The University of Texas at Dallas - Naveen Jindal School of Management

Predictive Modeling of Housing Prices Using Key Property Variables

Group 2: Chris Espino, Zohair Shakir, Pavan Kantipudi, Harshita Paila, Hruthik Rudravaram, Ikramulla Shaik, Om Simha Reddy

BUAN 6390.002

Professor Jose Alvarez

11 November 2024

Table of contents

1. Introduction

1A. Context regarding housing prices in the U.S

According to a study conducted by McKinsey & Company, the increasing adoption of hybrid work models within the United States has had significant effects on residential properties, and will be here to stay in the near future. Although some believed that the hybrid work model was a response to the COVID-19 pandemic and was not permanent, analysis of residential prices suggest that hybrid work models are the new norm. The new norm of hybrid work models have resulted in:

- Recurring predictions that demand for offices will continue to decrease as hybrid
- Office-working Americans spending more time at home, leading to an increase in demand and prices for residential homes in residential/mixed-use neighborhoods
- Increased earning potential from residential homes for real estate companies/firms

To support the analysis conducted by Mckinsey & Company, we've found that Goldman Sachs has forecasted an increase in the average US home price. Goldman Sachs analysts have forecasted that home prices will appreciate 4.5% this year and 4.4% in 2025, which is a significant increase when compared to their earlier forecasts, which were 4.2% this year and 3.2% in 2025.

1B. Reasoning behind selection/problem statement

In response to our findings within the aforementioned publications of McKinsey & Company and Goldman Sachs, **we as a group were determined to play a part in the increased demand and prices for residential homes**. As a result, we decided that **the creation of a robust predictive model to predict the price of a home given its key characteristics would be a beneficial contribution to many key stakeholders**. We aimed to create this predictive model in hopes to benefit the following stakeholders:

1. Homebuyers
2. Government entities
3. Real estate agencies
4. Financial institutions

5. Property developers

For the previously-mentioned stakeholders, the intended benefits of our predictive model includes:

- Increased market transparency and stability
- Improved decision-making for buyers and sellers
- Minimized risk for financial institutions
- Support for affordable housing initiatives

2. Strategy

2A. High level strategy

In order to maintain organization and uniformity, we decided that creating a high-level strategy that contained key phases would serve well to keep us on track to complete our project within the timeframe of this course. Upon the creation of our group, we drafted the following 5 step plan:

1. Exploratory Data Analysis (EDA)

In this phase, each individual member of the group conducted an EDA to identify and better understand the dataset that was utilized to create this model. After each of our individual EDA was completed, we met and discussed each of our key finds to create an EDA report that best explained the ins and outs of our dataset.

2. Model creation

In this phase, we assigned the task of model creation to 4 members of the group who had the experience and technical ability to create predictive models. The key task assigned to each member was to create models that they believed would best predict the price of a home given its key characteristics, with no requirement on the model used or the amount of models created. The 4 members that participated in the model creation phase are as follows: Chris Espino, Zohair Shakir, Om Simha Reddy, and Ikramulla Shaik.

3. Group discussion

Following the model creation phase, we collectively met as a group to discuss our findings. The topics that were discussed included tools used, models used, any challenges faced, and key events.

4. Model selection and tuning

Once the findings were discussed as a group, we met again to make our final decision on which model we believed performed the best. After this, we performed model tuning to better refine the predictive capability of our selected model.

5. Documentation and recommendation

For the last step in our plan, we met as a group to document the progress of this project from start to finish. In totality, we completed the documentation for our final slide presentation, our final report, our submitted code, and our final project charter. Lastly, we documented further recommendations for our project, expanding further on our vision and providing examples of what our final intended product would appear like.

2B. Tools used

We utilized a variety of tools that played varying roles in the completion of our project. We separated the tools into two sections; technical tools and non-technical tools. Technical tools include tools that were used for our models, and non-technical tools include tools used for everything else. The tools that we used are as follows:

- **Non-Technical tools:** Microsoft Suite (Word, PowerPoint, Teams), Google Slides, Google Docs
- **Technical tools:** Python (pandas, NumPy, Matplotlib, Seaborn, Scikit-learn, XGBoost), Microsoft Excel

The following are tools that we originally intended to use, but ultimately decided against using them.

- **R:** We decided to use only Python for uniformity and ease of collaboration purposes
- **Tableau/PowerBI:** Due to Python having the necessary visualization functionalities for our project as these two tools, we decided against using them

3. Findings

3A. Dataset introduction

The dataset we used in this project is titled “Housing Prices Dataset”. This dataset was posted by user “M Yasser H” on the website “Kaggle.com”, which is a data science community hosting various datasets and data science discussions. Through our research, we could not locate a time from where this dataset is from, but a general location on the US east coast was provided, specifically the east coast states. The dataset contains 13 key features, which were of varying data types but were reformatted for usability with our Python libraries. The 13 key features are as shown below:

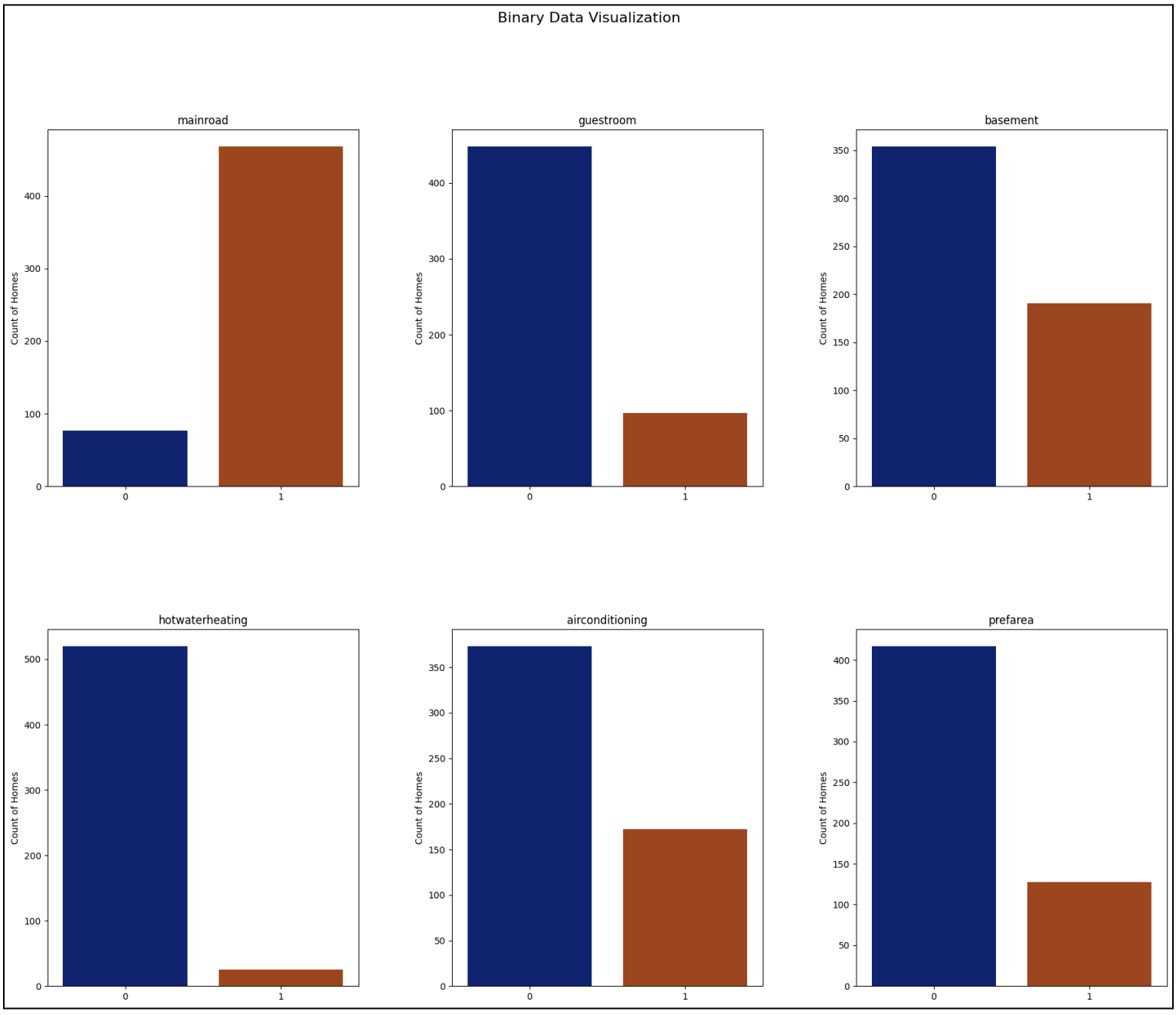
```
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   price             545 non-null    int64 
 1   area              545 non-null    int64 
 2   bedrooms          545 non-null    int64 
 3   bathrooms         545 non-null    int64 
 4   stories           545 non-null    int64 
 5   mainroad          545 non-null    int64 
 6   guestroom         545 non-null    int64 
 7   basement          545 non-null    int64 
 8   hotwaterheating   545 non-null    int64 
 9   airconditioning   545 non-null    int64 
 10  parking           545 non-null    int64 
 11  prefarea          545 non-null    int64 
 12  furnishingstatus  545 non-null    int64 
dtypes: int64(13)
memory usage: 55.5 KB
```

Since we will be using the other 12 columns to predict the price variable, the dependent variable is the price variable, and the other 12 variables are the independent variables. Key information and summary statistics of the 13 variables are as shown below:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
count	5.450000e+02	545.000000	545.000000	545.000000	545.000000	545.000000	545.000000	545.000000	545.000000	545.000000	545.000000	545.000000	545.000000
mean	4.766729e+06	5150.541284	2.965138	1.286239	1.805505	0.858716	0.177982	0.350459	0.045872	0.315596	0.693578	0.234862	0.930275
std	1.870440e+06	2170.141023	0.738064	0.502470	0.867492	0.348635	0.382849	0.477552	0.209399	0.465180	0.861586	0.424302	0.761373
min	1.750000e+06	1650.000000	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	3.430000e+06	3600.000000	2.000000	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	4.340000e+06	4600.000000	3.000000	1.000000	2.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
75%	5.740000e+06	6360.000000	3.000000	2.000000	2.000000	1.000000	0.000000	1.000000	0.000000	1.000000	1.000000	0.000000	2.000000
max	1.330000e+07	16200.000000	6.000000	4.000000	4.000000	1.000000	1.000000	1.000000	1.000000	1.000000	3.000000	1.000000	2.000000

3B. Exploratory Data Analysis (EDA)

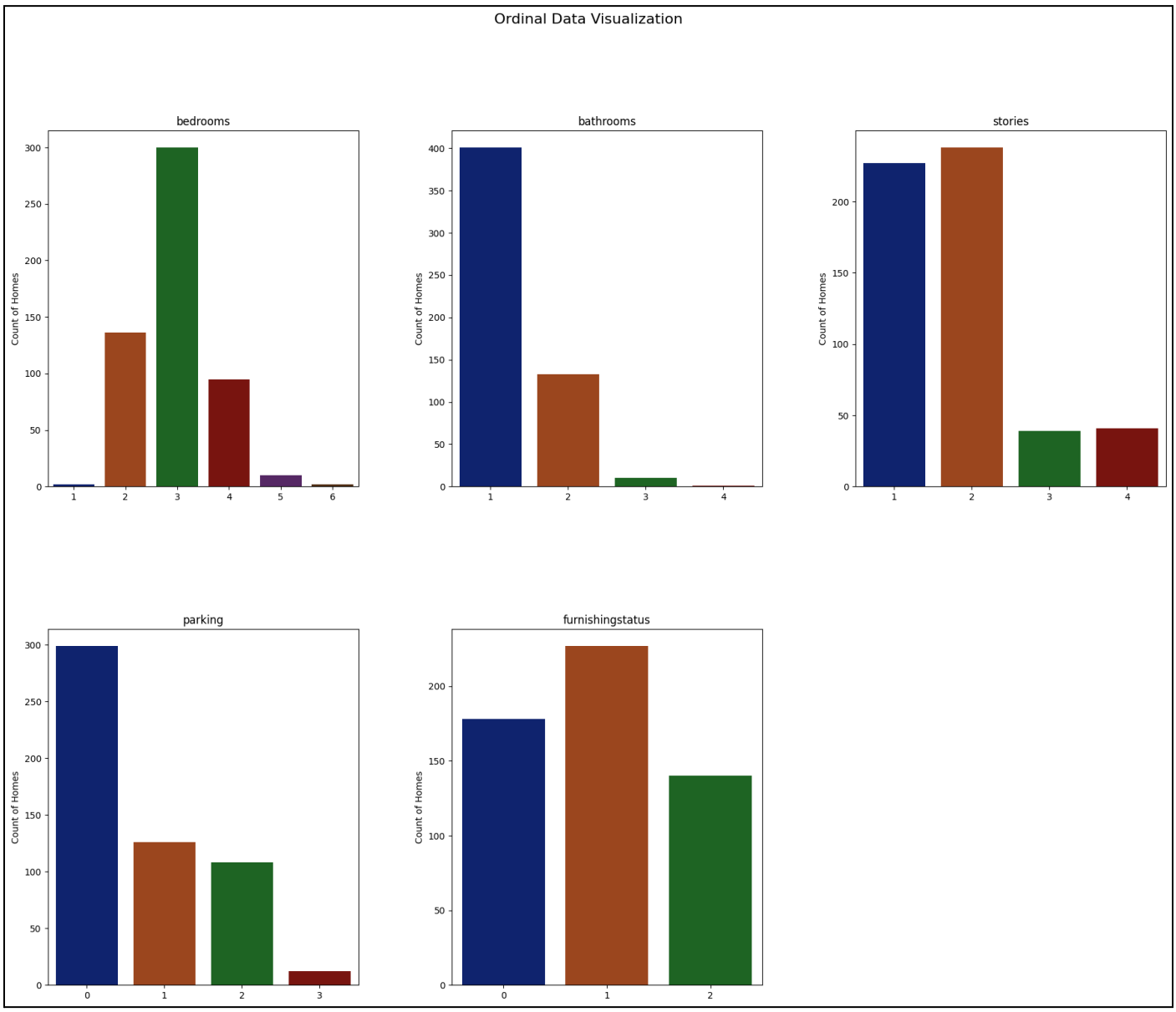
The majority of our coding work was done on Python. Prior to performing our EDA, we utilized Python and the Pandas library to clean and format our dataset. After this, we performed our EDA using the following libraries in Python: pandas, NumPy, Matplotlib, and Seaborn. The first visualization we will include is the binary data, which are columns with either yes or no values. Yes is encoded as 1, and no is encoded as 0.



Based on generated visualizations of these variables, we came to the following conclusions:

- Majority of the homes in this dataset do not have a basement or a guest room
- Majority of the homes lack appliances that we would generally consider necessities, such as hot water heating or air conditioning
- Majority of the homes are not in a preferred area
- Majority of the homes are connected to the main road

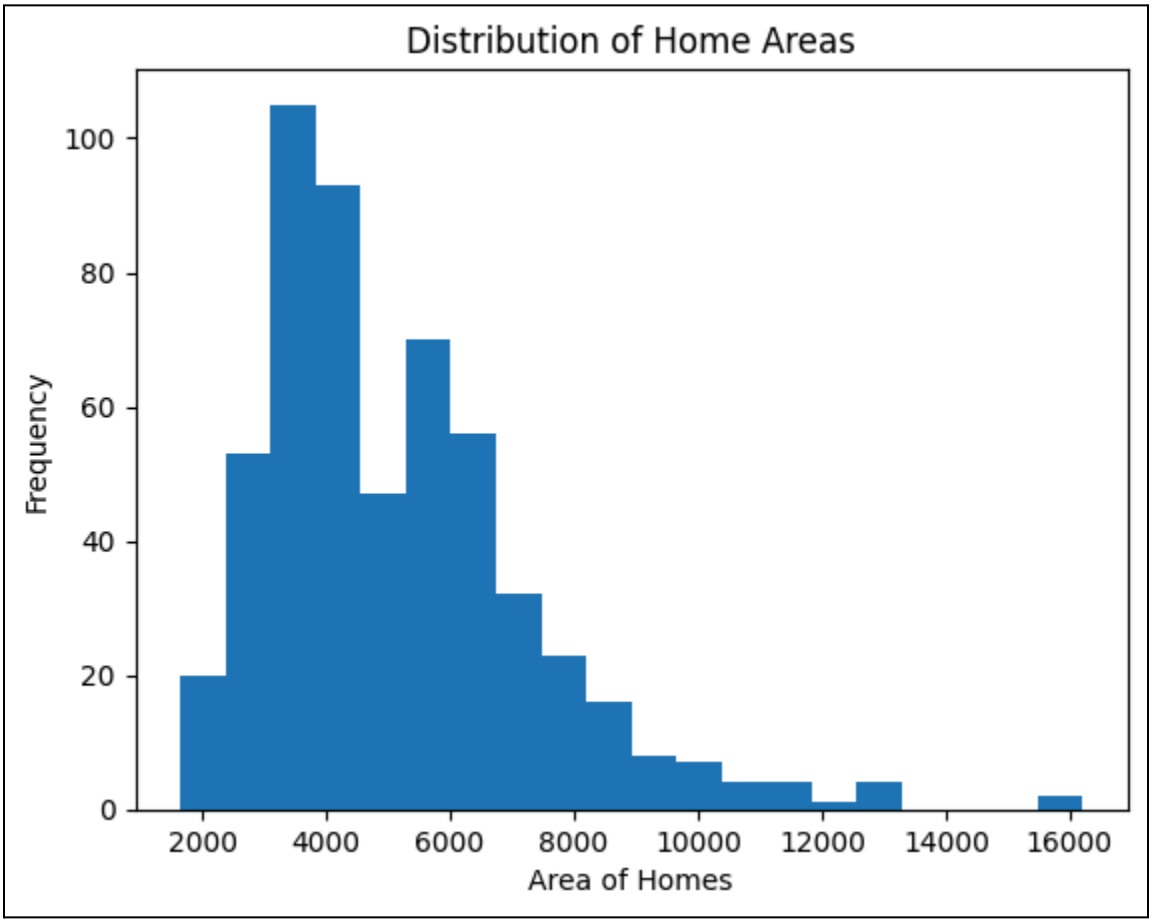
The next analysis we explored is the ordinal data. The ordinal data represents the amount present in said variable. For example, the bathrooms feature has values present from 1-4, where 1-4 represents the amount of bathrooms present within the home.



Based on the visualizations created for the ordinal features, we came to the following conclusions:

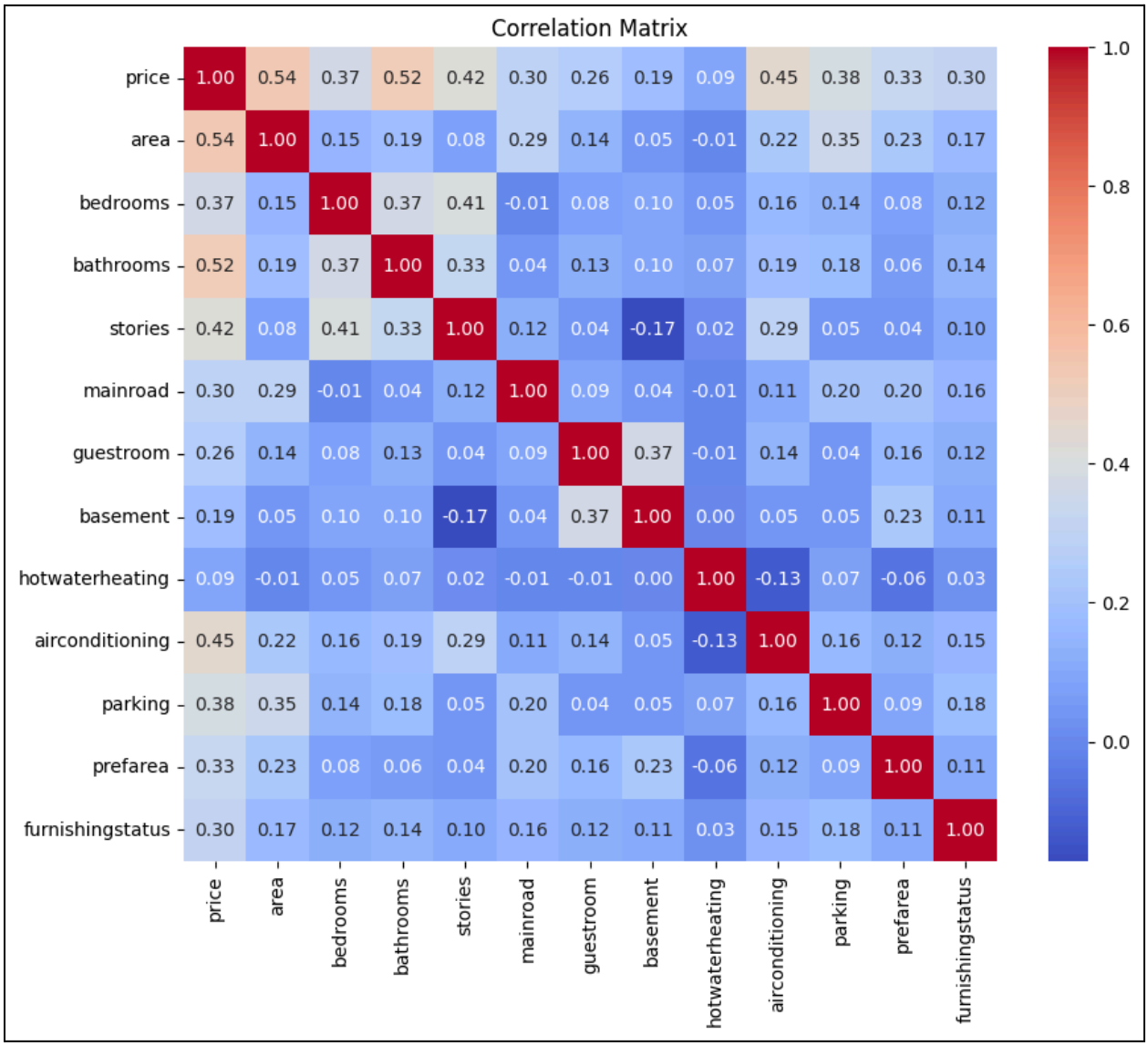
- Most of the homes have 3 bedrooms. Following this, the most common amount of bedrooms found within the homes are 2 and 4 bedrooms
- Majority of the homes have only 1 bathroom. Following this, the next most common amount of bathrooms is 2
- Majority of the homes are 2 or 1 story homes, with a couple having 3 or 4 stories
- Majority of the homes lack parking, while some have 1 or 2 spots of parking
- While most homes are semi-furnished, there is a good mix of homes that are unfurnished, semi-furnished, or furnished

The next analysis we're performing is of the "area" variable. As this variable is a quantitative variable, we've separated the analysis.



Based on the area of all the homes, we've found that the majority of the homes are in the region of ~4000 square feet. Past this size, the frequency of homes decreases.

The next, and final, EDA we're performing is of all 13 variables. We'll be using a correlation matrix to measure the correlation of all 13 variables with each other.



Using the above correlation matrix, we've taken note of the following:

- The correlation between the independent variables are mostly under 25%
- The correlation between price and the 12 independent variables are mostly above 25%, with the exception of the basement variable
- The variables area, bathrooms,and hotwaterheating seem to have the largest correlation with the price

In summary, our EDA revealed key insights regarding our selected dataset. Most homes lack certain amenities such as basements and guest rooms, and are not located in preferred areas. The analysis showed that area, bathrooms, and hot water heating have the strongest correlations with price, suggesting these features are significant in predicting housing prices. Overall, the dataset has minimal multicollinearity, with most independent variables showing low correlation with each other.

3C. Models created

Chris, Zohair, Om, and Ikramulla created a total of **13 models**. Although there was overlap in the models that were used by each person, the accuracy results of each model differed. During our discussions, we collectively agreed that we would use the R-squared value to measure the accuracy of our models. The models created and their performances are below:

Chris: Linear Regression, Lasso Regression, Ridge Regression, Lasso Regression with Grid Search and Cross Validation

Zohair: Linear Regression with Recursive Feature Elimination and Cross Validation

Om: Linear Regression, Decision Trees, Random Forest, Gradient Boost

Ikramulla: Linear Regression, Random Forest, XGBoost

3D. Performance of models

In this section, we will provide each of our created models and their performance. To measure the performance of our models, we collectively agreed to use the R-squared metric, aiming for an R-squared value of 85% or above.

- **Chris** (refer to file "BUAN 6390 - Chris Code").

  a. **Linear Regression:** 62.88% R-squared

```
# predicted value of test set
lr_test_pred = lr.predict(X_test)

# rmse
test_resid = y_test - lr_test_pred
print("Testing set RMSE:", np.mean(test_resid**2)**(1/2))

# mae
print("Testing set MAE:", np.mean(np.abs(test_resid)))

# r-squared value
print("Testing set R-squared:", lr.score(X_test, y_test))
```

```
Testing set RMSE: 1169463.0598567484
Testing set MAE: 840447.3526790561
Testing set R-squared: 0.6287638839312104
```

  b. **Ridge Regression:** Two models were created, one with **alpha=1**, and **alpha=100**:

     i. **Alpha = 1**: 62.75% R-squared

     ii. **Alpha = 100**: 54.01% R-squared

```
Ridge Regression Model with alpha=1 Performance Metrics
RMSE: 1171472.1362494333
MAE: 839859.8299244401
R-squared: 0.6274872598101641

Ridge Regression Model with alpha=100 Performance Metrics
RMSE: 1301671.5209983727
MAE: 904657.3050266728
R-squared: 0.5400824202601554
```

  c. **Lasso Regression:** Two models were created, one with **alpha=100,** and **alpha=1000:**

     i. **alpha=100:** 62.87% R-squared

     ii. **alpha=1000:** 62.79% R-squared

```
Lasso Regression Model with alpha=100 Performance Metrics
RMSE: 1169603.9726592435
MAE: 840441.7638339244
R-squared: 0.6286744153965912

Lasso Regression Model with alpha=1000 Performance Metrics
RMSE: 1170895.9457709198
MAE: 840391.4877088971
R-squared: 0.6278536116887705
```

d. Lasso Regression with Grid Search and Cross Validation: 62.88% R-squared

```python
from sklearn.linear_model import Lasso
from sklearn.model_selection import GridSearchCV

# define range of alphas that will be tested
alpha_range = {'alpha': [0.001, 0.01, 0.1, 1, 10, 100]}

# re-initialize the Lasso model
lasso = Lasso()

# set up Grid Search with Cross Validation (5 folds)
grid_search = GridSearchCV(estimator=lasso, param_grid=alpha_range, cv=5, scoring='r2')

# fit the model to the data
grid_search.fit(X_train, y_train)

# Fit the model on the training data and evaluate R^2
best_alpha = grid_search.best_params_['alpha']
best_model = grid_search.best_estimator_

# Evaluate R^2 on training data
print(f"Best model R^2 score on training set: {best_model.score(X_train, y_train):.4f}")

# Predict and evaluate on testing data
y_pred = best_model.predict(X_test)
test_r2 = best_model.score(X_test, y_test)
print(f"Best model R^2 score on testing set: {test_r2:.4f}")

```

```
Best model R^2 score on training set: 0.6944
Best model R^2 score on testing set: 0.6288
```

- **Zohair**

Fitting the initial model:

```
[102]: X, y=df1.drop(['price',], axis=1), df1['price']

[103]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state=1)

[104]: model = LinearRegression()

       model.fit(X_train,y_train)

[104]: ▾ LinearRegression
       LinearRegression()

[105]: model.score(X_test,y_test)

[105]: 0.6221280430074472
```

Next, RFE was used to get a ranking of parameters. This was done so that we had an understanding of which variables had the highest importance according to this algorithm.

```
rfe = RFE(estimator=model, n_features_to_select=1)
rfe.fit(X_train, y_train)


feature_ranking = pd.DataFrame({'Feature': X_train.columns, 'Ranking': rfe.ranking_})
print(feature_ranking.sort_values(by='Ranking'))
```

	Feature	Ranking
2	bathrooms	1
4	mainroad	2
8	airconditioning	3
7	hotwaterheating	4
10	prefarea	5
5	guestroom	6
9	parking	7
3	stories	8
11	unfurnished	9
6	basement	10
1	bedrooms	11
13	furnished	12
12	semi-furnished	13
0	area	14

Lastly, RFECV (Recursive Feature Elimination with Cross Validation) was used to detect and eliminate any unnecessary features. However, the algorithm determined that all features were relevant, and so none were dropped. The result was the same as the initial model.

```
[107]: from sklearn.feature_selection import RFECV

[108]: selector = RFECV(model, scoring = 'r2', cv=5)
       selector = selector.fit(X_train, y_train)

[109]: print("Optimal number of features: %d" % selector.n_features_)
       print("Selected features: %s" % selector.support_)

       Optimal number of features: 14
       Selected features: [ True  True  True  True  True  True  True  True  True  True  True  True
         True  True]

[110]: selector.score(X_test,y_test)

[110]: 0.6221280430074472
```

- **Ikramulla**

  e. **Linear Regression:** 64.95% R-squared

```
# Encode binary columns
binary_columns = ['mainroad', 'guestroom', 'basement', 'hotwaterheating', 'airconditioning', 'prefarea']
for col in binary_columns:
    housing_dataset[col] = housing_dataset[col].map({'yes': 1, 'no': 0})

# Ordinally encode 'furnishingstatus'
housing_dataset['furnishingstatus'] = housing_dataset['furnishingstatus'].map({'unfurnished': 0, 'semi-furnished': 1, 'furnished': 2})
# Separate features and target
X = housing_dataset.drop(columns=['price'])
y = housing_dataset['price']
# 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)

# Initialize and train the model
linear_model = LinearRegression()
linear_model.fit(X_train, y_train)

# Predict and evaluate
y_pred = linear_model.predict(X_test)
r2 = r2_score(y_test, y_pred)
# Display metrics
print("Linear Regression Results:")
print(f"R-squared: {r2:.2f}")

# Print R-squared as a percentage to represent model accuracy
accuracy_percentage = r2 * 100
print(f"Model Accuracy (R-squared as percentage): {accuracy_percentage:.2f}%")
```

```
Linear Regression Results:
R-squared: 0.65
Model Accuracy (R-squared as percentage): 64.95%
```

f. Random Forest: 41.60% R-squared

```
# Separate features and target
X = housing_dataset.drop(columns=['price'])
y = housing_dataset['price']
# Split the data into training and testing sets
# 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)

# Initialize and train the model
forest_model = RandomForestRegressor(random_state=42)
forest_model.fit(X_train, y_train)

# Predict and evaluate
y_pred = forest_model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Display metrics
print("Random Forest Regressor Results:")
print(f"R-squared: {r2:.2f}")
print(f"Model Accuracy (R-squared as percentage): {accuracy_percentage:.2f}%")
```

```
Random Forest Regressor Results:
R-squared: 0.61
Model Accuracy (R-squared as percentage): 41.60%
```

C. XGBoost: 59.79% R-squared

```
# Separate features and target
X = housing_dataset.drop(columns=['price'])
y = housing_dataset['price']

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train the XGBoost regressor
xgboost_model = XGBRegressor(objective='reg:squarederror', random_state=42)
xgboost_model.fit(X_train, y_train)

# Predict on the test set and calculate R-squared
y_pred = xgboost_model.predict(X_test)
r2 = r2_score(y_test, y_pred)

# Print the R-squared and model accuracy
print("XGBoost:")
print(f"R-squared: {r2:.2f}")
print(f"Model Accuracy (R-squared as percentage): {r2 * 100:.2f}%")
```

```
XGBoost:
R-squared: 0.60
Model Accuracy (R-squared as percentage): 59.79%
```

- ## Om: Feature Engineering and New Feature Creation

  - **Objective of Feature Engineering**
  - Feature engineering was undertaken to enhance the model's predictive performance by creating additional derived features that capture key relationships in the data. These features were designed based on domain knowledge and exploratory data analysis.
  - **New Features Created**
    - Total Rooms
      - Definition: The sum of the number of bedrooms and bathrooms in a house.
      - Reason for Selection:
        - This feature captures the overall size and usability of a house in terms of living space, which directly influences housing prices.
        - A combined metric reduces potential noise from individual features (bedrooms and bathrooms) and emphasizes their combined effect on price.
      - Calculation: total_rooms = bedrooms+bathrooms
    - Price per Square Foot
      - Definition: The price of the house divided by its area.
      - Reason for Selection:
        - This feature normalizes the price based on property size, allowing the model to account for variations in house sizes and providing a standardized measure of property value.
        - It helps identify outliers and areas where price deviations exist per unit area.
      - Calculation: price_per_sqft = price / area

```
# Create a total_rooms feature
housing_dataset['total_rooms'] = housing_dataset['bedrooms'] + housing_dataset['bathrooms']

# Create a price per square foot feature
housing_dataset['price_per_sqft'] = housing_dataset['price'] / housing_dataset['area']

# Display the updated DataFrame with new features
print(housing_dataset[['price', 'area', 'bedrooms', 'bathrooms', 'total_rooms', 'price_per_sqft']].head())
```

Model Comparison and Performance Evaluation

```
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
def train_and_evaluate_model(model, X_train, y_train, X_test, y_test):
    # Train the model
    model.fit(X_train, y_train)

    # Make predictions
    y_pred = model.predict(X_test)

    # Calculate evaluation metrics
    mae = mean_absolute_error(y_test, y_pred)
    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)

    return mae, mse, r2
# Initialize models
linear_model = LinearRegression()
decision_tree_model = DecisionTreeRegressor(random_state=42)
random_forest_model = RandomForestRegressor(random_state=42)
gbm_model = GradientBoostingRegressor(random_state=42)

# Evaluate each model
models = {
    "Linear Regression": linear_model,
    "Decision Tree": decision_tree_model,
    "Random Forest": random_forest_model,
    "Gradient Boosting": gbm_model
}

# Store results
results = {}

for model_name, model in models.items():
    mae, mse, r2 = train_and_evaluate_model(model, X_train, y_train, X_test, y_test)
    results[model_name] = {
        "MAE": mae,
        "MSE": mse,
        "R²": r2
    }

# Display the results
results_df = pd.DataFrame(results).T
print(results_df)
```

	MAE	MSE	R²
Linear Regression	515486.980078	6.295328e+11	0.875453
Decision Tree	384748.256881	4.153628e+11	0.917824
Random Forest	289505.544037	3.442401e+11	0.931895
Gradient Boosting	268800.907748	1.882740e+11	0.962752

Successful models/selected model: We believe that Om's Gradient Boost regression model is the most successful model, due to his model having the best R-squared value out of all the models selected. In addition, we also believe that the other models created by Om are also successful, due to all of Om's models having an R-squared value higher than 85%.

Unsuccessful models: We've identified all of the models with an R-squared value lower than 85% as our unsuccessful models. Although R-squared values in the range of ~60% does not represent a bad model, it shows that improvement is necessary to deem it reliable, therefore these models are not deemed successful. Therefore, all models created by Chris, Zohair, and Ikramulla have been deemed unsuccessful.

3E. Revisiting the problem statement

By creating the Gradient Boost regression model with an R-squared value of 96.28%, we believe that we successfully fulfilled our project statement, which was to create a robust model with accurate prediction capabilities of a home's price using its key characteristics.

4. Challenges

4A. Technical Challenges

Throughout the course of this project, we encountered minimal technical challenges. Although each member of the group had varying levels of technical proficiency, we assigned tasks to maximize the strengths of each member of the group. Having said that, we encountered 3 challenges that affected both our collective productivity and model performances:

1. **Proper version control practices.** Throughout the EDA and model creation processes, we occasionally faced instances where we accidentally deleted parts of our code that were important to the model we created. Although we were able to figure out and re-code what we deleted each instance, the 15-30 minutes spent each instance negatively affected our productivity.

2. **Jupyter Notebook versus Google Colab.** During the portions of the project where coding was necessary, the majority of team members used Jupyter Notebook, which has no cloud syncing capabilities. As a result, we were limited to either watching one member code one notebook file, or having each member of the group make contributions to a local notebook file that would be later consolidated into one notebook file. Additionally, there were some packages that were incompatible between platforms, requiring users to rewrite code before using it on another platform.
3. **Technical limitations affecting model performance.** The best performing models for Chris, Zohair, and Ikramulla were in the range of 60-65%, which we were not content with. Despite this, they could not figure out how to increase the performance of their respective models. Although this was the case, Om's technical proficiency allowed him to create a model with an R-squared value of ~96%, which we deemed sufficient.
4. **Dataset limitations.** We were not able to find any references regarding the date/time that this dataset references. We believe that this would've been a useful context to better understand the prices of the homes, and would've also allowed us to perform comparisons of other housing data from the same time frame. In addition, this dataset was smaller than we preferred, with only 545 observations. According to data provided by Point2homes, Frisco, Texas has roughly ~70,000 homes. Comparing our dataset to the amount of homes there are in Frisco, we can safely infer that a sample of 545 homes is comparable to the amount of homes in a suburban subdivision.

4B. Non-Technical Challenges

As a group, we encountered non-technical challenges that were remediated through the course of the project. The challenges are as follows:

1. **Scheduling constraints.** Due to all members of the group having obligations in other courses and outside of school, there were instances where we struggled to schedule both vital and non-vital meetings. Despite this, we were able to figure out scheduling practices that worked with all members.

2. **Delayed deadlines.** As a result of failing to meet our self-assigned metrics in the model creation phase, we had to delay our deadlines. As a result, we had to move several meetings to accommodate the delayed deadlines. Despite this, we were able to complete our project within our intended timeline.
3. **No budget available.** As a group, we believed that having a budget available could've increased the quality of our project in two manners: data quality and tools used. Regarding data quality, we believe that having a budget would have allowed us to pay for a higher quality dataset that could have led to a more precise model. In addition, having a budget would have allowed us to purchase beneficial tools that provide more data science capabilities.

5. Conclusion and Recommendation

5A. Conclusion

As a group, we collectively learned more about the role that a home's features play in predicting its price, while also learning more about various data analysis & science techniques that were necessary in the completion of our project. Although we faced minor technical and non-technical challenges, we were able to overcome them and learn from them to increase our individual skill sets. In conclusion, our group fulfilled our project statement by creating a robust, precise model that predicts the price of a home given its key characteristics.

5B. Intended Final Product and Recommendations

Overall, we believe that we efficiently and effectively used the resources at our disposal to create an accurate and dependable model to predict housing prices. Despite this, we believe that the addition of the 3 following suggestions will create our true intended final product. We've noted our 3 suggestions below:

1. **Model implementation info a fully-functional application ecosystem.** We believe that integrating our model into an application with a vast, supportive ecosystem will allow the model to serve as the primary product, with supporting features that enhance its value. By doing so, we believe that users will feel more incentive to use our model and its added features. In turn, the application can be turned into a pay-to-use, subscription-based model, where customers pay on a timed basis to use the application.
2. **Marketing and partnerships.** Although we have belief in our product, we aim to also instill the same beliefs in our potential customers. This can be done through marketing and partnership efforts, which can serve to expand our applications reach and revenue streams by establishing strategic partnerships with both private and non-private entities. While we have mentioned usage of a subscription-based model, we can also utilize our marketing and partnership efforts to secure contracts with larger, more established entities. Through usage of the contract model, we can lock in large customers with lengthy, profitable contracts to ensure mid and long-term revenue.
3. **Legal and compliance regulations.** Abiding with legal and compliance regulations will significantly benefit our product in the long run. We can establish credibility and trust with both private and non-private customers, while also avoiding expensive legal fines and penalties. In addition, having assurance that our product/company is abiding by the law can allow us to expand in highly-regulated markets that our competitors may not have access to.

6. Project Progression Log

6A. Group Key Discussions

The group will establish communication with the Professor as necessary to ensure alignment and address any relevant updates or guidance. Any new or revised information will be instantly communicated to the team via the designated team chat platform, and reinforced during our weekly interaction meetings. To maintain transparency and ensure project progress, each group member will deliver detailed status reports to the team periodically. We've decided to use a hybrid work approach, combining campus facilities for in-person contacts with Microsoft Teams for virtual meetings, to provide flexibility and effective cooperation throughout the project.

6B. Meetings Summary

Our group project is structured around a series of key milestones and deadlines. We began with the planning phase on September 9, 2024, followed by a project alignment session on September 27, 2024. The dataset was finalized by September 30, 2024, and the first dataset analysis check-in occurred on October 4, 2024. A second check-in for dataset analysis was held on October 11, 2024, leading to the finalization of the dataset approach by October 18, 2024. The initial model construction check-in took place on October 25, 2024, followed by the model construction finalization on November 1, 2024. We then moved into the reporting phase, with a brainstorming session on November 9, 2024, and final reporting completed by November 16, 2024. After a meeting with Professor on November 18, 2024, we incorporated the changes suggested to add the final refinements to our project.

7. References

Sanghvi, Aditya, and Lola Woetzel. “The Future of Real Estate in a Hybrid World.” *McKinsey & Company*, McKinsey & Company, 19 Oct. 2023, www.mckinsey.com/mgi/our-research/the-future-of-real-estate-in-a-hybrid-world.

“US House Prices Are Forecast to Rise More than 4% next Year.” *Goldman Sachs*, 11 Sept. 2024, www.goldmansachs.com/insights/articles/us-house-prices-are-forecast-to-rise-more-than-4-percent-next-year?chl=ps&plt=go&cid=20314087364&agp=156316993088&kid=housing+market&mtype=p&gad_source=1&gclid=Cj0KCQiAlsy5BhDeARIsABRc6ZtAltMuzikth5Bg9bmEnLlRh-QT6PzC2U0_chKkD9iFVUcx_4BGTMYaAiQxEALw_wcB&gclsrc=aw.ds.

Dscode. “Mastering Mobile App UI Design: A Beginner’s Guide.” *Medium*, Medium, 9 Apr. 2024, medium.com/@dscode/mastering-mobile-app-ui-design-a-beginners-guide-2080a01c9df1.

https://www.kaggle.com/datasets/yasserh/housing-prices-dataset

https://www.point2homes.com/US/Neighborhood/TX/Frisco-Demographics.html#:~:text=There%20are%2074%2C212%20housing%20units,have%20renters%20living%20in%20them.