Research Question

Home prices in the United States have risen substantially over the past two years. According to Reuters, the average home price has been up 17% over the last year and is expected to rise another 10.3%. (Kishan & Ganguly, 2022) The causes of this dramatic rise in prices include low-interest rates, low inventory of homes for sale, a soaring stock market, and covid. People have more cash on hand because of the lockdown. These influences have led to multiple offers above the asking price for all properties on the market. (Orton, 2022)

Knowing which features of the home are most statistically significant and influence the price would benefit investors and individuals looking to buy a home. Investors will want to maximize their return, and individuals will want to get the features they want at the best price. This study will use multiple linear regression to explore the home's features and find if these features affect the price.

The research question for this study is: Does ward affect the price of a home in Washington D.C.? What other features affect the price? The null hypothesis states that Ward and other features do not statistically significantly affect the price of a home when the alpha level is 0.05. The alternative hypothesis is that Ward and other features statistically significantly affect the home price when the alpha level is 0.05.

Data Collection

The data set for this study was downloaded from D.C. Residential Properties on Kaggle. (Chrisc, 2018) It contains 159,000 records of homes in Washington, D.C. The data set is publicly available with an Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) license. The data set is comprised of 49 columns, 22 of which were used in this study. The columns used were:

Column Name	**Type**	**Description of Data**
BATHRM	Numeric	Number of Bathrooms
HF_BATHRM	Numeric	Number of Half Bathrooms
HEAT	Categorical	Heating Type
AC	Boolean	Air Conditioning Y/N
NUM_UNITS	Numeric	Number of Units

ROOMS	Numeric	Number of Rooms
BEDRM	Numeric	Number of Bedrooms
AYB	Numeric	When main portion of house was built
EYB	Numeric	Year of last improvement
STORIES	Numeric	Number of Stories
GBA	Numeric	Gross building area in square feet
STYLE	Categorical	Style Description
STRUCT	Categorical	Structure Description
GRADE	Categorical	Grade Description
CNDTN	Categorical	Condition Description
EXTWALL	Categorical	Exterior Wall Description
ROOF	Categorical	Roof Type Description
INTWALL	Categorical	Interior Wall Description
KITCHENS	Categorical	Number of Kitchens
FIREPLACES	Categorical	Number of Fireplaces
LANDAREA	Categorical	Land Area of Property in square feet
WARD	Categorical	City Ward

One advantage of this data-gathering methodology is its accessibility. The data set is available for free online at Kaggle.com. It has a high useability rating from Kaggle and includes many different features for homes.

A disadvantage of this data set was that many columns have a large percentage of null values. The target variable, Price, consists of 39% nulls.

Data Extraction and Preparation

Python was used to extract and prepare for analysis. Python is an easy-to-use, open-source language that offers an extensive library of external packages to expand Python to fit the user's needs. Pandas and NumPy were used to import, clean, and transform the data set. Seaborn and Matplotlib were used to create visualizations of the data. Python does have some disadvantages. Because Python is an interpreted and dynamically typed language, it can execute code slower than other languages. Python also uses more memory. (Python Advantages and Disadvantages - Step in the right direction, n.d.)

The data was loaded into a Jupyter notebook for cleaning and analysis. A new DataFrame was created with the columns being used in the study. The following steps were used to prepare the data:

- Since PRICE is the target variable, all rows that do not have a price are removed.

```
# Remove all Rows where PRICE is null

studydf = studydf[studydf['PRICE'].isnull() == False]
studydf.shape
```

```
(98216, 23)
```

- The word ward was removed from all entries in the WARD column, and the data type was changed to integer. (Code From: https://stackoverflow.com/questions/51778480/remove-certain-string-from-entire-column-in-pandas-dataframe)

```
studydf['WARD'] = studydf['WARD'].str.replace('Ward', '')
studydf['WARD'] = studydf['WARD'].astype(str).astype(int)

studydf['WARD'].head()
```

- The percent of nulls for each column was calculated. There were 11 columns with 25% of the data missing. The null values need to be filled for the columns to be kept. There are many ways to handle this. I choose a combination of imputing the mode or median and creating a 'Missing' classification.

```
per_missing = studydf.isnull().sum()*100/len(df.iloc[:,1])
per_missing
```

- The mode was used to replace the missing data for NUM_UNITS, STORIES, EXTWALL, INTWALL, and KITCHENS. The mode was used because, in each case, the mode occurred much more frequently than any other. For example, NUM_UNITS had 49339 rows with one as its value. The next closest was 2 with 6050.

```
studydf['NUM_UNITS'] = studydf['NUM_UNITS'].fillna(1)
studydf['NUM_UNITS'].value_counts()
```

```
studydf['STORIES'] = studydf['STORIES'].fillna(2)
studydf['STORIES'].value_counts()
```

```
studydf['EXTWALL'] = studydf['EXTWALL'].fillna('Common Brick')
studydf['EXTWALL'].value_counts()
```

```
studydf['INTWALL'] = studydf['INTWALL'].fillna('Hardwood')
studydf['INTWALL'].value_counts()
```

```
studydf['KITCHENS'] = studydf['KITCHENS'].fillna(1)
studydf['KITCHENS'].value_counts()
```

- The values for GBA are skewed to the right. The median was used to fill the null values.

```
GBAmedian = studydf['GBA'].median()
studydf['GBA'] = studydf['GBA'].fillna(GBAmedian)
```

- STRUCT, GRADE, CNDTN, and ROOF did not have a classification that had more than the others. A 'Missing' class was used to fill the null values.

```
studydf['STRUCT'] = studydf['STRUCT'].fillna('Missing')
studydf['STRUCT'].value_counts()
```

```
studydf['GRADE'] = studydf['GRADE'].fillna('Missing')
studydf['GRADE'].value_counts()
```

```
studydf['CNDTN'] = studydf['CNDTN'].fillna('Missing')
studydf['CNDTN'].value_counts()
```

```
studydf['ROOF'] = studydf['ROOF'].fillna('Missing')
studydf['ROOF'].value_counts()
```

- STYLE is redundant to STORIES. This column was dropped.

```
studydf = studydf.drop('STYLE', 1)
```

- AYB (year built) had 112 rows with null values. These rows were removed.

```
studydf['AYB'].isnull().sum()
```

- AYB and EYB (year of improvement) are float and int types, respectively. These features were changed to DateTime with the year only. (Code from: https://stackoverflow.com/questions/66468687/how-can-i-convert-float-column-to-datetime)

```
studydf['EYB']=pd.to_datetime(studydf['EYB'], format='%Y').dt.year
studydf['AYB']=pd.to_datetime(np.int16(studydf['AYB']), format='%Y')
studydf['AYB']=studydf['AYB'].dt.year
studydf.head()
```

- The summary statistics and boxplots were used to find outliers in the features. Several of the features have values that are not realistic. For example, the maximum value of

STORIES is 826. It is not realistic for a house to have that many stories. After inspecting the boxplots, the values that were above average were removed.

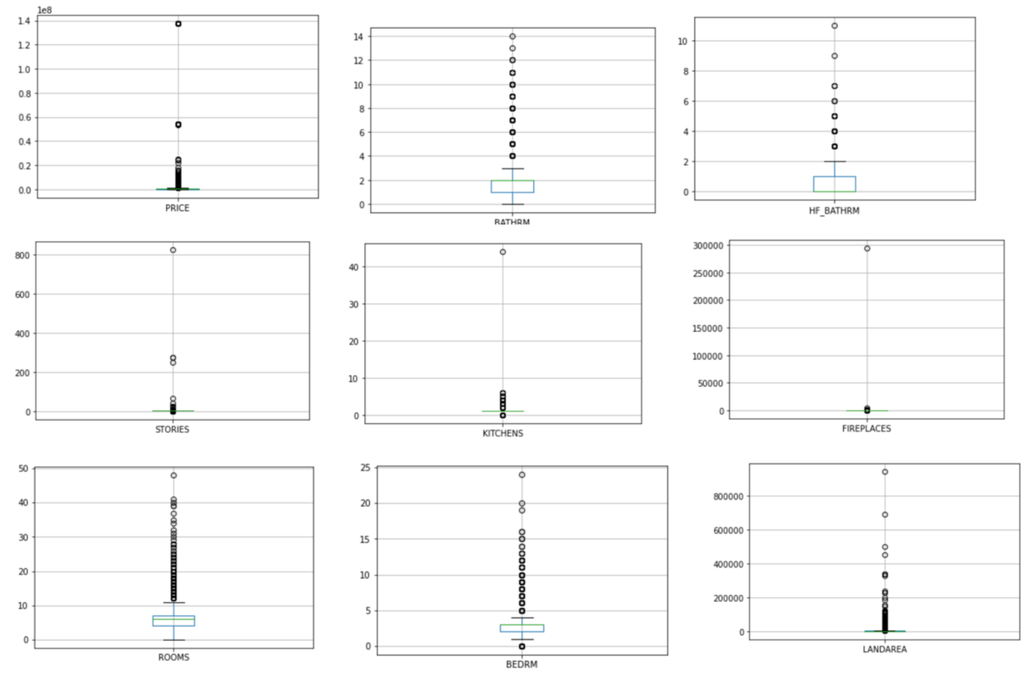
```
: studydf.describe()
```

```
:
             PRICE        BATHRM     HF_BATHRM     NUM_UNITS         ROOMS         BEDRM           AYB           EYB
count 9.810400e+04  98104.000000  98104.000000  98104.000000  98104.000000  98104.000000  98104.000000  98104.000000  98
 mean 9.317931e+05      1.859099      0.444396      1.130290      5.920544      2.605928   1944.903888   1965.771722
  std 7.065297e+06      0.978038      0.576163      0.490895      2.637495      1.393758     36.254098     27.504401
  min 1.000000e+00      0.000000      0.000000      0.000000      0.000000      0.000000   1754.000000   1800.000000
  25% 2.400000e+05      1.000000      0.000000      1.000000      4.000000      2.000000   1917.000000   1954.000000
  50% 3.999990e+05      2.000000      0.000000      1.000000      6.000000      3.000000   1939.000000   1965.000000
  75% 6.520000e+05      2.000000      1.000000      1.000000      7.000000      3.000000   1966.000000   1981.000000
  max 1.374275e+08     12.000000     11.000000      6.000000     31.000000     20.000000   2018.000000   2018.000000   8
```

(Code From: https://stackoverflow.com/questions/51777217/how-to-plot-a-boxplot-for-each-column-in-a-dataframe)

```python
box_plots=studydf[['PRICE','BATHRM', 'HF_BATHRM','STORIES', 'KITCHENS', \
                   'FIREPLACES','ROOMS', 'BEDRM', 'LANDAREA']]

for column in box_plots:
    plt.figure()
    df.boxplot([column])
```



```
studydf = studydf[studydf['PRICE'] < 1000000]
studydf = studydf[studydf['BATHRM'] < 6]
studydf = studydf[studydf['HF_BATHRM'] < 4]
studydf = studydf[studydf['STORIES'] < 4]
studydf = studydf[studydf['KITCHENS'] < 10]
studydf = studydf[studydf['FIREPLACES'] < 10]
studydf = studydf[studydf['PRICE'] < 6000000]
studydf = studydf[studydf['ROOMS'] < 15]
studydf = studydf[studydf['BEDRM'] < 10]
studydf = studydf[studydf['LANDAREA'] < 200000]
studydf.describe()
```

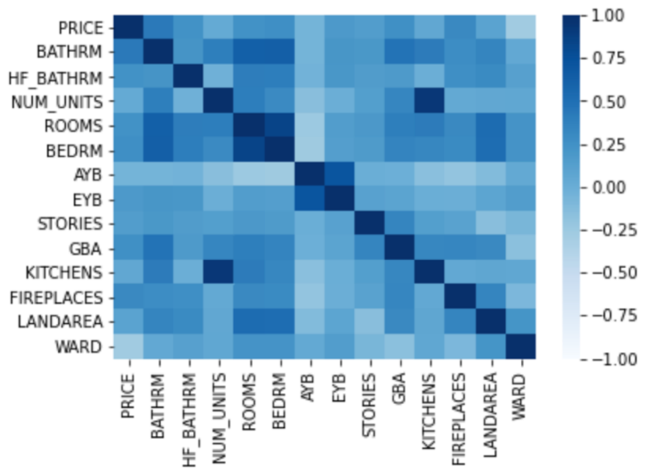
- Duplicates were dropped.

```
studydf = studydf.drop_duplicates()
studydf.shape
```

- A heatmap was created to examine the correlation between features. Kitchen and the number of units are strongly correlated. Rooms/Bedrooms and AYB/EYB have a high correlation.

```
sns.heatmap(studydf.corr(), vmin = -1, vmax = 1, cmap = 'Blues')
```

```
<AxesSubplot:>
```



- Dummy variables were created for categorical features.

```
study_df = pd.get_dummies(studydf, prefix={'HEAT':'HEAT','AC':'AC','STRUCT':'STRUCT',\
                                           'GRADE':'GRADE','CNDTN':'CNDTN', \
                                            'EXTWALL':'EXTWALL', 'ROOF':'ROOF', \
                                           'INTWALL':'INTWALL'}, \
                          drop_first=True)
study_df.head()
```

- The prepared data set was saved as a csv file.

Analysis

In this study, Ordinary Least Squares (OLS) regression was used to find the relationship between Price and the predictive features. OLS does this by finding coefficients for the predictive features that minimize the sum of the squares of the differences between the observed Price and the predictive Price. (Sharma, 2020) The step to perform an OLS regression are as follows:

- Split the data set into Price and the predictive features.

```
y = study_df['PRICE']
X = study_df.drop('PRICE', axis = 1)
```

- Add a constant term. Statsmodels does not include a constant and must be added. This constant functions as the intercept in the model.

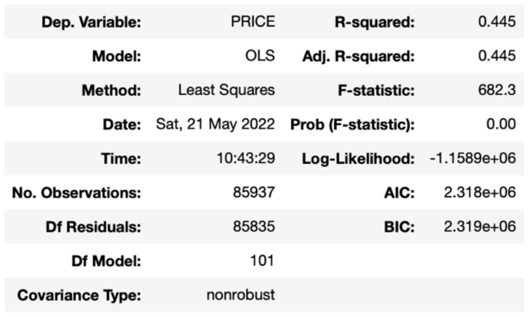
```
X = sm.add_constant(X)
```

- Train and fit the model.

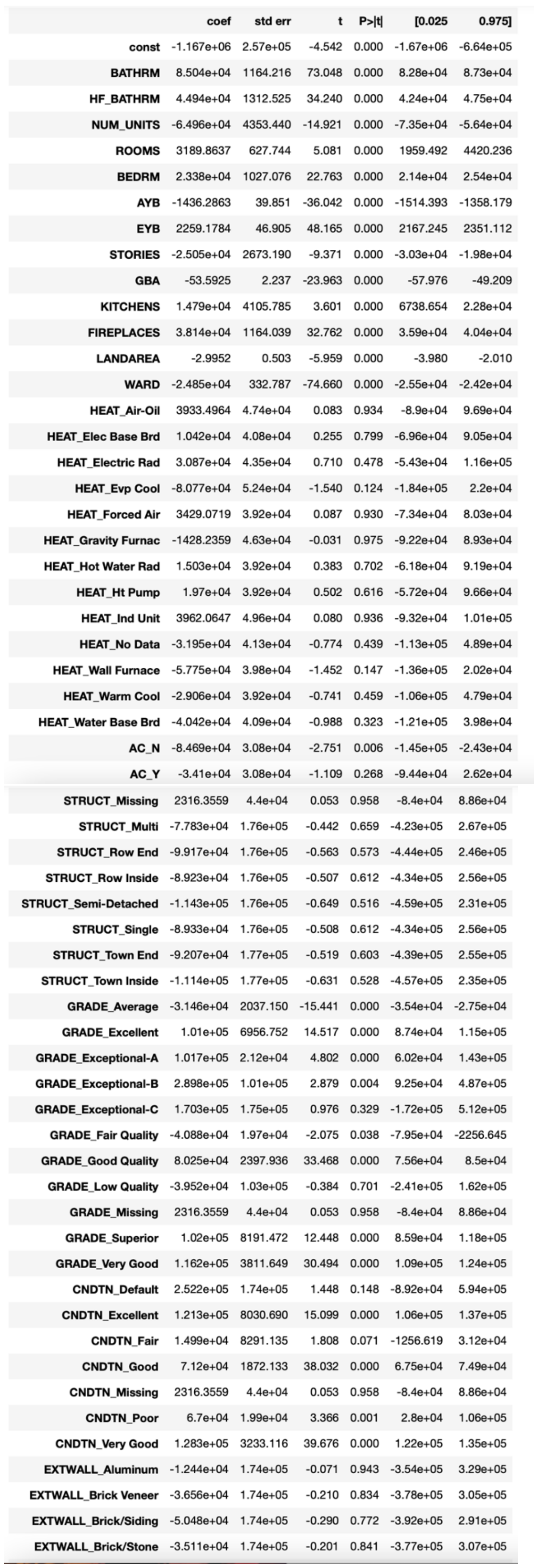
```
model = sm.OLS(y, X)
results = model.fit()
```

- Run a summary of the results

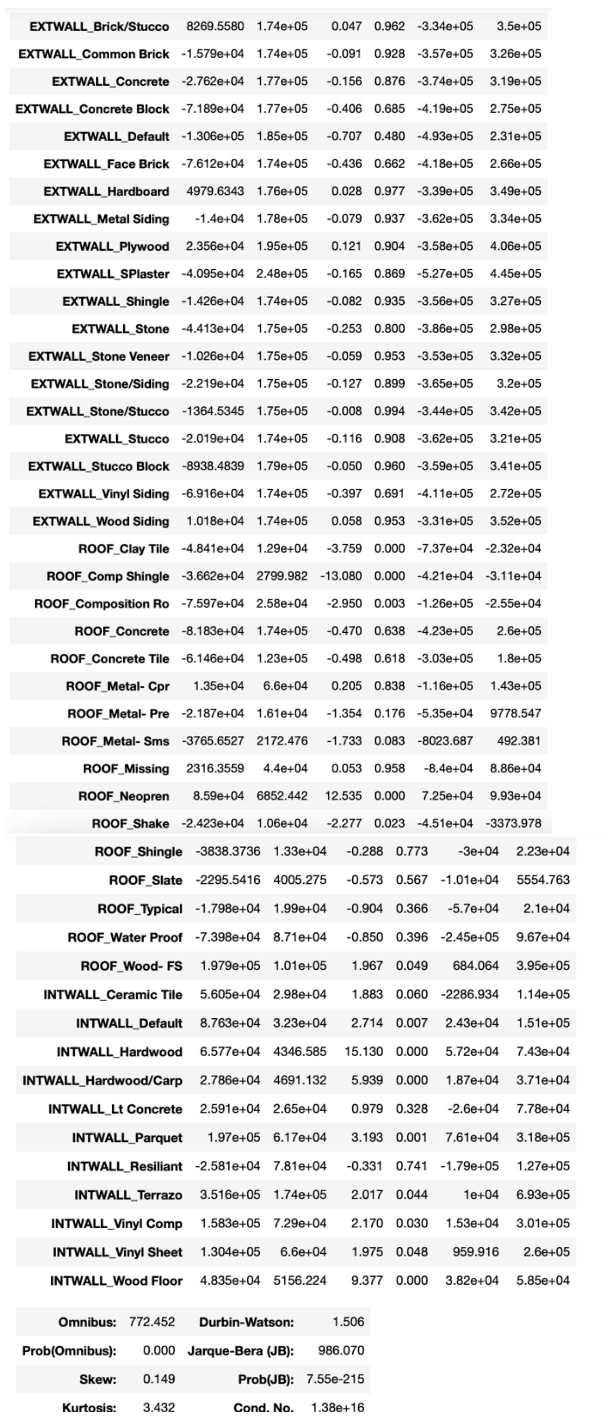
```
results.summary()
```



Dep. Variable:	PRICE	**R-squared:**	0.445
Model:	OLS	**Adj. R-squared:**	0.445
Method:	Least Squares	**F-statistic:**	682.3
Date:	Sat, 21 May 2022	**Prob (F-statistic):**	0.00
Time:	10:43:29	**Log-Likelihood:**	-1.1589e+06
No. Observations:	85937	**AIC:**	2.318e+06
Df Residuals:	85835	**BIC:**	2.319e+06
Df Model:	101		
Covariance Type:	nonrobust		



	coef	std err	t	P>|t|	[0.025	0.975]
const	-1.167e+06	2.57e+05	-4.542	0.000	-1.67e+06	-6.64e+05
BATHRM	8.504e+04	1164.216	73.048	0.000	8.28e+04	8.73e+04
HF_BATHRM	4.494e+04	1312.525	34.240	0.000	4.24e+04	4.75e+04
NUM_UNITS	-6.496e+04	4353.440	-14.921	0.000	-7.35e+04	-5.64e+04
ROOMS	3189.8637	627.744	5.081	0.000	1959.492	4420.236
BEDRM	2.338e+04	1027.076	22.763	0.000	2.14e+04	2.54e+04
AYB	-1436.2863	39.851	-36.042	0.000	-1514.393	-1358.179
EYB	2259.1784	46.905	48.165	0.000	2167.245	2351.112
STORIES	-2.505e+04	2673.190	-9.371	0.000	-3.03e+04	-1.98e+04
GBA	-53.5925	2.237	-23.963	0.000	-57.976	-49.209
KITCHENS	1.479e+04	4105.785	3.601	0.000	6738.654	2.28e+04
FIREPLACES	3.814e+04	1164.039	32.762	0.000	3.59e+04	4.04e+04
LANDAREA	-2.9952	0.503	-5.959	0.000	-3.980	-2.010
WARD	-2.485e+04	332.787	-74.660	0.000	-2.55e+04	-2.42e+04
HEAT_Air-Oil	3933.4964	4.74e+04	0.083	0.934	-8.9e+04	9.69e+04
HEAT_Elec Base Brd	1.042e+04	4.08e+04	0.255	0.799	-6.96e+04	9.05e+04
HEAT_Electric Rad	3.087e+04	4.35e+04	0.710	0.478	-5.43e+04	1.16e+05
HEAT_Evp Cool	-8.077e+04	5.24e+04	-1.540	0.124	-1.84e+05	2.2e+04
HEAT_Forced Air	3429.0719	3.92e+04	0.087	0.930	-7.34e+04	8.03e+04
HEAT_Gravity Furnac	-1428.2359	4.63e+04	-0.031	0.975	-9.22e+04	8.93e+04
HEAT_Hot Water Rad	1.503e+04	3.92e+04	0.383	0.702	-6.18e+04	9.19e+04
HEAT_Ht Pump	1.97e+04	3.92e+04	0.502	0.616	-5.72e+04	9.66e+04
HEAT_Ind Unit	3962.0647	4.96e+04	0.080	0.936	-9.32e+04	1.01e+05
HEAT_No Data	-3.195e+04	4.13e+04	-0.774	0.439	-1.13e+05	4.89e+04
HEAT_Wall Furnace	-5.775e+04	3.98e+04	-1.452	0.147	-1.36e+05	2.02e+04
HEAT_Warm Cool	-2.906e+04	3.92e+04	-0.741	0.459	-1.06e+05	4.79e+04
HEAT_Water Base Brd	-4.042e+04	4.09e+04	-0.988	0.323	-1.21e+05	3.98e+04
AC_N	-8.469e+04	3.08e+04	-2.751	0.006	-1.45e+05	-2.43e+04
AC_Y	-3.41e+04	3.08e+04	-1.109	0.268	-9.44e+04	2.62e+04
STRUCT_Missing	2316.3559	4.4e+04	0.053	0.958	-8.4e+04	8.86e+04
STRUCT_Multi	-7.783e+04	1.76e+05	-0.442	0.659	-4.23e+05	2.67e+05
STRUCT_Row End	-9.917e+04	1.76e+05	-0.563	0.573	-4.44e+05	2.46e+05
STRUCT_Row Inside	-8.923e+04	1.76e+05	-0.507	0.612	-4.34e+05	2.56e+05
STRUCT_Semi-Detached	-1.143e+05	1.76e+05	-0.649	0.516	-4.59e+05	2.31e+05
STRUCT_Single	-8.933e+04	1.76e+05	-0.508	0.612	-4.34e+05	2.56e+05
STRUCT_Town End	-9.207e+04	1.77e+05	-0.519	0.603	-4.39e+05	2.55e+05
STRUCT_Town Inside	-1.114e+05	1.77e+05	-0.631	0.528	-4.57e+05	2.35e+05
GRADE_Average	-3.146e+04	2037.150	-15.441	0.000	-3.54e+04	-2.75e+04
GRADE_Excellent	1.01e+05	6956.752	14.517	0.000	8.74e+04	1.15e+05
GRADE_Exceptional-A	1.017e+05	2.12e+04	4.802	0.000	6.02e+04	1.43e+05
GRADE_Exceptional-B	2.898e+05	1.01e+05	2.879	0.004	9.25e+04	4.87e+05
GRADE_Exceptional-C	1.703e+05	1.75e+05	0.976	0.329	-1.72e+05	5.12e+05
GRADE_Fair Quality	-4.088e+04	1.97e+04	-2.075	0.038	-7.95e+04	-2256.645
GRADE_Good Quality	8.025e+04	2397.936	33.468	0.000	7.56e+04	8.5e+04
GRADE_Low Quality	-3.952e+04	1.03e+05	-0.384	0.701	-2.41e+05	1.62e+05
GRADE_Missing	2316.3559	4.4e+04	0.053	0.958	-8.4e+04	8.86e+04
GRADE_Superior	1.02e+05	8191.472	12.448	0.000	8.59e+04	1.18e+05
GRADE_Very Good	1.162e+05	3811.649	30.494	0.000	1.09e+05	1.24e+05
CNDTN_Default	2.522e+05	1.74e+05	1.448	0.148	-8.92e+04	5.94e+05
CNDTN_Excellent	1.213e+05	8030.690	15.099	0.000	1.06e+05	1.37e+05
CNDTN_Fair	1.499e+04	8291.135	1.808	0.071	-1256.619	3.12e+04
CNDTN_Good	7.12e+04	1872.133	38.032	0.000	6.75e+04	7.49e+04
CNDTN_Missing	2316.3559	4.4e+04	0.053	0.958	-8.4e+04	8.86e+04
CNDTN_Poor	6.7e+04	1.99e+04	3.366	0.001	2.8e+04	1.06e+05
CNDTN_Very Good	1.283e+05	3233.116	39.676	0.000	1.22e+05	1.35e+05
EXTWALL_Aluminum	-1.244e+04	1.74e+05	-0.071	0.943	-3.54e+05	3.29e+05
EXTWALL_Brick Veneer	-3.656e+04	1.74e+05	-0.210	0.834	-3.78e+05	3.05e+05
EXTWALL_Brick/Siding	-5.048e+04	1.74e+05	-0.290	0.772	-3.92e+05	2.91e+05
EXTWALL_Brick/Stone	-3.511e+04	1.74e+05	-0.201	0.841	-3.77e+05	3.07e+05



EXTWALL_Brick/Stucco	8269.5580	1.74e+05	0.047	0.962	-3.34e+05	3.5e+05
EXTWALL_Common Brick	-1.579e+04	1.74e+05	-0.091	0.928	-3.57e+05	3.26e+05
EXTWALL_Concrete	-2.762e+04	1.77e+05	-0.156	0.876	-3.74e+05	3.19e+05
EXTWALL_Concrete Block	-7.189e+04	1.77e+05	-0.406	0.685	-4.19e+05	2.75e+05
EXTWALL_Default	-1.306e+05	1.85e+05	-0.707	0.480	-4.93e+05	2.31e+05
EXTWALL_Face Brick	-7.612e+04	1.74e+05	-0.436	0.662	-4.18e+05	2.66e+05
EXTWALL_Hardboard	4979.6343	1.76e+05	0.028	0.977	-3.39e+05	3.49e+05
EXTWALL_Metal Siding	-1.4e+04	1.78e+05	-0.079	0.937	-3.62e+05	3.34e+05
EXTWALL_Plywood	2.356e+04	1.95e+05	0.121	0.904	-3.58e+05	4.06e+05
EXTWALL_SPlaster	-4.095e+04	2.48e+05	-0.165	0.869	-5.27e+05	4.45e+05
EXTWALL_Shingle	-1.426e+04	1.74e+05	-0.082	0.935	-3.56e+05	3.27e+05
EXTWALL_Stone	-4.413e+04	1.75e+05	-0.253	0.800	-3.86e+05	2.98e+05
EXTWALL_Stone Veneer	-1.026e+04	1.75e+05	-0.059	0.953	-3.53e+05	3.32e+05
EXTWALL_Stone/Siding	-2.219e+04	1.75e+05	-0.127	0.899	-3.65e+05	3.2e+05
EXTWALL_Stone/Stucco	-1364.5345	1.75e+05	-0.008	0.994	-3.44e+05	3.42e+05
EXTWALL_Stucco	-2.019e+04	1.74e+05	-0.116	0.908	-3.62e+05	3.21e+05
EXTWALL_Stucco Block	-8938.4839	1.79e+05	-0.050	0.960	-3.59e+05	3.41e+05
EXTWALL_Vinyl Siding	-6.916e+04	1.74e+05	-0.397	0.691	-4.11e+05	2.72e+05
EXTWALL_Wood Siding	1.018e+04	1.74e+05	0.058	0.953	-3.31e+05	3.52e+05
ROOF_Clay Tile	-4.841e+04	1.29e+04	-3.759	0.000	-7.37e+04	-2.32e+04
ROOF_Comp Shingle	-3.662e+04	2799.982	-13.080	0.000	-4.21e+04	-3.11e+04
ROOF_Composition Ro	-7.597e+04	2.58e+04	-2.950	0.003	-1.26e+05	-2.55e+04
ROOF_Concrete	-8.183e+04	1.74e+05	-0.470	0.638	-4.23e+05	2.6e+05
ROOF_Concrete Tile	-6.146e+04	1.23e+05	-0.498	0.618	-3.03e+05	1.8e+05
ROOF_Metal- Cpr	1.35e+04	6.6e+04	0.205	0.838	-1.16e+05	1.43e+05
ROOF_Metal- Pre	-2.187e+04	1.61e+04	-1.354	0.176	-5.35e+04	9778.547
ROOF_Metal- Sms	-3765.6527	2172.476	-1.733	0.083	-8023.687	492.381
ROOF_Missing	2316.3559	4.4e+04	0.053	0.958	-8.4e+04	8.86e+04
ROOF_Neopren	8.59e+04	6852.442	12.535	0.000	7.25e+04	9.93e+04
ROOF_Shake	-2.423e+04	1.06e+04	-2.277	0.023	-4.51e+04	-3373.978
ROOF_Shingle	-3838.3736	1.33e+04	-0.288	0.773	-3e+04	2.23e+04
ROOF_Slate	-2295.5416	4005.275	-0.573	0.567	-1.01e+04	5554.763
ROOF_Typical	-1.798e+04	1.99e+04	-0.904	0.366	-5.7e+04	2.1e+04
ROOF_Water Proof	-7.398e+04	8.71e+04	-0.850	0.396	-2.45e+05	9.67e+04
ROOF_Wood- FS	1.979e+05	1.01e+05	1.967	0.049	684.064	3.95e+05
INTWALL_Ceramic Tile	5.605e+04	2.98e+04	1.883	0.060	-2286.934	1.14e+05
INTWALL_Default	8.763e+04	3.23e+04	2.714	0.007	2.43e+04	1.51e+05
INTWALL_Hardwood	6.577e+04	4346.585	15.130	0.000	5.72e+04	7.43e+04
INTWALL_Hardwood/Carp	2.786e+04	4691.132	5.939	0.000	1.87e+04	3.71e+04
INTWALL_Lt Concrete	2.591e+04	2.65e+04	0.979	0.328	-2.6e+04	7.78e+04
INTWALL_Parquet	1.97e+05	6.17e+04	3.193	0.001	7.61e+04	3.18e+05
INTWALL_Resiliant	-2.581e+04	7.81e+04	-0.331	0.741	-1.79e+05	1.27e+05
INTWALL_Terrazo	3.516e+05	1.74e+05	2.017	0.044	1e+04	6.93e+05
INTWALL_Vinyl Comp	1.583e+05	7.29e+04	2.170	0.030	1.53e+04	3.01e+05
INTWALL_Vinyl Sheet	1.304e+05	6.6e+04	1.975	0.048	959.916	2.6e+05
INTWALL_Wood Floor	4.835e+04	5156.224	9.377	0.000	3.82e+04	5.85e+04

Omnibus:	772.452	**Durbin-Watson:**	1.506
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	986.070
Skew:	0.149	**Prob(JB):**	7.55e-215
Kurtosis:	3.432	**Cond. No.**	1.38e+16

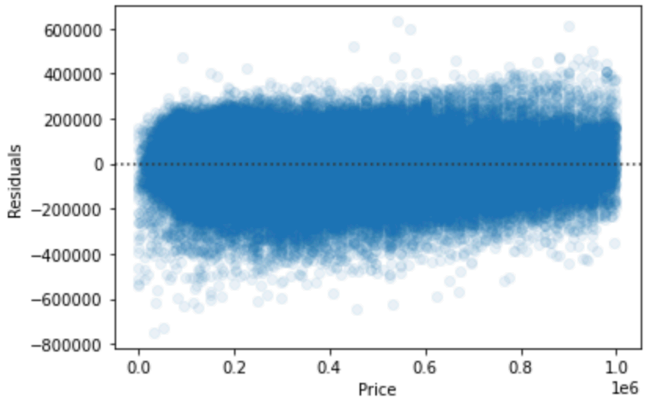
- The mean absolute error was calculated.

```
metrics.mean_absolute_error(y, pred_y)
```

```
135808.04005241138
```

- A residual plot was created.

```
residuals = y - pred_y
ax1 = sns.residplot(x = y, y = residuals, scatter_kws={'alpha': 0.08})
ax1.set(xlabel = 'Price', ylabel = 'Residuals');
```



Ordinary Least Squares was chosen for this study because the dependent variable is continuous. Another consideration was that the model summary offers the p-values for the predicting features. One advantage of OLS is it is efficient. It can create models of large data sets without needing extra computing power. One disadvantage is that OLS does not handle outliers well. Since the emphasis of OLS is to minimize the sum of the errors, any value that differs from the rest of the values by a significant amount will have a disproportionately large effect on the model. (ClockBackward, n.d.)

Results

The model has an R-squared value of 44.5%. The model explains 44.5% of the variance. The mean absolute error is $135808, which means that the predicted price is off by this amount on average. The residuals plot does appear to have an equal number of points above and below the line. It is hard to tell if they are spread evenly because of the density of the points themselves.

The main result of this study is that the null hypothesis is rejected. Ward was found to be statistically significant. Of the 105 features used, 37 were found to be statistically significant. They are as follows:

Feature	Coefficient
GRADE_Good_Quality	802500

INTWALL_Terrazo	351600
GRADE_Exceptional-B	289800
ROOF_Wood-FS	197900
INTWALL_Parquet	197000
INTWALL_Vinyl Comp	158300
INTWALL_Vinyl Sheet	130400
CNDTN_Very Good	128300
CNDTN_Excellent	121300
GRADE_Very Good	116200
GRADE_Superior	102000
GRADE_Exceptional-A	101700
INTWALL_Default	87630
ROOF_Neopren	85900
BATHRM	85040
CNDTN_Good	71200
CNDTN_Poor	67000
INTWALL_Hardwood	65770
INTWALL_Wood Floor	48350
HF_BATHRM	44944
FIREPLACES	38140
ROOF_Comp Shingle	36620
INTWALL_Hardwood.Carp	27860
BEDRM	23380
KITCHENS	14790
ROOMS	3189.8637
EYB	2259.1784
LANDAREA	-2.9952
GBA	-53.5925
AYB	-1436.2863
ROOF_Shake	-24230
WARD	-24850
STORIES	-25050
ROOF_Clay Tile	-48410
NUM_UNITS	-64960
ROOF_Composition Ro	-75970
AC_N	-84690

One limitation of this study is the amount of data that is missing. Deleting rows and adding a missing class to some features may have introduced bias into the study. The new

'Missing' class also added new features to the study when the categorical features were encoded, which may have contributed to the model's poor performance.

Based on this study, we can see that the feature that adds the most value to a home is having a grade of good. Not having an air conditioner lowers the home price the most. While all these features are significant to the price of a house, investors and homebuyers should focus on the features that add the most value, like condition, grade, and the type of interior.

One direction that a future study could take is to take a non-linear approach to the data. The data does not meet all the assumptions of using a linear model. A non-linear model may be better at predicting the final home price.

A second direction for a future study would be to see if the sales price has a systematic pattern over a period of time.

Works Cited

Chrisc. (2018). *D.C. Residential Properties*. Retrieved March 2022, from Kaggle: https://www.kaggle.com/datasets/christophercorrea/dc-residential-properties?select=DC_Properties.csv

ClockBackward. (n.d.). *Ordinary Least Squares Linear Regression: Flaws, Problems and Pitfalls*. Retrieved May 2022, from University of Illinois Chicago: https://lcn.people.uic.edu/classes/che205s17/docs/che205s17_reading_06c.pdf

Kishan, H., & Ganguly, S. (2022, March 2). *U.S. house prices to rise another 10% this year*. Retrieved May 2022, from Reuters: https://www.reuters.com/business/us-house-prices-rise-another-10-this-year-2022-03-02/

Orton, K. (2022, April 28). *2021 was another strong year for the D.C. area housing market*. Retrieved May 2022, from The Washington Post: https://www.washingtonpost.com/business/2022/04/28/2021-dc-area-housing-market/

Python Advantages and Disadvantages - Step in the right direction. (n.d.). Retrieved March 2022, from TechVidvan.com: https://techvidvan.com/tutorials/python-advantages-and-disadvantages/

Sharma, N. (2020, August 13). *Ordinary Least Squared (OLS) Regression*. Retrieved May 2022, from Medium: https://medium.com/analytics-vidhya/ordinary-least-squared-ols-regression-90942a2fdad5