IDS 400- Programming for Data Science

Final Project

Zillow Zestimate: Property Price Valuation

Jigyasa Sachdeva
664791188
jsachd3@uic.edu

Varun Maheshwari
671624467
vmahes3@uic.edu

Introduction

- Zillow's Zestimate home valuation has shaken up the U.S. real estate industry since first released 11 years ago. For context, when Zillow first launched the Zestimate in 2006, it marked the first time that homeowners could find an instant, free estimate of their home's value. It empowered people with information and fundamentally changed the dynamics in the real estate industry. We've worked over the last 13 years to improve it, from adding new features and more data sources to enabling homeowners to edit their home facts. Thanks to these improvements over time, I'm proud to say that the Zestimate now has a median error rate of less than 2 percent for homes listed for sale, meaning half of all Zestimate fall within 2 percent of the home's eventual sales price.
- Zillow uses statistical and machine learning algorithm to predict the true valuation of the property through combining and understanding multiple data points and key features of the house.

Objective

The goal of our project is to create an automated valuation model to harness data to produce an estimate of a property's market value — where it would transact between a willing buyer and seller, at arm's length, without compulsion.

The aim of the project can be broken down to the following aspects:

- Understand the business implication
- Deep dive into data characteristics
- Univariate and bivariate analysis
- Machine Leaning Models
- Inferences for meaningful insights

Data Description

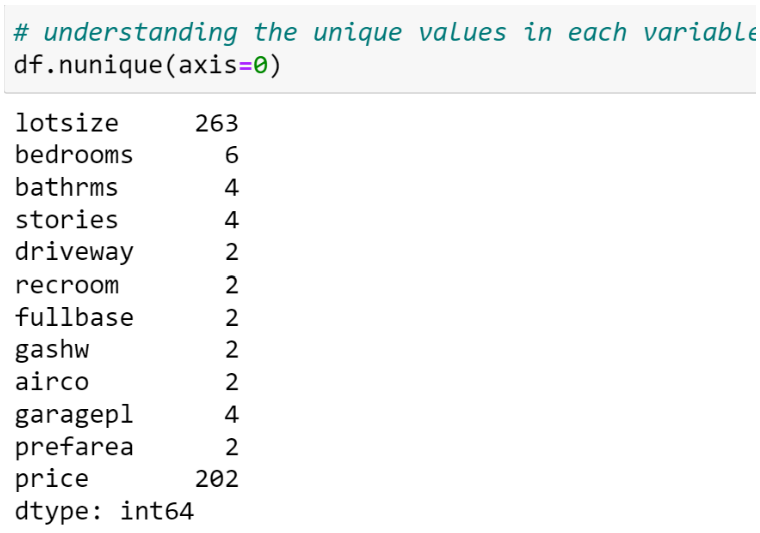
- We are using a real estate sample dataset to evaluate a property price and provide a correct valuation for the market based the characteristics of the property
- The data has 500 rows and 12 columns and includes key features of the property as shown:
    - lotsize: The size of the lot
    - bedrooms: Number of bedrooms in the property
    - bathrms: Number of bathrooms in the property
    - driveway: Driveway existence
    - recroom: Recreational area existence
    - fullbase: Full basement presence
    - gashw: Gas and Heat control inclusion
    - airco: Air Conditioner presence
    - garagepl: Number of Garage Place
    - prefarea: If the property is located in the preferred area
    - stories: Number of Stories in the unit
    - price: Actual valuation of the property

Observations:

- lotsize, bedrooms, bathrms, stories, garagepl, price are numerical variables since they continuous in nature
- driveway, recroom, fullbase, gashw, airco, prefarea are categorical variables since they have two level yes or no
- No null values in the dataset was detected using isna function of pandas and summing up the null values in each variable
- To explore further in each variable we looked towards understanding the unique values in our dataset

Exploratory Data Analysis

Understanding the number of unique levels of all the variables in the data:



```
# understanding the unique values in each variabl
df.nunique(axis=0)
```

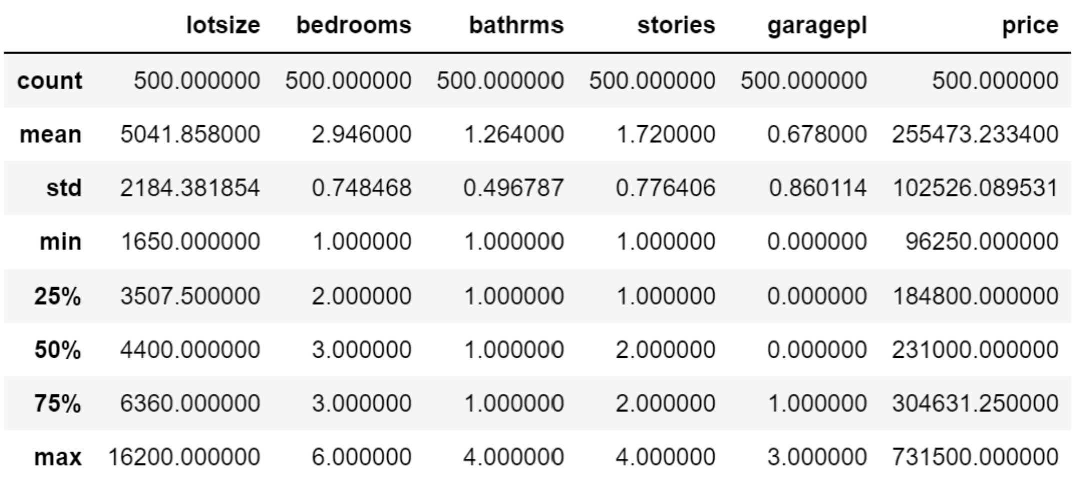
```
lotsize     263
bedrooms      6
bathrms       4
stories       4
driveway      2
recroom       2
fullbase      2
gashw         2
airco         2
garagepl      4
prefarea      2
price       202
dtype: int64
```

The observation can be rendered moot for lot size and price. But for other numerical variables, the observation is noteworthy. The number of bedrooms range from 1 through 6, the number of bathrooms range from 1 through 4, there are up to 1 through 4 stories in a unit, the number of garage is 0 through 3.

For categorical variables, the levels are 2- yes and no, which have been changed to 1 and 0 respectively.

Univariate Analysis

The following table shows the preliminary descriptive statistics for numeric variables in the dataset. **.describe()** summarizes the count, mean, standard deviation, min, and max for numeric variables.



	lotsize	bedrooms	bathrms	stories	garagepl	price
count	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000
mean	5041.858000	2.946000	1.264000	1.720000	0.678000	255473.233400
std	2184.381854	0.748468	0.496787	0.776406	0.860114	102526.089531
min	1650.000000	1.000000	1.000000	1.000000	0.000000	96250.000000
25%	3507.500000	2.000000	1.000000	1.000000	0.000000	184800.000000
50%	4400.000000	3.000000	1.000000	2.000000	0.000000	231000.000000
75%	6360.000000	3.000000	1.000000	2.000000	1.000000	304631.250000
max	16200.000000	6.000000	4.000000	4.000000	3.000000	731500.000000

Observations:

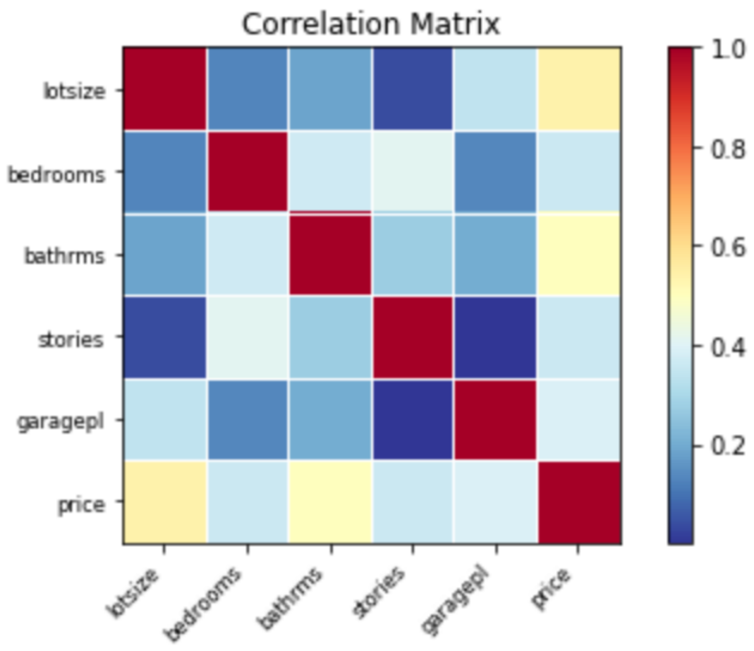
- Average size of the properties is 5041 squared units. Average price valuation of the properties is $255,473.2334.
- The smallest property has the lot size of 1650 squared units and the largest property has the lot size of 16,200 squared units.
- Similarly, the minimum price valuation of a property is $96,250 and the maximum is $731500.
- On average, there are around 3 bedrooms, 1 bathroom, 2 stories in a property.

Bivariate Analysis

- Correlation Matrix

  The first thing we did for bivariate relationship is to do when analyzing variables is visualizing it through a correlation matrix because it's the fastest way to develop a general understanding of **all** of variables. To review, **correlation** is a measurement that describes the relationship between two variables Thus, a **correlation matrix** is a table that shows the correlation coefficients between many variables.

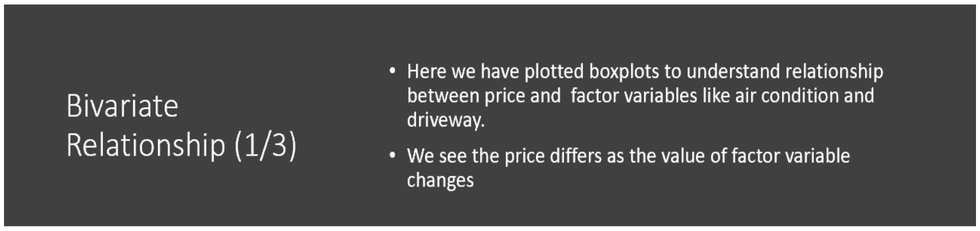
```
fig=plot_corr(corr,xnames=corr.columns)
```



Observations:

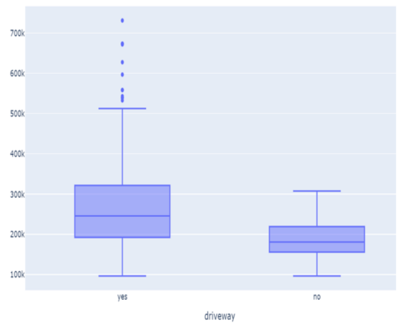
- A high correlation between dependent and independent variables is desired whereas the high correlation between 2 independent variables is undesired.
- Here we see that price which our dependent variable has a higher correlation with lot size and bathroom

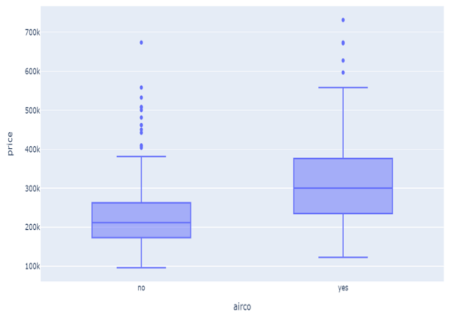
- <u>Box plot</u>

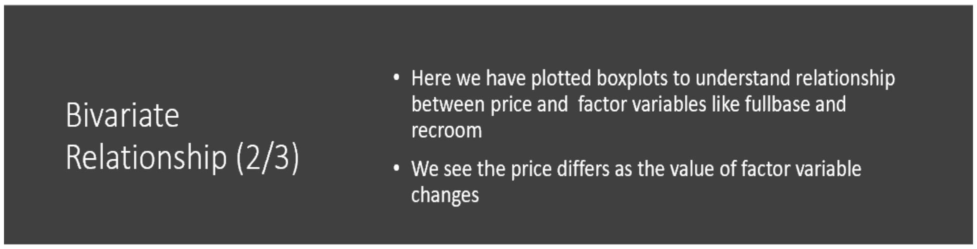


Bivariate Relationship (1/3)

- Here we have plotted boxplots to understand relationship between price and factor variables like air condition and driveway.
- We see the price differs as the value of factor variable changes

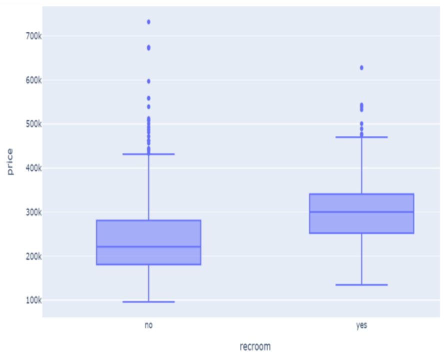


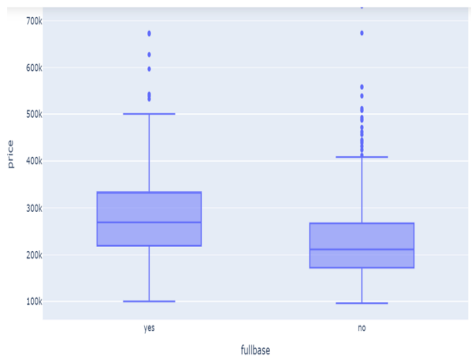




Bivariate Relationship (2/3)

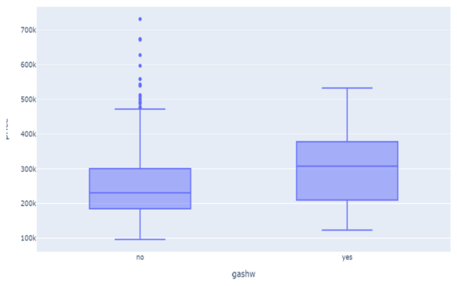
- Here we have plotted boxplots to understand relationship between price and factor variables like fullbase and recroom
- We see the price differs as the value of factor variable changes

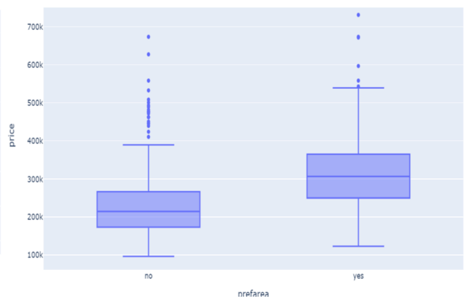




Bivariate Relationship (3/3)

- Here we have plotted boxplots to understand relationship between price and factor variables like gashw and prefarea
- We see the price differs as the value of factor variable changes





Machine Learning Models

There are two kinds of machine learning models, in general: supervised and unsupervised. Supervised models have the presence of a target variable, whereas unsupervised models do not. This problem is prediction of property valuation, based on historical data which are properties with their respective valuation. Hence, the presence of target variable makes it a supervised learning algorithm.

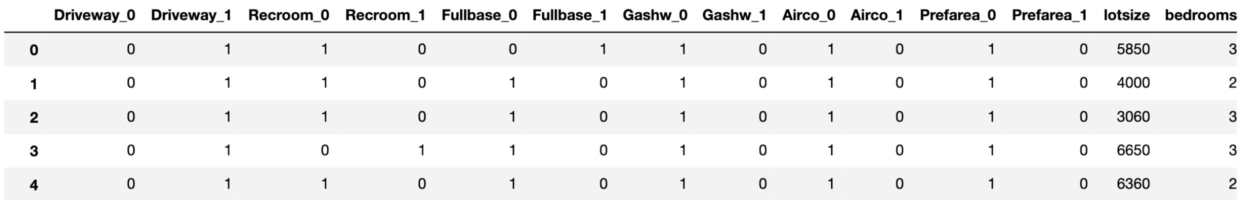
Supervised learning algorithms are of broadly two types: classification and regression. Classification algorithms have a categorical target whereas regression algorithms have a continuous target. Since price is a continuous float variable, this is a regression problem.

Model 1: Multivariate Linear Regression

Linear regression is a model based on exponential family and assumes normal distribution of standard errors. The loss function is the mean square difference between the actual and predicted values of the target variable and is optimized using gradient descent algorithm. The predicted values can be determine as a linear relationship to the matrix of independent variables. The linear relationship has one intercept and coefficients for each variable. the coefficients determine the degree of effect of the variables in determining the fit. The p-value determines whether they affect or do not affect the model fit.

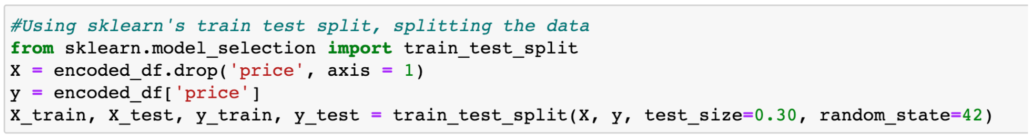
Steps taken in the algorithm:

- One hot encoding the categorical data



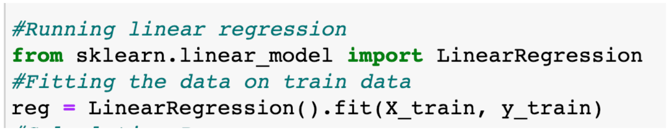
	Driveway_0	Driveway_1	Recroom_0	Recroom_1	Fullbase_0	Fullbase_1	Gashw_0	Gashw_1	Airco_0	Airco_1	Prefarea_0	Prefarea_1	lotsize	bedrooms
0	0	1	1	0	0	1	1	0	1	0	1	0	5850	3
1	0	1	1	0	1	0	1	0	1	0	1	0	4000	2
2	0	1	1	0	1	0	1	0	1	0	1	0	3060	3
3	0	1	0	1	1	0	1	0	1	0	1	0	6650	3
4	0	1	1	0	1	0	1	0	1	0	1	0	6360	2

- Splitting the data into 70-30 train and test:



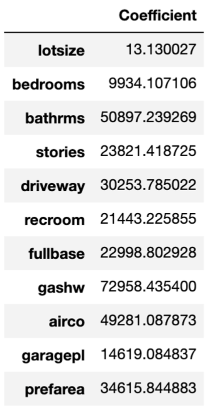
```
#Using sklearn's train test split, splitting the data
from sklearn.model_selection import train_test_split
X = encoded_df.drop('price', axis = 1)
y = encoded_df['price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=42)
```

- Fitting the model on train data:

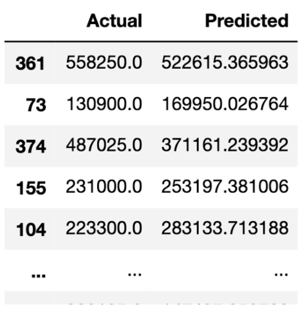


```
#Running linear regression
from sklearn.linear_model import LinearRegression
#Fitting the data on train data
reg = LinearRegression().fit(X_train, y_train)
```

- Coefficient interpretation: Example: 1 unit increase in lot size increases the property valuation by 13.130027$. Prediction on test data and comparing the actual and predicted:

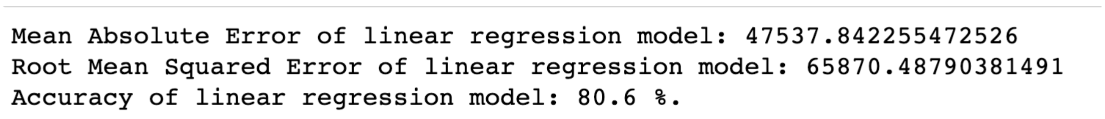


	Coefficient
lotsize	13.130027
bedrooms	9934.107106
bathrms	50897.239269
stories	23821.418725
driveway	30253.785022
recroom	21443.225855
fullbase	22998.802928
gashw	72958.435400
airco	49281.087873
garagepl	14619.084837
prefarea	34615.844883



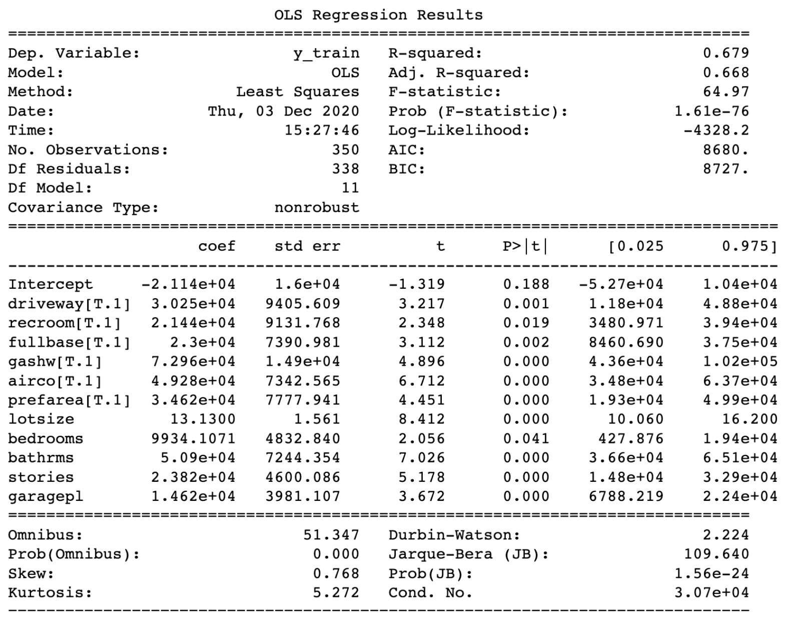
	Actual	Predicted
361	558250.0	522615.365963
73	130900.0	169950.026764
374	487025.0	371161.239392
155	231000.0	253197.381006
104	223300.0	283133.713188
...	...	...

- Evaluating mean absolute error, root mean square error and accuracy:



```
Mean Absolute Error of linear regression model: 47537.842255472526
Root Mean Squared Error of linear regression model: 65870.48790381491
Accuracy of linear regression model: 80.6 %.
```

- P-values, Adjusted R squares and Standard error:



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                y_train   R-squared:                       0.679
Model:                            OLS   Adj. R-squared:                  0.668
Method:                 Least Squares   F-statistic:                     64.97
Date:                Thu, 03 Dec 2020   Prob (F-statistic):           1.61e-76
Time:                        15:27:46   Log-Likelihood:                -4328.2
No. Observations:                 350   AIC:                             8680.
Df Residuals:                     338   BIC:                             8727.
Df Model:                          11
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
Intercept     -2.114e+04     1.6e+04     -1.319      0.188   -5.27e+04    1.04e+04
driveway[T.1]  3.025e+04   9405.609      3.217      0.001    1.18e+04    4.88e+04
recroom[T.1]   2.144e+04   9131.768      2.348      0.019    3480.971    3.94e+04
fullbase[T.1]    2.3e+04   7390.981      3.112      0.002    8460.690    3.75e+04
gashw[T.1]     7.296e+04   1.49e+04      4.896      0.000    4.36e+04    1.02e+05
airco[T.1]     4.928e+04   7342.565      6.712      0.000    3.48e+04    6.37e+04
prefarea[T.1]  3.462e+04   7777.941      4.451      0.000    1.93e+04    4.99e+04
lotsize          13.1300      1.561      8.412      0.000      10.060      16.200
bedrooms       9934.1071   4832.840      2.056      0.041     427.876    1.94e+04
bathrms          5.09e+04   7244.354      7.026      0.000    3.66e+04    6.51e+04
stories        2.382e+04   4600.086      5.178      0.000    1.48e+04    3.29e+04
garagepl       1.462e+04   3981.107      3.672      0.000    6788.219    2.24e+04
==============================================================================
Omnibus:                       51.347   Durbin-Watson:                   2.224
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              109.640
Skew:                           0.768   Prob(JB):                     1.56e-24
Kurtosis:                       5.272   Cond. No.                     3.07e+04
------------------------------------------------------------------------------
```

<u>Observations:</u>

- Based on adjusted R squared: The model can explain 66.8% of the total variance in the target variable
- All the p-values are less than 0.05, indication the influence of all the variables

Model 2: Random Forest Regressor

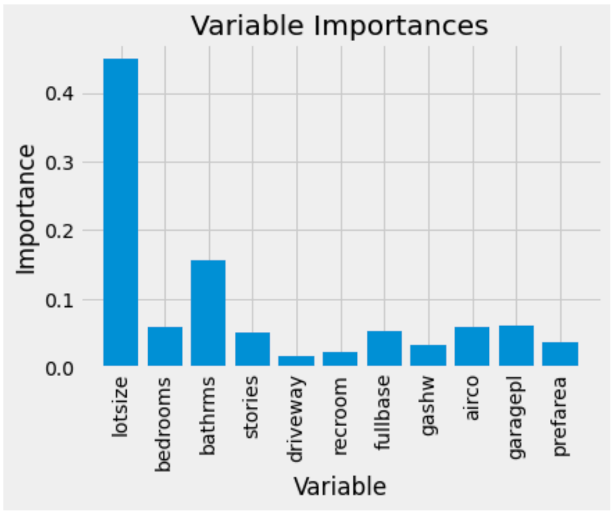
Random Forest is an ensemble model which bootstrap aggregates multiple decision trees to ensure the decrease in variance of individual decision trees. The aggregated model determines the target variable on the basis of weighted average of prediction of each tree. The number of trees argument has to be passed while fitting the model, which can be arbitrary. The higher the trees, the more computation time.

Steps taken in the algorithm:

- On the same train-test split, fit a random forest regressor:

```
#Random Forest with 1000 trees
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(n_estimators = 1000, random_state = 42)
# Train the model on training data
rf.fit(X_train, y_train)
```

- Visualize variable importance of the fit:



It can be seen that lot size has the highest importance for predicting the property price followed by the number of bathrooms, bedrooms and garage place.

- Calculate mean absolute error and accuracy:

```
Mean Absolute Error of random forest regressor is: 47944.03 degrees.
 Accuracy of random forest regressor is: 80.8 %.
```

Observations:

- The model performed better than linear regression with an accuracy of 80.2% which is over 0.2% than the linear regression model
- This model tells us the importance of variables in the fit, which is of high value

Model 3: Lasso Regression

Lasso regression is a way of regularizing the linear models. Regularization is a process that balances the trade of between a high accuracy and a good variance. For linear regression, the weights and loss function are balanced using the regularizers.

Steps taken in the algorithm:

- On the same train-test split, fit a lasso regressor:

```
#Lasso Regression
from sklearn import linear_model
lasso_reg = linear_model.Lasso(alpha= 0.05)
lasso_reg.fit(X, y)
```

- Understand the coefficients of the fit:

```
[-24409.40808615      0.          -16813.53394569      0.
 -22918.7643637       0.          -51361.15885068      0.
 -46908.35492088      0.          -36703.48673715      0.
     14.16062912   8007.49023737   55048.825448     24095.27838134
  16055.31077508]
```

  Same interpretation as of linear regression

- Intercept of the fit:

```
#Intercept
print(lasso_reg.intercept_)
```

```
180607.2642943912
```

- Use the fitted model to make predictions on the test data and compare your results with actual test data. The following is the accuracy metric:

```
#Test error computation
y_pred = lasso_reg.predict(X_test)
errors = abs(y_pred - y_test)
mape = 100 * (errors / y_test)
# Calculate and display accuracy
accuracy = 100 - np.mean(mape)
```

```
accuracy
```

```
80.89643874780093
```

Observations:

- Due to the regularizers, the accuracy has increased to 80.89%. It is better than the random forest by 0.08% and the linear regression model by 0.29%
- The same test accuracy was obtained with different alpha values for the fit.

Conclusion:

- The independent variables have multicollinearity amongst them as number of bedrooms is usually directly proportional to number of bathrooms
- Lot size and number of bedrooms are the most significant while calculating property valuation
- The price of the property increases with increase in bedrooms, bathrooms, air conditioners, garage areas etc.
- The accuracy of the models are as follows: linear regression with 80.6%, random forest with 80.8% and lasso regression with 80.89%.
- The mean absolute error is around 40k, i.e. the error while estimating property valuation is +/- 40k of the actual price
- Tuning the hyper parameters and applying advanced algorithms might be helpful for the Zestimate

Libraries used in the project:

- numpy: for mathematical operations
- pandas: for manipulating data frames
- plotly and matplotlib: to make visualization plots
- sklearn: for model fitting, predicting and evaluating
- statsmodel: to understand the underlying statistics of linear regression