House Price Prediction using multivariable Regression

Gaurao Diwase
Machine Learning Intern
https://www.ai-techsystems.com
Nagpur,India
diwasegaurao99@gmail.com

***Abstract*—Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But, as it is evident now that a lot more features than above mentioned features influence price negotiations.**

This paper provides detailed report on the methodology used to predict house prices using 79 explanatory variables describing (almost) every aspect of residential homes in Ames, lowa.

The dataset used for the purpose was taken from:
https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

Introduction

This report is based on work predicting the house prices at Ames, Lowa. The goal for this project is to use advanced regression techniques in order to estimate the SalePrice of any given house in Ames, Lowa given the feature and pricing data for around 3,000 houses sold.I have used Multinomial Regression Model.

About The DataSet

(My dataset was taken from Kaggle competition for House Prices: Advanced Regression Techniques)

1. File Description:
   - train.csv - the training set
   - test.csv - the test set
   - data_description.txt - full description of each column, originally prepared by Dean De Cock but lightly edited to match the column names used here.
2. Data Fields:
   - Data fields are well described at: https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

Methodology

The term **Methodology** refers to a system of methods used in a particular area of study or activity.So what does this term signifies in a Data Science model.? the Data Science Methodology aims to answer basic questions in a prescribed sequence.
First,we have to identify the problem that we have to solve.In this case, we are required to predict house prices using the given dataset.

Competition Description

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

With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

Then ,we need to analyse the data and decide how we can use the data to meet our demand.Also, we identify the source of data and how it has been acquired. In this very case , it is from the houses sold in a particular timespan at Ames.
Now, after all the above steps performed ,Now it is time to bring the theory to Practical.Let us take a look at the model.

I will use multivariable Regression Model to predict the house prices using various features described in the dataset.
Similar to multiple linear regression, the multinomial regression is a predictive analysis. Multinomial regression is used to explain the relationship between one nominal dependent variable and one or more independent variables.

Preparing the model.

1. Import the libraries and create Environment:
   The tools used for the project are:

- Pandas- a very powerful library providing Easy to use data structures and various data analysis tools
- numpy - provides advanced math functions
- seaborn- provides a high-level interface for drawing attractive and informative statistical graphics.
- Matplotlib - helps with data analyzing, and is a numerical plotting library.

- scikit learn - simple and efficient tools for data mining and data-analysis.

2. The Data:

Our data comes from a Kaggle competition named "House Prices: Advanced Regression Techniques". It contains 1460 training data points and 80 features that might help us predict the selling price of a house.

3. Load the Data:

Import the train dataset and print the shape and first five rows of the dataset.This can be done using `pd.read_csv()` and `pd.head()`.

4. Exploring the Target variable:

We're going to predict the SalePrice column ($ USD), let's explore it.
First, take a look at all the features using `pd.columns()` method.
Then , getting all the effective information of the SalePrice using `pd.describe()` method.
Plot the histogram of SalePrice using `plt.hist()`.Then, taking a look at the skewness of the plot , we see that it is positively skewed .
Then to make this more linear we take log values of the SalePrice using `np.log()`.Now,again plotting the histogram we see that it is more linear and skewness is near to zero .Now , we will need these values further also , so I will add one more column into the pandas dataframe named 'log_sp' indicating the logarithmic value of SalePrice.

5. Feature engineering:

This step involves the art and science of transforming raw data into features that better represent a pattern to the learning algorithms.

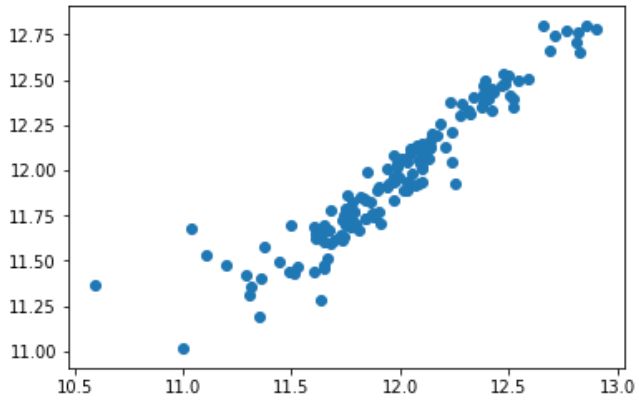
- Now, we start by deleting the Id column,as it has absolutely no impact on the variation of SalePrice .This can be done using `del []` keyword.
- Since, SalePrice being a numeric feature it is best correlated to other numeric features in the dataset.
- Now, selecting the features that have numeric datatype. This can be done using `tr.select_dtypes(include=[np.number])` method.
- We get 38 features that we may find very deeply impacting the variations of SalePrice.
- Constructing the correlation matrix for the same, using `corr()` function.
- But ,this is a humongous matrix to take a look at . Let us plot a heatmap for the correlation matrix using functions from ***seaborn library***.
- This can be done using `sns.heatmap()` function.Thus ,taking a keen look at the above heat map ,we can easily spot that SalePrice is most affected by 'OverallQual' ,'GrLivArea',etc.
- Now visualising most dependent feature one by one and removing outliers to clean our data .
- Now, plotting the Scatter Plot to visualize the relationship between 'SalePrice' and 'OverallQual'. Also ,Note that there are not much outliers ,so no need of manipulating the data frame. This was done using `tr.plot.scatter()` function.
- Similarly , we can perform analysis on GrLivArea .The next most impacting feature is GarageArea.
- Visualising the scatter plot between SalePrice and GarageArea, we see that for *GarageArea > 1200* can be considered as outlier , so removing these rows and moving forward.
- **Dealing with the missing Values:** We can take a look at the count at number of missing values in each column and getting them arranged according to the sum from highest to lowest number.This can be done using `tr.isnull().sum()` function.
- I infer that ,most of the missing values are from columns that are object data type . Therefore , it does not bother our model as we are considering only those features that are numeric . So, this does not bother our model.

6. Making actual **multivariable Regression Model:**
   - ➔ We first transform our dataframe by dropping and filling NaN values. This can be achieved using `interpolate()` and `dropna()` methods .
   - ➔ Divide the data frame into 2 parts , one:The *target* variable and two: the *variable features*.
   - ➔ Generally , **x** is called the **"predictor"** columns and **y** is called the **"response"** . Therefore , x is assigned all the numeric values after dropping SalePrice(as we are going to predict it) and log_sp(This was the column that stored log value of SalePrice).
   - ➔ Split the data into train and test : The train test split consists in randomly making 2 subsets of the data: the t*raining set*, used to fit our learning algorithm so it learns how to predict, and the t*est set* which we use to get an idea of how the model would perform with the new data . This can be done using `train_test_split()` function.The attributes are: x, y ,`test_size=0.1` (this assign the 10% of data set to test), `random_state=1` (this is used for initializing the internal random number generator).
   - ➔ Fit the model on the training set :This is done by instantiating LinearRegression from sklearn.Linea_model as **lm** and fit using `lm.fit(x_train,y_train)`.
   - ➔ Now , make predictions using `predict()` method.
   - ➔ Print the co-efficients:1)**Intercept** : The intercept can be found using df.intercept__ function.
     2)**Constants**: The regression model has to find the most optimal coefficients for all the attributes. To see what coefficients our regression model has chosen,we can use the following line of code.
     `pd.DataFrame(lm.coef_, x.columns, columns=['Coefficient'])`

7. Checking the correctness of model:

We plot the actual and predicted house prices using the

scattered plot. It can be depicted as below:



This figure is plotted considering the logarithmic values of the SalePrice.
The table below shows snippet of the actual and predicted value :

	Actual	Predicted
1379	12.028739	11.888085
1359	12.660328	12.801091
1094	11.767568	11.783847
920	12.211060	12.126778
37	11.938193	11.907590
1228	12.813918	12.703601
1274	11.842229	11.731078
91	11.498827	11.692595

We can easily note that the predicted values are quite closer to the predicted value , thus suggesting that our model has done a pretty decent job.The random values for Id's is because of the random_state attribute we used in the `train_test_split()`.
Model evaluation using **Estimator score method**:
Estimators have a score method providing a default evaluation criterion for the problem they are designed to solve.
`lm.score(x_test,y_test)` = **0.8929964019904535.**
Thus , we got about ***89.23 %*** accuracy in our results which is very good.

The mean squared error (MSE) or mean squared deviation (MSD) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors—that is, the average squared difference between the estimated values and the actual value.This could be evaluated using `mean_squared_error()` function. For our model, it is **0.01728006094326084** which suggests that our model is predicting the values quite closer to that of actual price.

7) Creating the submission :
   - ➢ First , read the test csv file and print top five rows of the table. Note that,there are only 80 columns in this dataframe as the 81'st column is SalePrice that we have to predict.
   - ➢ Create a new dataframe and add Id column to it .
   - ➢ Repeat, all the steps that we performed on train dataset to modify the test dataset .
   - ➢ Then predict the SalePrice using `predict()` method.
   - ➢ Remember that we predicted the logarithmic value of SalePrice , this has to be converted back into actual values using `np.exp()` function.
   - ➢ Now, add another column SalePrice to the freshly created DataFrame.
   - ➢ Convert the submission file into csv file using df.`to_csv()` method.
   - ➢ Submission is ready.
   - ➢ snippet of submission:

	Id	SalePrice
0	1461	124929.869831
1	1462	141017.659717
2	1463	173045.235633
3	1464	196259.155367
4	1465	179804.872662

Conclusion

In this project , successful prediction of House Price was made using multivariable regression model. The efficiency of the model could have been increased using methods like using Gradient Boost. Also, other Machine Learning algorithms could be used to meet the same need. It was tried that the model be kept as simple as possible without hampering the efficiency of model.It was found that , the model performed really well with the test data set and the submission.csv was successfully generated.

Acknowledgment

The Ames Housing dataset was compiled by Dean De Cock for use in data science education. It's an incredible alternative for data scientists looking for a modernized and expanded version of the often cited Boston Housing dataset.

References

[1] https://www.kaggle.com/c/house-prices-advanced-regression-techniques/overview
[2] https://scikit-learn.org/stable/index.html
[3] https://seaborn.pydata.org/
[4] https://pandas.pydata.org/
[5] https://numpy.pydata.org/
[6] https://www.kdnuggets.com/2018/12/six-steps-master-machine-learning-data-preparation.html
[7] www.stackexchange.com
[8] encyclopedia
[9] www.geeksforgeeks.com