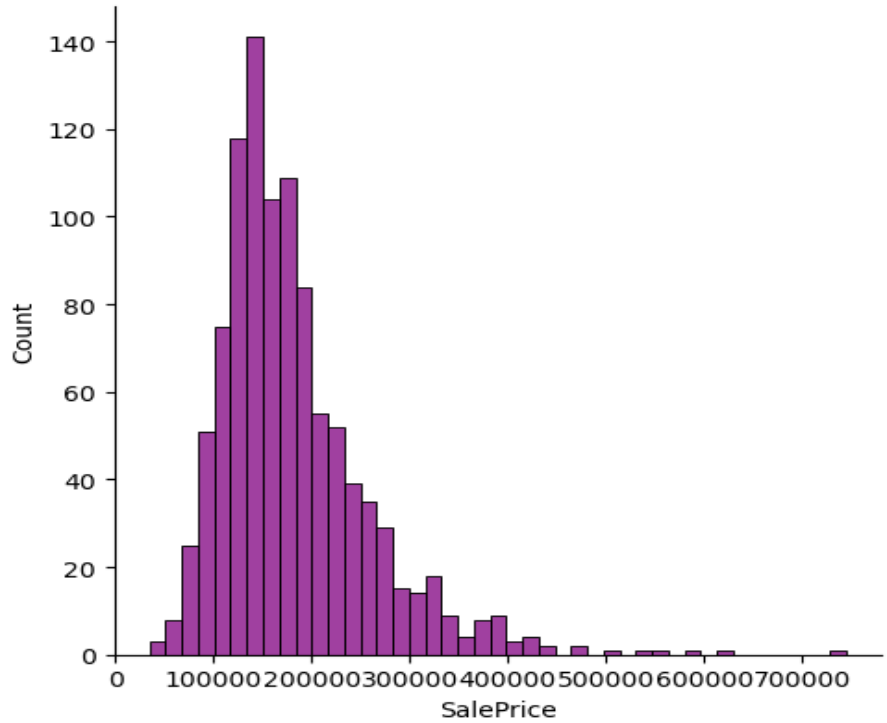
```
# Fill missing values with -1
data.fillna(value=-1, inplace=True)

# Split the data into training and test sets
train_data, test_data = train_test_split(data, test_size=0.3, random_state=42)

# Display the shape of the training and test sets
print("Training data shape:", train_data.shape)
print("Test data shape:", test_data.shape)
```

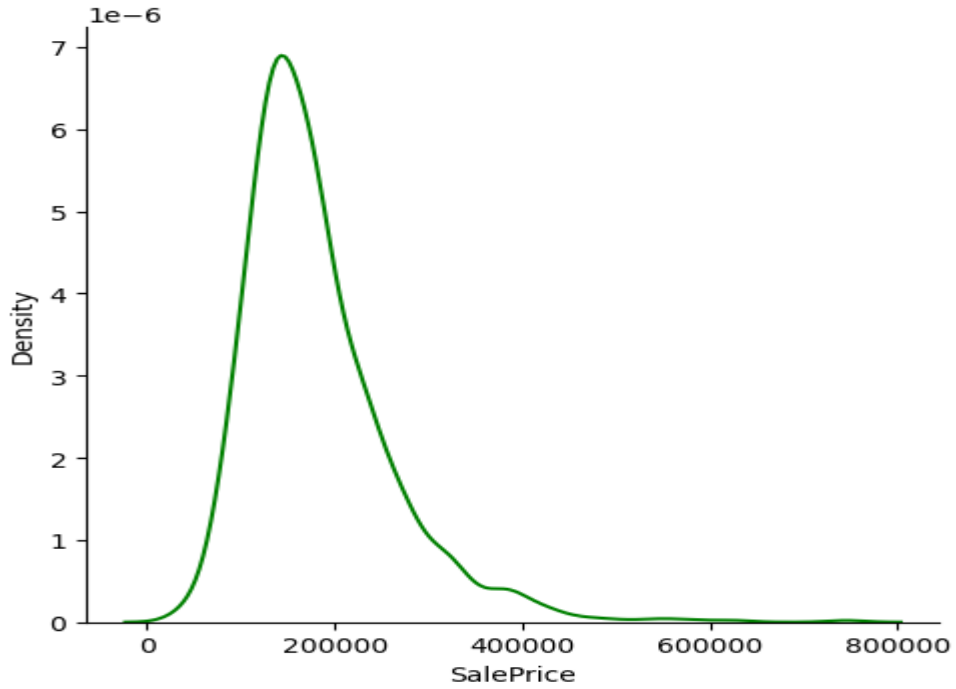
Data Preparation

The firm’s dataset is then subset based on various attributes. For instance, to know and study the dataset and then create histograms to show frequencies of each attribute.

```
#The x-axis represents the values of the 'SalePrice' feature, which is the variable being plotted.
sns.displot(df_num['SalePrice'], kind="hist", color="purple")
#The y-axis represents the frequency or count of occurrences of each value of the 'SalePrice' feature.
sns.displot(df_num['SalePrice'], kind="kde", color="green")
```

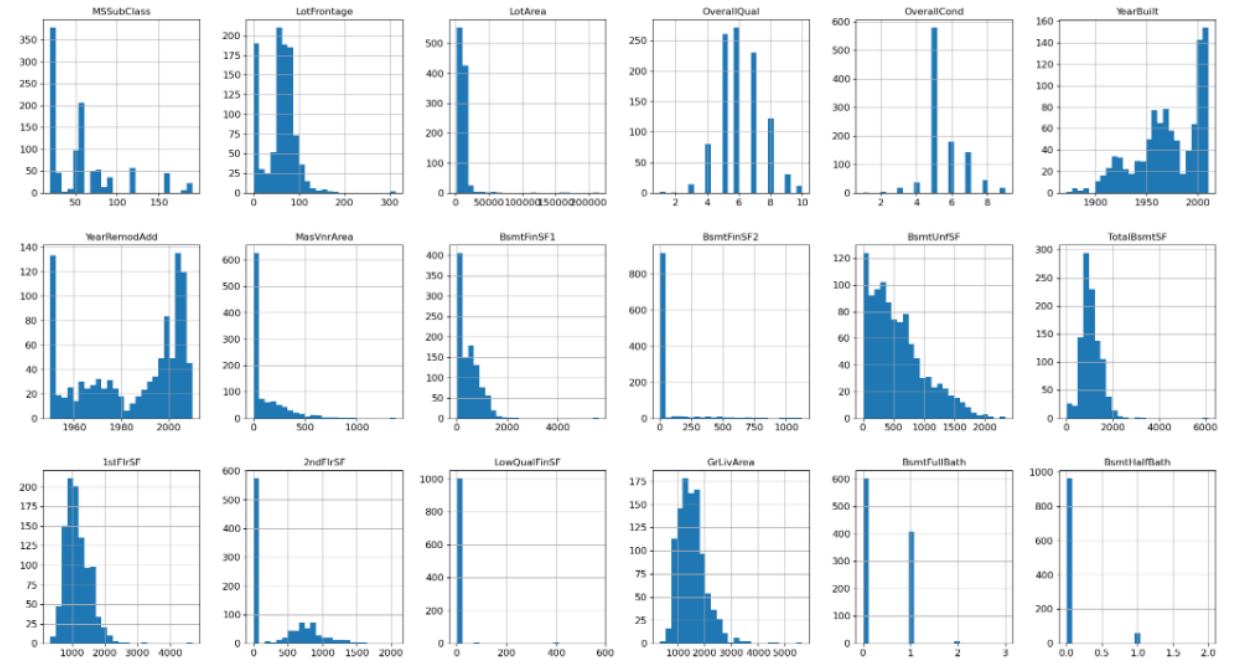
```
<seaborn.axisgrid.FacetGrid at 0x2c11a3cb610>
```

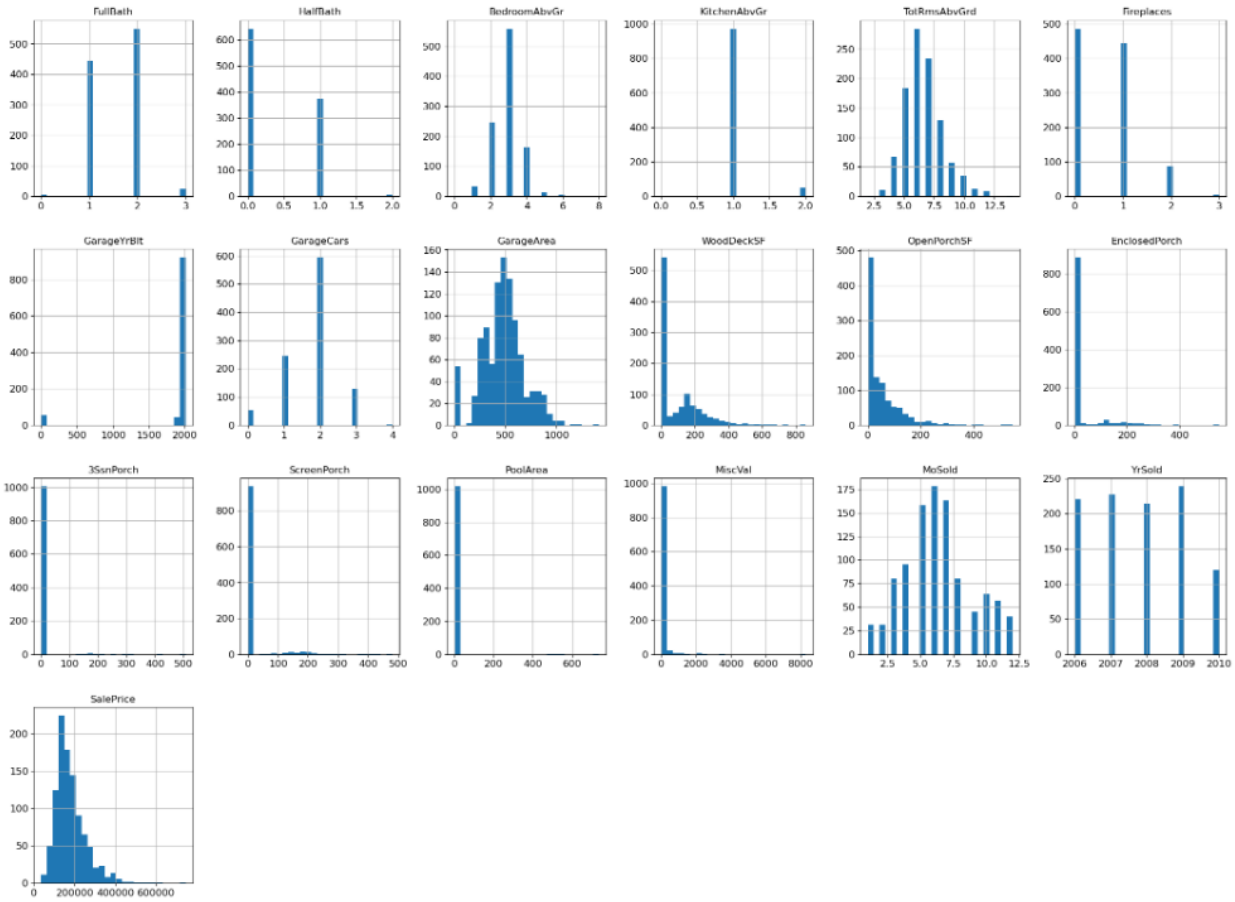




Histograms of other variables

```
df_num.hist(figsize=(26, 34), bins=25, xlabelsize=12, ylabelsize=12);
```

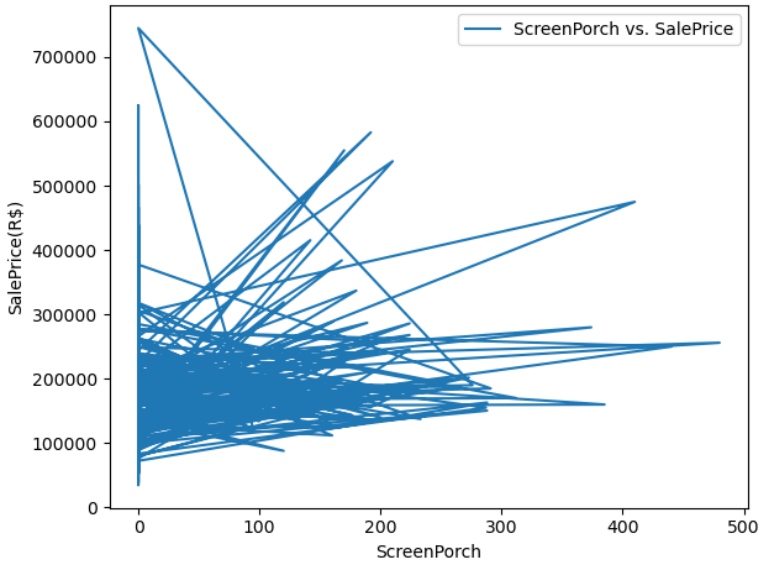




I was curious to see the relationship between saleprice and total basement square feet through a histogram to spot trends but I realized that the price of houses doesn't necessarily go up as basement

size increases.

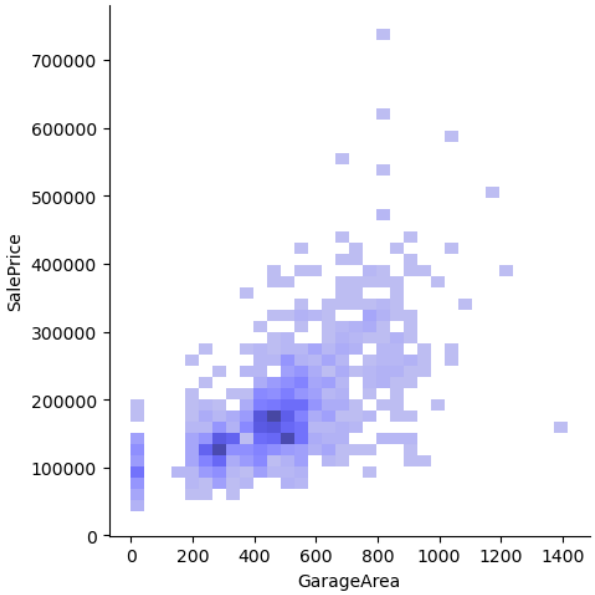
```
#This code snippet creates a scatter plot of 'TotalBsmtSF' (total basement square footage) against 'SalePrice'
plt.plot(df_num['ScreenPorch'],df_num['SalePrice'], label='ScreenPorch vs. SalePrice')
plt.xlabel('ScreenPorch')
plt.ylabel('SalePrice(R$)')
plt.legend()
plt.tight_layout()
```




```
sns.displot(data=df_num, x='GarageArea', y='SalePrice', kind='hist', color='blue', height=5)
```

```
C:\Users\nnodu\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

```
<seaborn.axisgrid.FacetGrid at 0x1cb830b1210>
```



I also looked at saleprice and garage area and noticed a similar trend

Modelling – Building Models

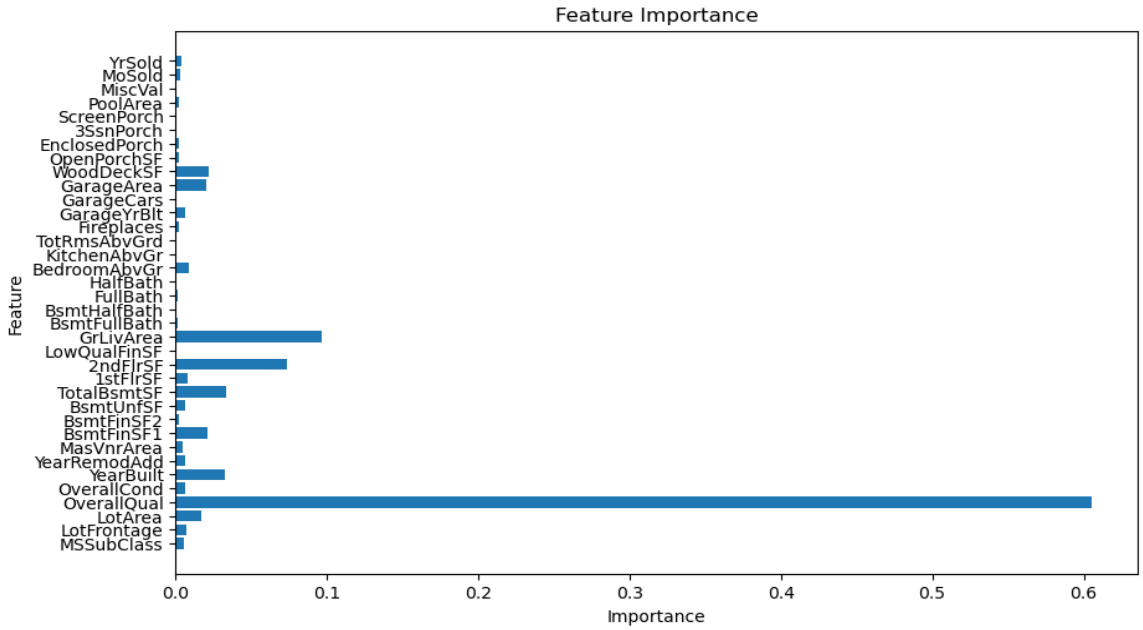
Data in detail

8 attributes were picked from the dataset. The following table displays the numerical and categorical attributes in a distinguished manner along with its measurement scale.

I picked these features specifically after training by using all features a feature importance model and ranking.

```
# Get feature importances
feature_importances = model.feature_importances_

# Zip feature names and importances
features = X_train.columns
feature_importance_pairs = zip(features, feature_importances)

# Sort feature importance pairs by importance
sorted_feature_importance_pairs = sorted(feature_importance_pairs, key=lambda x: x[1], reverse=True)

# Print or visualize feature importance
for feature, importance in sorted_feature_importance_pairs:
    print(f"{feature}: {importance}")
```

```
# Optionally, visualize feature importance
plt.figure(figsize=(10, 6))
plt.barh(features, feature_importances)
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.title('Feature Importance')
plt.show()
```



Attribute Name	Variable Type	Measurement Scale
Garage area	Numerical	Ratio
Year built	Numerical	Nominal
Wood deck	Numerical	Ratio
OverALL quality	Numerical	Ordinal
Total basement sf	Numerical	Ratio

Grliv area	Numerical	Ratio
2nd floor SF	Numerical	Ratio

What type of decision-making model is appropriate for the decision-making tasks?

Based on our target variable which is "Sale price", we concluded of moving ahead with building a Decision Trees regression model for prediction of price of houses because I was able to spot nonlinear relationships between multiple variables and sales price and the flexibility of decision trees model is capable of handling such problems. Decision trees can also handle both categorical and numerical features, which is important in real estate where features such as property type (categorical) and square footage (numerical) may influence the sales price.

Decision trees provide a measure of feature importance, indicating which features have the most significant impact on the predicted sales price. This can provide valuable insights into the factors driving property values and inform decision-making for buyers, sellers, and real estate professionals. Feature importance will be shown later.

PYTHON

1. Handling non-linearity – As classification trees are known for handling nonlinear data, we used it to interpret the results for our analysis. They recursively split the data based on features, allowing us to model intricate decision boundaries.
2. Handling missing data – The classification tree is equipped to handle null values and thus taking this to our advantage, we used this method.

Detailed model development and output.

For Decision Tree,

1. Firstly, read the csv file in excel.
2. Sort the data set to make it more readable and understandable so that we can identify trends.
3. Identify important features that significantly affect Sale Price.
4. get the default tree.
5. Get the Mean Square and Absolute Errors.

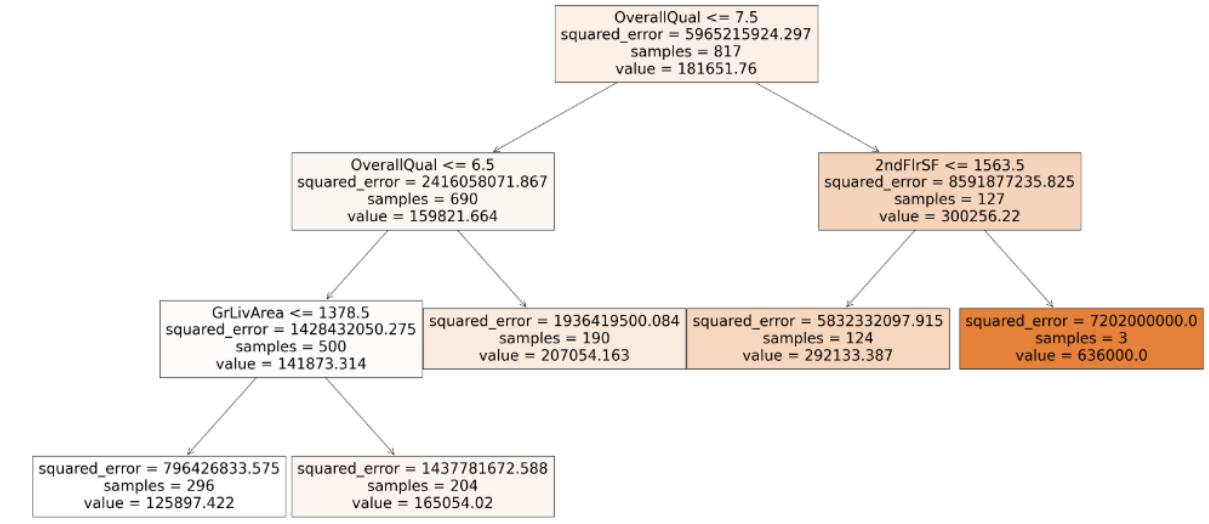
It's a comprehensive analysis involving data preprocessing, model training, evaluation, and comparison.

The output for the model is as follows:

```python
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree


#adding multiple features to the mix
#Define Features (X) and Target Variable (y):
X = df_num[['OverallQual', 'GrLivArea', '2ndFlrSF', 'YearBuilt', 'TotalBsmtSF', 'GarageArea', 'WoodDeckSF']]  # Features
y = df_num['SalePrice']      # Target variable

# Split the data 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)

# Train a decision tree regressor
model = DecisionTreeRegressor(random_state=42)
model.fit(X_train, y_train)
```

```
DecisionTreeRegressor(random_state=42)
```

```python
# Train a pruned decision tree regressor
model_pruned = DecisionTreeRegressor(max_depth=5, min_samples_split=5, random_state=42, max_leaf_nodes=5)
model_pruned.fit(X_train, y_train)

# Visualize the decision tree
plt.figure(figsize=(40,20))
plot_tree(model_pruned, feature_names=X.columns.tolist(), filled=True)
plt.show()
```



```
#ACCURACY
from sklearn.metrics import accuracy_score


# Calculate accuracy
model_score = model.score(X_test, y_test)
print(f"Accuracy: {model_score}")
```

```
Accuracy: 0.7408947933449082
```

Model Evaluation

Classification Tree: On running our developed model for Decision tree, the model accuracy was found to be 0.7408947933449082 when the cut off is 0.7, this means the model correctly predicted the house price 74.09% of the times.

As you can see, the features used for the decision tree were derived from the features importance histogram I created.

```
# Evaluate the model MSE & MAE
mse = mean_squared_error(y_test, model_predict)
mae = mean_absolute_error(y_test, model_predict)
print("Mean Squared Error:", mse)
print("Mean Absolute Error:", mae)
```

```
Mean Squared Error: 1613953002.619512
Mean Absolute Error: 27377.39024390244
```

What are the limitations of the model been used?

Mean Square Error: In this case, an MSE of approximately 1.61 billion suggests that, on average, the squared difference between the predicted and actual sales prices is around 1.61 billion squared units of the target variable which is reasonable based on the features but not low enough as a lower MSE indicates that the model's predictions are closer to the actual values on average.

Mean Absolute Error: In this case, an MAE of approximately 27,377 suggests that, on average, the absolute difference between the predicted and actual sales prices is around $27,377 which is also reasonable considering the metrics used but a lower difference between the predicted sales prices and the actual sales prices is always better.

Data Integrity: Any model's ability to predict the future is dependent on the caliber of the training data. Predictions might be skewed by incomplete or inaccurate data.

What cognitive biases would you expect to influence the decision-making process? How does decision support mitigate some/all these?

Bias in Confirmation: Results that support stakeholders' (Brentford Real Estate) preconceived notions or expectations regarding house prices may be more favorable.

Overdependence on Historical Data: The algorithm may reinforce prejudices found in previous house prices if the historical data used to train it exhibits such biases.

Bias in Availability: Decision-makers may disregard less evident but significant aspects in favor of easily accessible data or trends.

Decision-Supported Mitigation: Reducing biases can be achieved by giving explicit explanations of the decision-making process models use. Pruning classification trees provides interpretability, which promotes trust.

What enhancements would you aim for to enable better decision support for this task?

Loop of Feedback for Ongoing Improvement: Establish a mechanism for obtaining input from house choices, features, and results to update and enhance the model over time.

Fairness Evaluations: To find and fix any biases in the model, especially with relation to the important features, do fairness assessments.