C964: Computer Science Capstone

Task 2 parts A, B, and D

Part A: Project Proposal for Business Executives

Letter of Transmittal

Ms. Jane Doe

Real Realtors

917 W Lincoln Rd

New York, New York

Dear Ms. Doe,

Real Realtors has built a strong brand in the real estate industry. To maintain our place in the industry it is imperative that we maintain a strong working knowledge of the housing market. With an ever-changing housing market, it becomes difficult to stay up to date on the value of houses and make the correct investments, as well as the proper recommendations to our clients. If we cannot keep up with market changes we risk losing business and in return risk profits. To better prepare Real Realtors for the future, I am proposing a machine-learning solution that predicts the value of the home based on amenities.

The product I propose will directly provide insight into the value of any home simply by providing five pieces of information about the house. With only the area of the house, the number of bedrooms, the number of bathrooms, the number of stories, and how much parking is available we will be able to estimate the value of a home. This estimation will be essential to future investments and to aid our clients. For example, with the ability to estimate a house's value we can help our customers narrow down their house search before they even have to travel. This will allow buyers with budgets to immediately filter their options while also saving gas money, miles traveled, and frustration. The best part is that this product will only cost us $5,000 upfront and $500 quarterly to maintain up-to-date data. Building this product will require one developer whose expertise includes a bachelor's degree in computer science and three years of experience developing applications including machine learning products.

Project Recommendation

Problem Summary

A volatile housing market makes buying and selling homes a constant challenge. Buying a home often takes time– sometimes months, sometimes years. When our buyers go to purchase a house in February they may find that they have to pause their search until October. In those eight months of saving or getting their affairs in order the available choices may, and usually do, change significantly. This translates to more searching, meaning more miles traveled, more gas spent, and more frustrated customers. To limit these malefactors, I recommend a product that will predict housing prices based on five features of the home. These five features are simple: the size of the home, the number of bedrooms, the number of bathrooms, the number of stories, and the amount of parking available. Many customers will come to us with some of these features already in mind. Providing the customer with an immediate housing estimate with features of a home will greatly improve our speed and efficiency in selling homes making our customers happier and our profits larger. To achieve this product we will have a developer create an application that uses an algorithm to identify patterns in housing features using free data from the internet.

Application Benefits

The product will simplify searches for our buyers which will lead to quicker closings for our real estate agents. With faster sell times, we will have more time to sell more homes. Not only does this simplify parts of our business; it also helps us to provide an immediate sense of where the housing market's value stands. This will help customers come to decisions on whether to buy or not quicker and help filter out irrelevant options for our customers aligning budgets and options instantaneously.

Application Description

The product will have several parts to it. First, data will be taken from a free data collection source called Kaggle.com. Next, the data will be processed for modeling by consolidating all housing features to only include lot size, number of bedrooms, number of bathrooms, number of stories, and parking allotment. Then this data will be split for training and testing. With the training section of the data, we will apply a regression algorithm called the Ridge algorithm. We will then use the test data to check for accuracy. Lastly, provided the test data proves accurate, we will then be able to make value predictions on homes using the five housing features mentioned earlier.

Data Description

The data for our product will come from a website called Kaggle.com. This data will contain the current house value as well as the five necessary features; area of the house, number of bedrooms, number of bathrooms, number of stories, and amount of parking spaces. The dependent variable will be the housing price, while the independent variables will be the corresponding five features. It will be important that the data be limited to these five features as they are numerical which the model will specifically be trained to observe. If the data extends beyond these features it will be necessary to drop or remove the extra data. Future maintenance will require searches for and training on more data to maintain accuracy in the current housing market.

Objectives and Hypothesis

There are several objectives to this project. The main objective is to improve the quality of service for our customers. To do this, we must accomplish another objective of developing a product that can predict the value of homes given data on specific features of the home. With this product, we will be able to filter irrelevant search options for our customers and save time, money, and frustration.

I hypothesize that we can apply a linear regression algorithm on housing data to achieve predictions of home values for our customers. These predictions don't need to be perfect or completely accurate, but instead accurate enough to provide insight to our customers to aid in their budgeting and expectations.

Methodology

For this project, we will use the waterfall methodology. The reason we would use the waterfall methodology over the agile methodology is because we do not need significant customer input to build as well as having clearly defined requirements.

Our waterfall process will look like this;

1. **Requirements:** We will meet with clients and supervisors to establish clear requirements to implement.

2. **Design:** Next, we will develop a design that ensures all requirements are met and can be implemented within reasonable parameters.
3. **Implementation:** In this phase, we will begin coding within the parameters of both the design and requirements.
4. **Verification:** This phase will happen when coding is complete and will encompass an accuracy test for predictions as well as a trial period for our staff to pilot with the end users to ensure functionality and observe for deficiencies.
5. **Maintenance:** To maintain this product, it will be necessary for a developer to insert modern housing data into the model. This will be necessary to ensure that the predictions stay within an acceptable accuracy range as the housing market continues to shift.

Funding Requirements

This product uses mostly free resources online meaning all the costs come from developer labor. We will need one developer for 100 hours upfront and 10 hours quarterly for maintenance. At an average salary of $100,000, we can expect to pay this single developer $50/hour. At this rate, it will cost $5,000 to build this product and $500 quarterly, or $2,000 yearly to maintain the accuracy of this product.

Data Precautions

Fortunately, we will not need to take data precautions. Our data is sourced from Kaggle.com which is a public source for data. Not only is it publicly sourced, it does not contain any PHI (personal health information), classified information, or company data that could harm business.

Developer’s Expertise

The developer for this product is well-suited to accomplish our objectives. They have three years of experience in developing applications, familiarity with the language used for this product (Python), have proposed a machine learning solution previously and holds a bachelor’s degree in computer science. These qualifications suit the needs of this project well considering it will be written in Python and it is a machine learning problem.

Part B: Project Proposal

Project Statement

Over the last century, houses have continued to rise in price leading to a volatile market. This increase seems to move at a rate that makes buying a home a different experience from one month to the next. Our customers spend increasing amounts of time searching for homes within their budget and trying to keep up with ever-changing prices. This is also particularly onerous on our realtors who spend that search time with customers and are incentivized to sell as many homes as quickly as possible. These extra miles traveled and wasted search times lead to increased frustration between our customers and realtors which is bad for business.

Customer Summary

Our customers are a range of adults from different regions across the country. Their needs and budgets vary widely from first-time home buyers to retirees looking to settle down for their golden years. Customers expect from us deliverable insight into the home buying process which can be difficult to keep up with as the market is constantly shifting. This can be alleviated through the proposed market analysis product. The product will provide instant feedback on the current housing market by analyzing hundreds and soon-to-be thousands of real-life examples of housing prices and comparing five typical housing features that directly affect those prices. With the ability to immediately give our customers housing analysis, they can come to quick conclusions on what their budget can get for them, or even how many bedrooms and bathrooms they can expect to afford. This immediate expectation setting will save traveled miles, time spent looking, and money in gas and time off of work which will lead to less frustration from our customers.

Existing System Analysis

Currently, our customers rely on applications that list houses for sale and their amenities. While this has made access to home buying easier and more convenient, it doesn’t quite tell our customers what they could afford, or much less expect to pay given the chance for outliers and fleeting opportunities. For example, a customer might find their dream home for a bargain but lose the home due to a bidding war, timing of finances, or work-related constraints. This can lead to a home buyer with an expectation that this is a normal price, or that their dream home is available and all they need to do is keep scrolling or refreshing. With our product, it would become clear what is the expectation or how much their average dream home would go for without suffering from confirmation bias.

Data

Our data comes from a free source on the internet called Kaggle.com. This set of data contains 545 entries of homes with pricing and features such as bedrooms, bathrooms, and the size of the home. Fortunately, this data has no repeating values or missing values. To process this data we will need to make sure we use the same features across the board, with strict attention to the nature of the data. What this means is, we must make sure the features we use will be numerical only. If the data is not numerical only, the model to be trained will require all predictions to be transformed and trained on non-numerical data making the predictions less user-friendly. Due to this constraint the data to start, as well as any future maintenance data, will be required to have the size of the home, the number of bedrooms, the number of bathrooms, the number of stories, and how many parking spaces are available. To do this, we will simply need to drop, or remove, any alternative features.

Project Methodology

We will be employing the waterfall method for this project. The reason we will utilize the waterfall method is that we will not require significant input from clients and we will also have well-defined requirements. Each stage of development will be performed by a single developer and will cascade from the first to the next to the last. Below is what our methodology will look like under the waterfall method:

1. Requirements: We will meet with end users such as our realtors and customers to discuss all necessary requirements that will aid in this endeavor or the home-buying process in general. It is in this phase that we will establish a scope for our product to prevent unnecessary features from being implemented.

2. Design: In this stage of the development we will analyze the requirements and plan for success. The design will describe how our product should work, and how each part will coalesce into the whole.

3. Implementation: From here, we will begin developing our product based on the design from the previous stage. We will functionally test each part before we move to the next. For example, we will ensure that our data is numerical and split before we begin training our model.

4. Verification: Next we will put our pieces together and make sure it works as intended. To do this we will utilize integration testing which ensures the working parts work as a whole. Not only will we perform integration testing, but we will also verify prediction accuracy with the R^2 scoring (Coefficient of Determination). This will ensure that our analysis is in a range acceptable for our customers.

5. Maintenance: At this stage, we will have a working product that can be deployed for pilot programs. In these pilot programs, we will collect feedback from our realtors and end users for bugs or defects. It will also be necessary to maintain this product with quarterly updates. Several times a year, it will be important to add new modern data on housing to ensure accurate predictions for the changing market.

Project Outcomes

When this project is completed, the product will be given to realtors in a pilot program to use with a portion of our customers. The product will have instructions on how to make predictions and a user guide to aid in inserting values. The user guide will also detail the necessary links and data to properly install the necessary components. From there, the product will be able to take the five features of typical homes: the size of the house, the number of bedrooms, the number of bathrooms, the number of stories, and the number of parking spaces available, to receive an instant prediction of home value.

Implementation Plan

To implement this product, we will have to integrate the product into the production environment. This is how we will go about doing so:

- General Strategy - To make this as simple for our end users as possible, we will aim to have all necessary libraries and data localized for use without installation. To do this, we can use a free service on the internet to host the Jupyter Notebook. Using a service like Google Colab we can provide a link for our realtors that has a setup environment with all the instructions and data ready for use. This will make it so we do not have to rely on integrating this product with other systems in our organization. With a small amount of training, coupled with a user guide, our realtors will be able to offer budget advice and insight into the housing market quickly and at a low cost to the company.

- Phases of the rollout - To start, we will pilot this program to a subsection of our realtors. This will be given to realtors with lower than average traffic as they will have fewer time constraints and subsequently will be easier to train and employ the new service. During this pilot, our realtors will be taking feedback from our customers and recording errors. Next, given a successful pilot program, the product will be distributed to more trafficked subsections where time will be made to train these realtors on our service. After that stage of the rollout, the product will be dispersed throughout the remaining subsections and unique areas where the last training will occur.
- Testing - Testing will occur at each stage of the rollouts. However, the first stage of the rollout will have the most impact on testing overall. With the first stage of the rollout completed, the trained realtors will have had the first opportunity to give feedback which will cascade to the rest of the deployment. At the end of each rollout, the newly trained realtors will submit errors to a hub where fixes can be applied before the next rollout phase.

Evaluation Plan

The product will be verified through a scoring metric. The R^2 (Coefficient of Determination) method will be applied to the model to measure the variance between the dependent variable (housing price) and the independent variables (housing features) in a regression algorithm. We will aim for a score of 0.7 which will indicate a high level of correlation. While we are aiming for a high level of correlation, it can be acceptable to reach close due to the nature of a volatile housing market. It will be considered ineffective if the model scores under

0.5. It will be necessary to perform this scoring during maintenance as well, for we will be adding more data over time which can lead to changes in accuracy.

Resources and Costs

Due to the free resources being utilized from the internet, many parts of this product will have negligible costs with the main cost coming from labor. Considering we already pay for electricity and terminals, as well as the salaries of our realtors it is safe to argue that this product will cost an estimated $5,000 in developer costs upfront, with an additional $2,000 in maintaining and updating the product yearly. Our data will be provided for free from Kaggle.com. Our hosting service in the form of Google Colab will also be free. This leaves the breakdown for labor. We will need one developer who is paid at an average of $50/hour for 100 hours of upfront labor. Our maintenance and updates will also be $50/hour for 10 hours quarterly (four times a year).

Timeline and Milestones

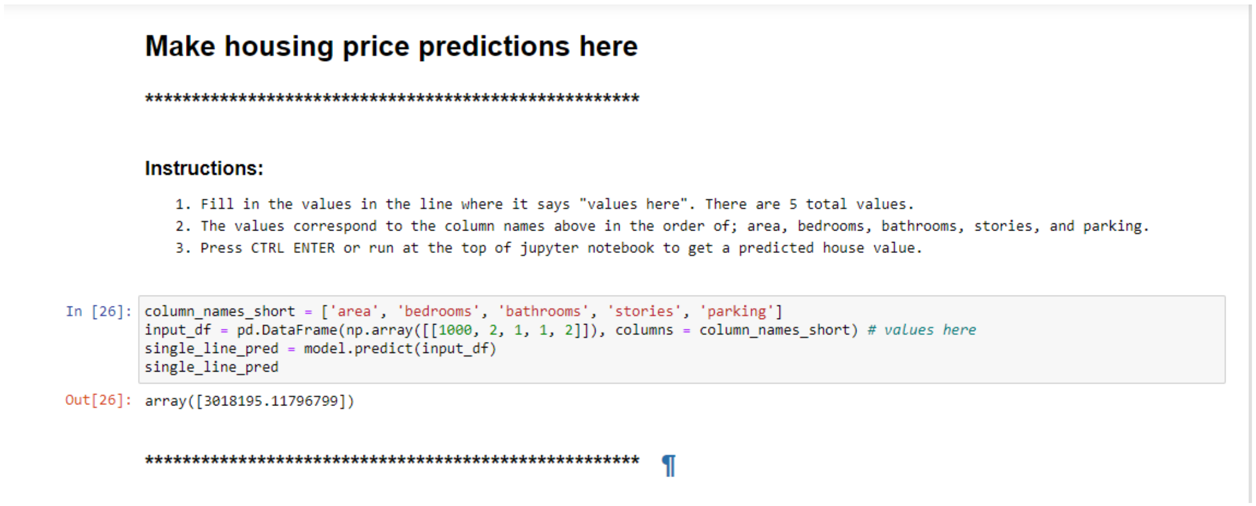
The project will take 100 hours to finish, and 40 hours per year to maintain.

Milestone	Start and End Dates	Duration	Resources
Requirement Gathering	May 22 - May 23	10 hours	End Users, Realtors, Software Developer
Design	May 24 - May 26	10 hours	Software Developer
Implementation	May 29 - June 2	40 hours	Software Developer
Testing	June 5 - Jun 7	20 hours	End Users, Realtors, Software Developer
Deployment	June 8 - June 12	20 hours	End Users, Realtors, Software Developer
Maintenance	quarterly	10 hours	Software Developer

Part D: Post-implementation Report

Project Purpose

With the ever-changing housing market and the nature of our realty business, making a poor investment can often be speculative or even perilous. Our customers want to make sure when they invest in housing that they are getting what they paid for. This housing value predictor aids in estimating this value, providing the client with insight as well as insight for our personal investments too. To accomplish this, the program trained a regression model on hundreds of housing data entries. This model sought out patterns in features such as; the area of the house, the number of bedrooms, the number of bathrooms, the number of stories, and the amount of parking. With these patterns, the regression algorithm estimated a value for the input.



Make housing price predictions here

****

Instructions:

1. Fill in the values in the line where it says "values here". There are 5 total values.
2. The values correspond to the column names above in the order of; area, bedrooms, bathrooms, stories, and parking.
3. Press CTRL ENTER or run at the top of jupyter notebook to get a predicted house value.

```
In [26]: column_names_short = ['area', 'bedrooms', 'bathrooms', 'stories', 'parking']
         input_df = pd.DataFrame(np.array([[1000, 2, 1, 1, 2]]), columns = column_names_short) # values here
         single_line_pred = model.predict(input_df)
         single_line_pred

Out[26]: array([3018195.11796799])
```

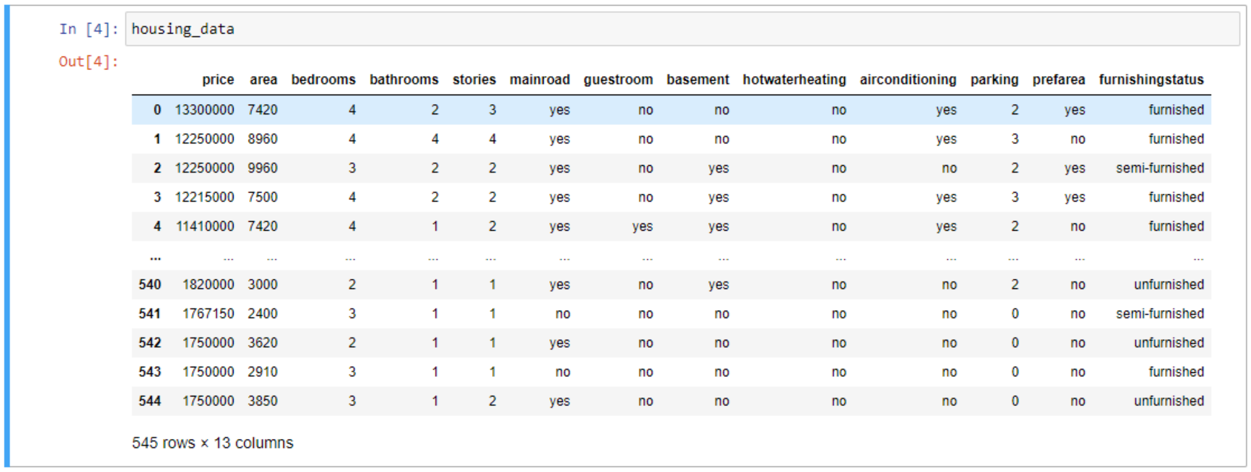
** ¶

With the example above, you can see in red the features of the house, and in green the corresponding values. The output below, 3018195 (rounded to the whole number) is the estimated value of the house.

Datasets

The raw data consists of 545 entries with 13 total features or columns. The processed data consists of 5 features and 1 target, split into training sets and test sets. Twenty percent of the data was split to provide a test set for more accurate measures. Fortunately, there was not a lot of processing needed for the data. There were no missing values and no duplicate values. The only processing involved the data split and dropping eight non-numerical features to input data frames for the user.

Preprocessed Data

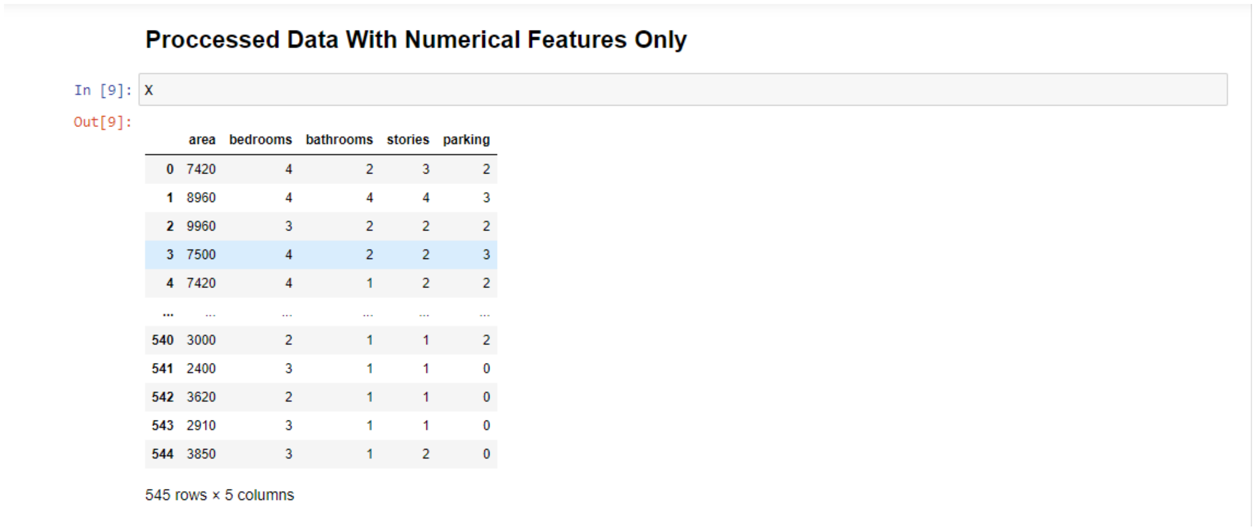


```
In [4]: housing_data
Out[4]:
```

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnished
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnished
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished
...	...	...	...	...	...	...	...	...	...	...	...	...	...
540	1820000	3000	2	1	1	yes	no	yes	no	no	2	no	unfurnished
541	1767150	2400	3	1	1	no	no	no	no	no	0	no	semi-furnished
542	1750000	3620	2	1	1	yes	no	no	no	no	0	no	unfurnished
543	1750000	2910	3	1	1	no	no	no	no	no	0	no	furnished
544	1750000	3850	3	1	2	yes	no	no	no	no	0	no	unfurnished

545 rows × 13 columns

Processed Data



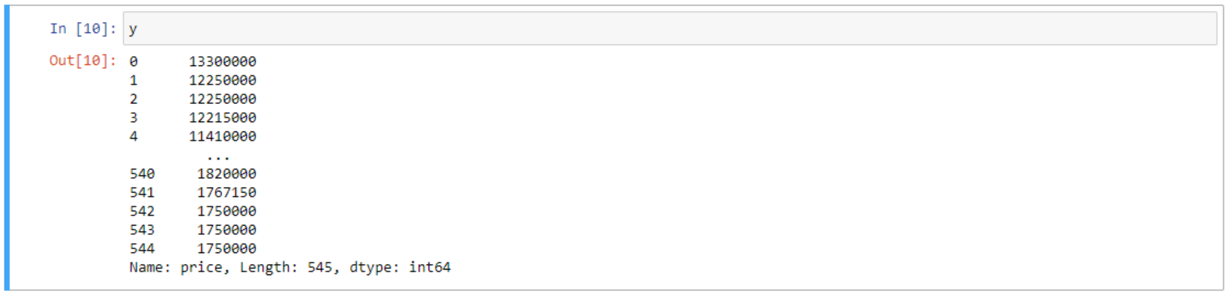
Proccessed Data With Numerical Features Only

```
In [9]: X
Out[9]:
```

	area	bedrooms	bathrooms	stories	parking
0	7420	4	2	3	2
1	8960	4	4	4	3
2	9960	3	2	2	2
3	7500	4	2	2	3
4	7420	4	1	2	2
...	...	...	...	...	...
540	3000	2	1	1	2
541	2400	3	1	1	0
542	3620	2	1	1	0
543	2910	3	1	1	0
544	3850	3	1	2	0

545 rows × 5 columns

Target Data (Prices)



```
In [10]: y
Out[10]: 0       13300000
         1       12250000
         2       12250000
         3       12215000
         4       11410000
                   ...
         540      1820000
         541      1767150
         542      1750000
         543      1750000
         544      1750000
         Name: price, Length: 545, dtype: int64
```

Access to Data Set

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

Data Product Code

This product predicts the value of a house when given features such as: the area of the house, the number of bedrooms, the number of bathrooms, the number of stories, and how much parking it can accommodate. To do this, a regression algorithm is applied to a training data set. The regression algorithm used is the Ridge algorithm. The Ridge algorithm was chosen because it scored the highest accuracy in the scoring method among three different regression algorithms. The Random Forest Regressor algorithm scored the lowest, despite tuning the n_estimators, with a score of .49 (rounded to hundredths). The next lowest-scoring algorithm was the Lasso algorithm which only narrowly lost to the Ridge algorithm with a score of .5464. The Ridge algorithm scored the highest, narrowly, with a score of .5465. The training data is eighty percent of the entire data set (545 total entries) made up of numerical data only. This was made possible by dropping features such as “furnishing status”. Non-numerical data would have made single predictions more burdensome for the user. With the training data set processed and the algorithm chosen, the model trained on eighty percent of the data looking for patterns between the 5 features and the target value. Finally, the test data, comprised of only twenty percent of the data, was applied to the trained model for scoring.

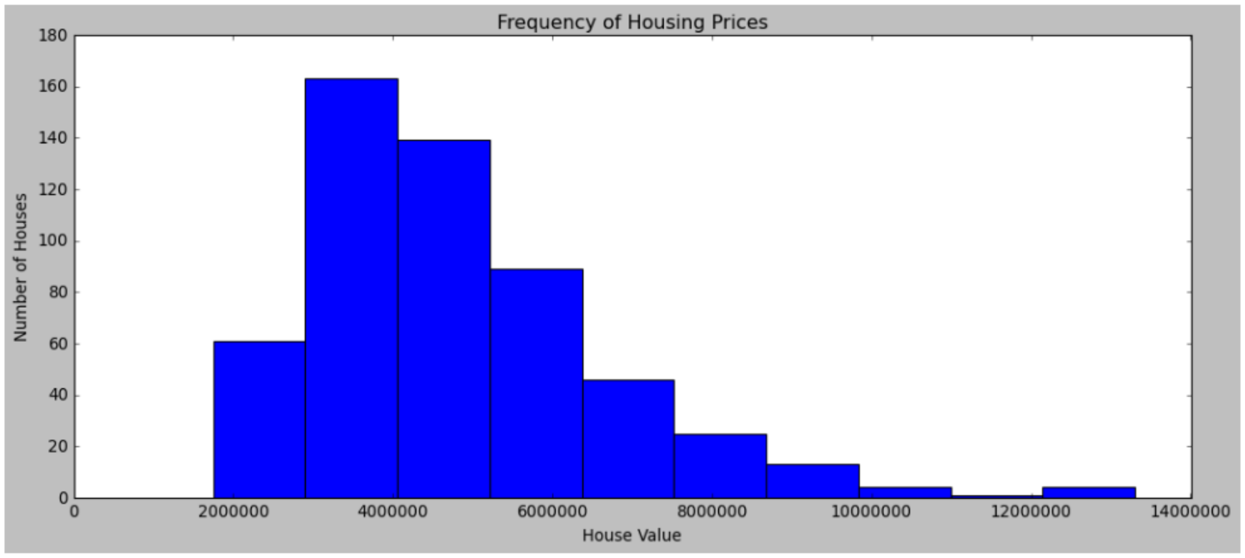
Three descriptive methods were used to get a feel for the data. There is a bar graph displaying how the number of bedrooms directly affects the value of the house. There is also a histogram that displays how many data entries fall under different housing values to provide a sense of the difference between the average housing price and the median housing price. Lastly, there is a scatter plot that compares the area of the house with its value to illustrate outliers that skew the average. These descriptive methods are created using the Matplotlib Library for the Python programming language. In addition, to insert the data into the Matplotlib Library, the Pandas Library was used to create data frames to hold the housing data.

Hypothesis Verification

The hypothesis for this project was that, if given enough data with enough features, a reasonable prediction can be made for the value of a house. Given the accuracy rating from the R^2 (R squared, or “The Coefficient of Determination”) evaluation the hypothesis is not yet proven. The score of .55 (rounded to hundredths) leaves much to be desired. Still, with room for improvement and fine-tuning, along with the use of larger data sets, it is not yet clear that the hypothesis has been ruled out.

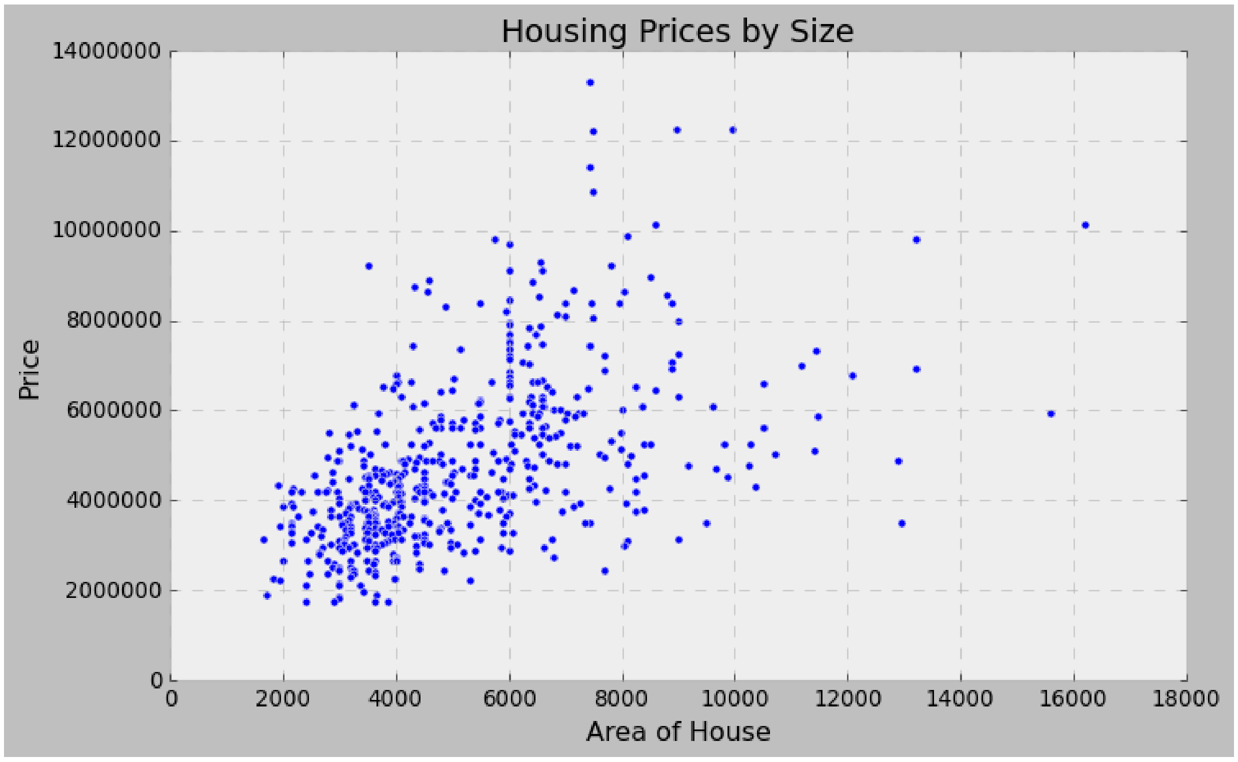
Effective Visualization and Reporting

The three descriptive methods used were a bar graph, a histogram, and a scatter plot. Each one gives a feeling for the data set and provides insight into the housing market. The histogram shows the number of times a house value appears in the data. This is beneficial to understand the mean home price, as there are few outliers.

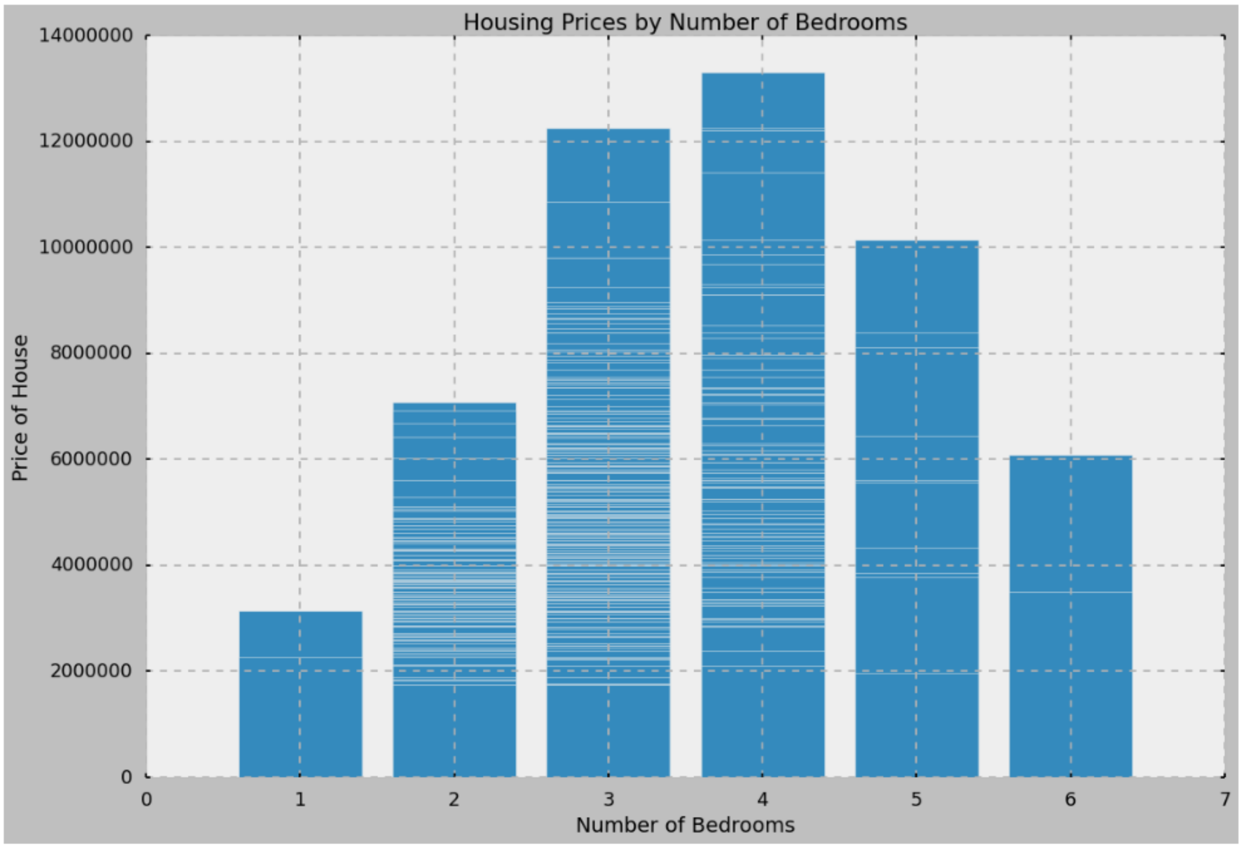


The scatter plot depicts the value of the house when compared to the size. This is useful for seeing where the concentration of the average-sized home falls in value.

The scatter plot also makes it very easy to see how many outliers there are providing more stable data analysis.



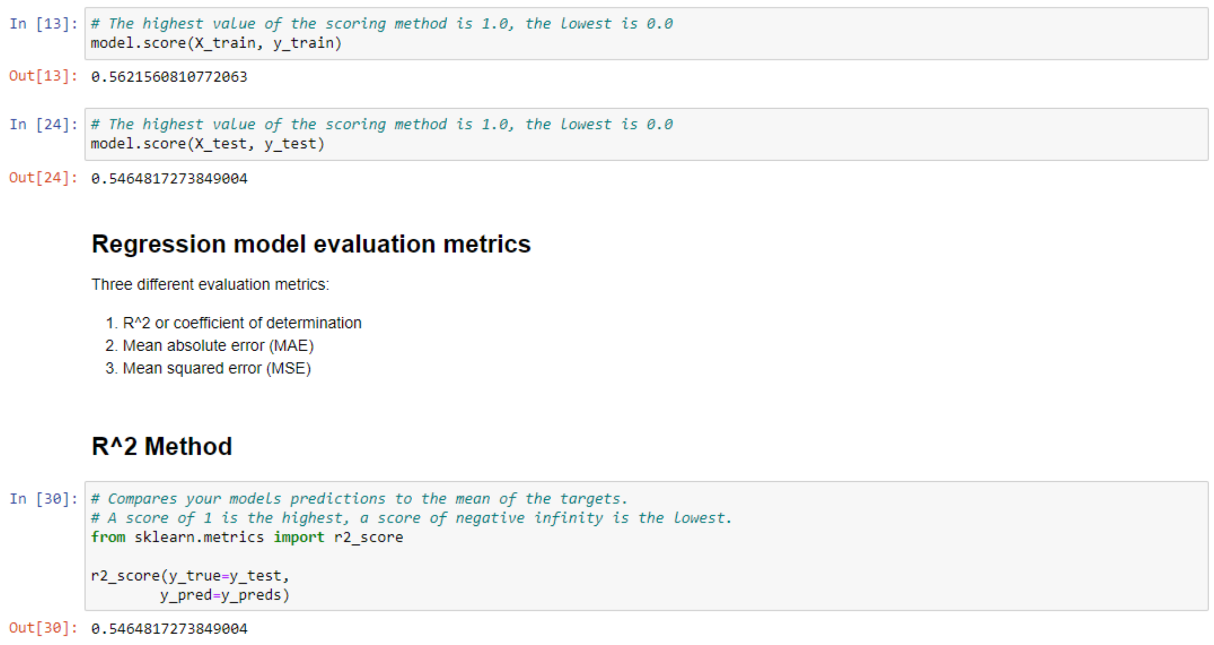
Lastly, there is a bar graph plotting how the number of bedrooms affects the value of the house. This was interesting because it showed the largest amount of bedrooms did not always correspond to the highest house value. Instead, the highest house values contained the average number of bedrooms (between three and four).



Accuracy Analysis

This program utilized a built-in method from the Scikit-Learn library. There are two similar variations for measurement used, the score() method and the imported R^2 method. Both of these methods focus on providing an accuracy percentage for predictions. The difference between the two is that R^2 (or Coefficient of Determination) is more robust and specifically targets the amount of error between the actual values and the predictions whereas the score()

method simply targets the error between actual values and predictions. For this program, each method scored the same, 55% (rounded). This score leaves much to be desired, likely due to the amount of data available to train on which was 436 housing entries (80% of the total 545). Further, as the model is trained on the data set and the housing market is constantly changing it can be understood there is significant room for error to apply housing features from future homes to this program.



```
In [13]: # The highest value of the scoring method is 1.0, the lowest is 0.0
         model.score(X_train, y_train)
Out[13]: 0.5621560810772063

In [24]: # The highest value of the scoring method is 1.0, the lowest is 0.0
         model.score(X_test, y_test)
Out[24]: 0.5464817273849004
```

Regression model evaluation metrics

Three different evaluation metrics:

1. R^2 or coefficient of determination
2. Mean absolute error (MAE)
3. Mean squared error (MSE)

R^2 Method

```
In [30]: # Compares your models predictions to the mean of the targets.
         # A score of 1 is the highest, a score of negative infinity is the lowest.
         from sklearn.metrics import r2_score

         r2_score(y_true=y_test,
                  y_pred=y_preds)
Out[30]: 0.5464817273849004
```

Application Testing

This application was evaluated with functional testing. Sections such as; splitting the data, training the model, scoring the model, and making predictions were all individually tested to determine if the software performed as it was supposed to. One way this improved the application was during the predictions section of the program. At first, the training data was modified to have more features by transforming non-numerical data into numerical data. This yielded a slightly higher accuracy rating, however, made it significantly more difficult for the user to make predictions on a single house. With the transformed features apart of the training data, the model expected new predictions also to have numerically transformed inputs. Ultimately, the non-numerical data was dropped to make the user experience more friendly through simplicity.

Application Files

Program Prerequisites: Jupyter Notebook, Numpy, Pandas, Matplotlib

To access this notebook click this link or copy and paste in URL:

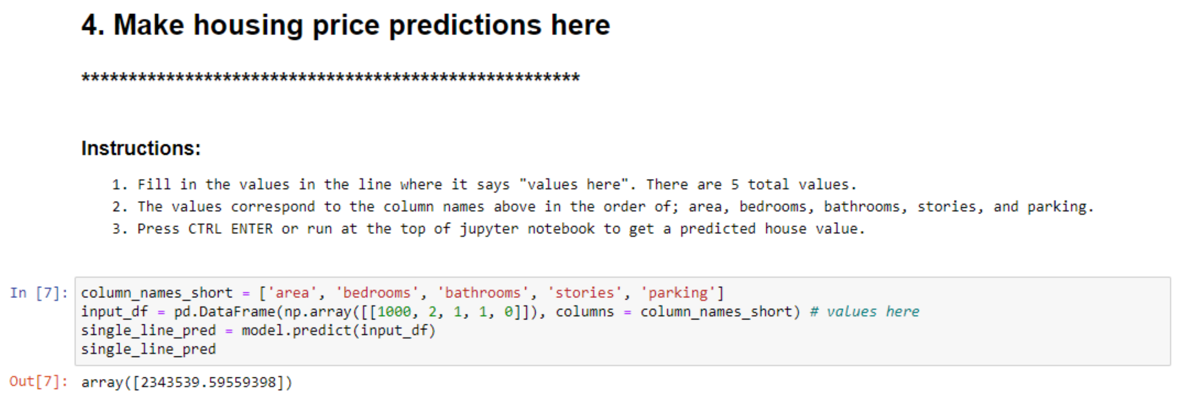
https://colab.research.google.com/drive/1nXb315JRH3IYmgAfsgwvmVIrVYs5Lx7m#scrollTo=1203c912

When tested, the link provided the notebook, and necessary libraries mentioned above, as well as the data set necessary to run without the need for installation.

User Guide

1. To start either click the link in the ‘Application Files’ section above or a copy of the link is provided in the zipped folder named “link_to_project.txt”
2. The link will take you to a Google Colab page where the program and libraries are ready to use.
3. Next you will need to upload the data csv for the session
4. On the left side of the page, click on the folder icon (if you hover over it, it’s labeled “files”)
5. Then you will click on an icon of a page with an upward arrow to upload the housing data.
6. Navigate to “Housing.csv” located in the zipped submission and click it.

7. Then, you will run the cells in order from 1 through 4. Simply locate the header with the number, and in the cell below run by pressing shift + enter.
8. When you reach cell number 4 there will be instructions on how to enter housing features to receive a single prediction on the value of the house.
9. To make a prediction, you will go to the cell below the header number '4' as seen below:



4. Make housing price predictions here

****

Instructions:

1. Fill in the values in the line where it says "values here". There are 5 total values.
2. The values correspond to the column names above in the order of; area, bedrooms, bathrooms, stories, and parking.
3. Press CTRL ENTER or run at the top of jupyter notebook to get a predicted house value.

```
In [7]: column_names_short = ['area', 'bedrooms', 'bathrooms', 'stories', 'parking']
        input_df = pd.DataFrame(np.array([[1000, 2, 1, 1, 0]]), columns = column_names_short) # values here
        single_line_pred = model.predict(input_df)
        single_line_pred

Out[7]: array([2343539.59559398])
```

10. Next, replace the **green** values with the values you wish to predict value from. The first green value represents the area of the house. The second green value represents the number of bedrooms in the house. The third green value represents the number of bathrooms in the house. The fourth green value represents the number of stories the house has. The last green value represents how much parking the house can accommodate.
11. Press shift + enter, and inside the brackets will be the predicted house value. In the example above the house value is predicted at $2,343,539 (rounded to whole number).

Summation of Learning Experience

Starting this project reminded me that there will always be more to learn. I had several academic experiences that helped me in my development. The first and most obvious, to me, was the ‘Data Structures and Algorithms II’ course. That course lead me to the Python language as I had learned Python syntax and its proclivity for data manipulation which seemed appropriate for a machine learning solution. Another source of academic aid came from both Software I and Software II courses. While those courses were in another language, one of the most valuable experiences I learned came from the way those courses were structured. I got hands-on learning by taking projects one step at a time. Incremental progress pays dividends and aided me through this project just as much as the software courses.

While I had many academic experiences that contributed to my planning and execution, I felt there were still subjects I wanted to better understand before I applied them, such as machine learning. To overcome gaps in knowledge I decided to further my learning with a course on Udemy. It wasn’t free, but the information is vital for my education and likely my future career. In this course, I got familiar with Jupyter Notebook and its cell-based run time. The course also familiarized me with my first libraries as most academic courses had not allowed external libraries. These libraries were; ‘Numpy’, ‘Pandas’, ‘Matplotlib’, and ‘Scikit-Learn’. Each library added value, and experience, and made it easier to accomplish tasks.

Ultimately, this project experience reminded me that there is always more to learn. Upon starting this degree I imagined when I reached this point I would feel like I have learned a significant amount of computer science. Now as I sit at the end, I realize that I am nowhere close to being an expert but instead have the tools to incrementally grow with each day, project, lesson, and error.