CS521 C1 Information Structures with Python (Summer 1 2022)

GitHub Repository: final-project-FinnG13

Names: Finn Graham, Khudeja Begum, Jiehua Liu

Final Project: Data Analysis with Python

Predicting Housing Prices (Linear Regression)

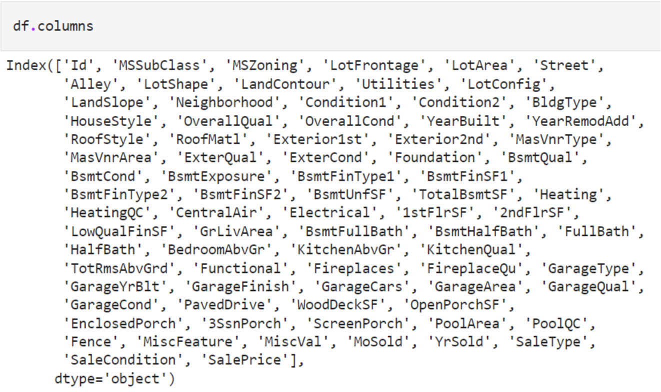
Due Date: August 12, 2022

Introduction

For our project, we chose to analyze housing prices, with the end goal of finding the best methods to predict the price of a given house. We were able to work with just a single, large data set containing columns for sale prices and a number of housing attributes and features, which we could use to run our analysis. Our first steps were to simply move our csv data into a python data frame for analysis and begin exploration. We experimented with the python packages pandas and sklearn to create multiple visualizations in order to simply get a better grasp on the data we had to work with. Once we were familiar with our dataset, we could clean out the data, remove all the visuals we used to experiment, and run a regression on our data set. This report will outline the important steps we implemented, but to see the process in more detail, reference the file “house_price.ipynb” in our github repository.

Our Data

The data file we used was csv file titled ‘House Prediction Data.csv’. The initial data contained 2919 rows and 81 columns. The large number of columns gave us room to test the correlation of numerous variables to house price. For reference, here is a list of columns we had at our disposal:

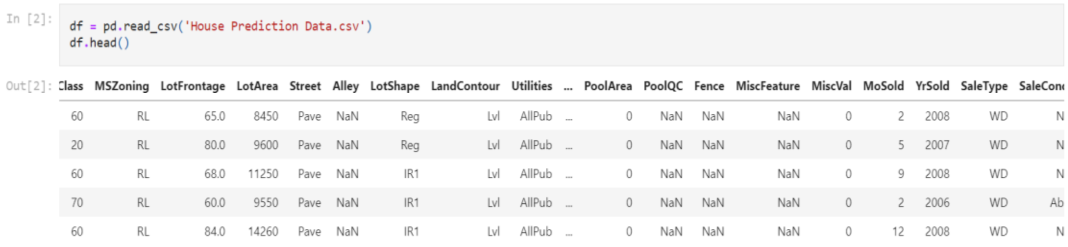


```
df.columns
```

```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```

Loading the Data

We used the python package “pandas” to load our csv into a python data frame that could be operated on.



In [2]:

```
df = pd.read_csv('House Prediction Data.csv')
df.head()
```

Out[2]:

Class	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	SaleType	SaleCon
60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	2	2008	WD	N
20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	5	2007	WD	N
60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	9	2008	WD	N
70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	2	2006	WD	Ab
60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	12	2008	WD	N

Cleaning the Data

Our initial plan to clean the data was to simply remove columns that we thought would have the least effect in determining housing prices. However, when working with a dataset of only 2919 rows, removing entire columns of data would not be required to improve the overall efficiency of our operations. Therefore, we simply chose to remove null values from our data as so:

1. Sorted all columns by number of null entries
2. Found that 6 columns had over 1000 null entries
3. Programmatically dropped the null values from all columns we saw the need

Data Exploration / Initial Approach

Throughout our process of learning and experimenting with pandas and sklearn, we created a wide variety of visuals and performed numerous operations on our data. Although most of this work was cut out of our final project, it was crucial in developing an understanding of what exactly we could do with our data.

Additionally, we performed research to find python packages that may be useful to us in our analysis. Here are all the packages we ultimately chose to include in our project:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
```

Data Exploratory Analysis – Extension 1: Correlation

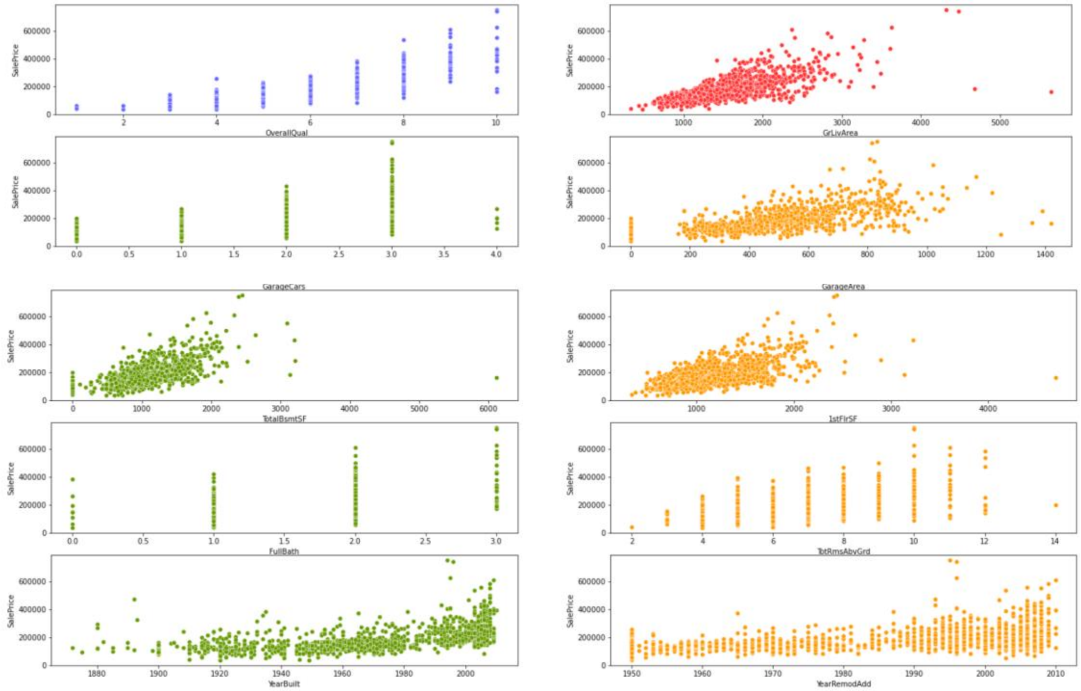
One key realization we made early on in our process was that data types differed across all of our columns. Some of the data was in numerical form (prices, areas, ratings) and some of it was categorical (typically a word indicating the presence of some amenity). In order to run a more thorough analysis, we chose to split our dataset into a numerical and categorical set.

Numerical Data:

- We used a built-in pandas method to sort all of our columns by their correlation coefficients in relation to Sale Price.
- 10 of our columns had correlation coefficients greater than 0.5, and therefore, we chose to move on with these columns for a visual analysis.

```
o  SalePrice        1.000000
o  OverallQual      0.790982
o  GrLivArea        0.708624
o  GarageCars       0.640409
o  GarageArea       0.623431
o  TotalBsmtSF      0.613581
o  1stFlrSF         0.605852
o  FullBath         0.560664
o  TotRmsAbvGrd     0.533723
o  YearBuilt        0.522897
o  YearRemodAdd     0.507101
```

- We then created scatter plots for those columns in relation to Sale Price.



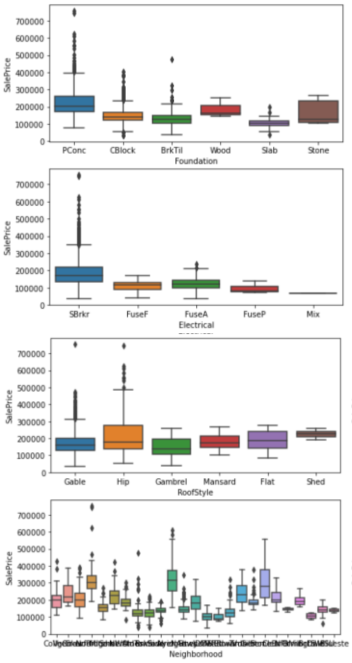
At first glance, we can see that some of the numerical columns with smaller ranges (x-axis) are likely counts (ex. Count of cars in garage or full bathrooms in the home), denoted by the graphs with vertical bars. Although they are a bit more difficult to read and predict, these graphs still show a positive correlation. Based upon the correlation coefficients and subsequent graphs, we can say with confidence that as [OverallQual, GrLivArea, GarageCars, GarageArea, TotalBsmtSF, 1stFlrSF, FullBath, TotRmsAbvGrd, YearBuilt, YearRemodAdd] increase, the Sale Price of a given house will increase.

Categorical Data:

- We performed an encoding operation to find the correlation coefficients for our categorical columns.
- 8 of our columns had correlation coefficients greater than 0.15, and therefore, we chose to move on with these columns for a visual analysis.

```
o  SalePrice        1.000000
o  Foundation       0.382479
o  CentralAir       0.251328
o  Electrical       0.234945
o  PavedDrive       0.231357
o  RoofStyle        0.222405
o  SaleCondition    0.213092
o  Neighborhood     0.210851
o  HouseStyle       0.180163
```

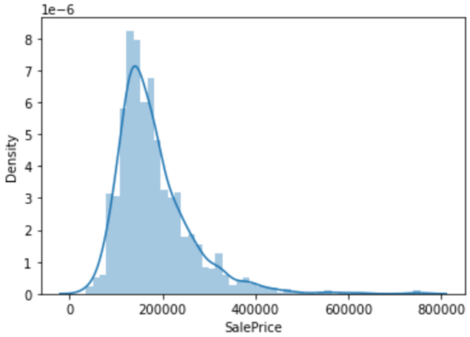
- We decided that box plots would best convey categorical columns in relation to Sale Price



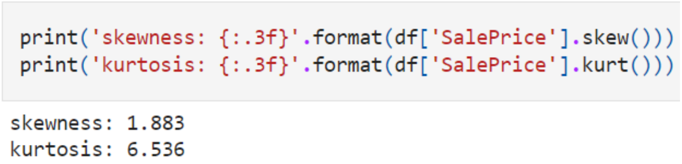
In general, we found that the categorical data is less effective in predicting housing prices, while we can’t say that there is no correlation entirely.

Data Exploratory Analysis – Extension 2: Distribution

Next, we hoped to learn house housing prices are distributed. We were curious if we would see a normal distribution or a skewed distribution. Using a python package called “seaborn”, we were able to create a distribution plot to get a rough idea.



The distribution was visibly skewed to the right. Therefore, we used built-in pandas operations to verify this observation.

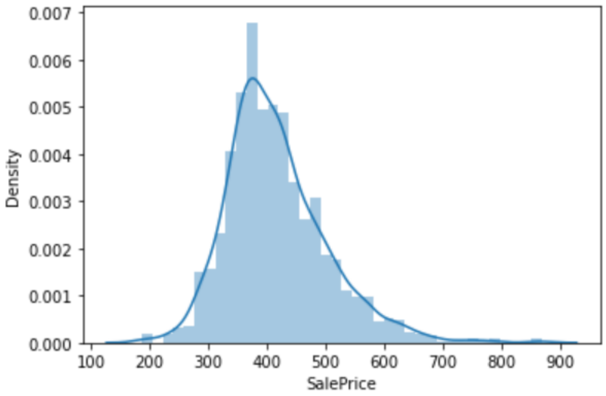


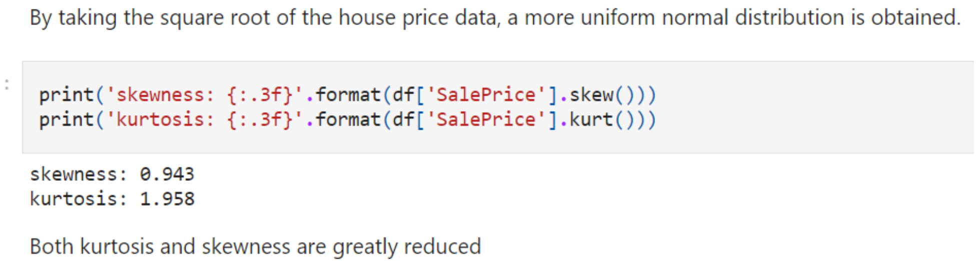
```
print('skewness: {:.3f}'.format(df['SalePrice'].skew()))
print('kurtosis: {:.3f}'.format(df['SalePrice'].kurt()))
```

```
skewness: 1.883
kurtosis: 6.536
```

Kurtosis is a statistic that describes the steepness of the distribution of all values of a variable; the kurtosis is greater than 0, which is steeper than the peak of the normal distribution. Skewness is a statistic that describes the symmetry of the value distribution of a variable; if the value is greater than 0, the positive deviation value is large, which is positive or right-skewed, and the long tail is dragging to the right.

Ultimately, we realized the following:





By taking the square root of the house price data, a more uniform normal distribution is obtained.

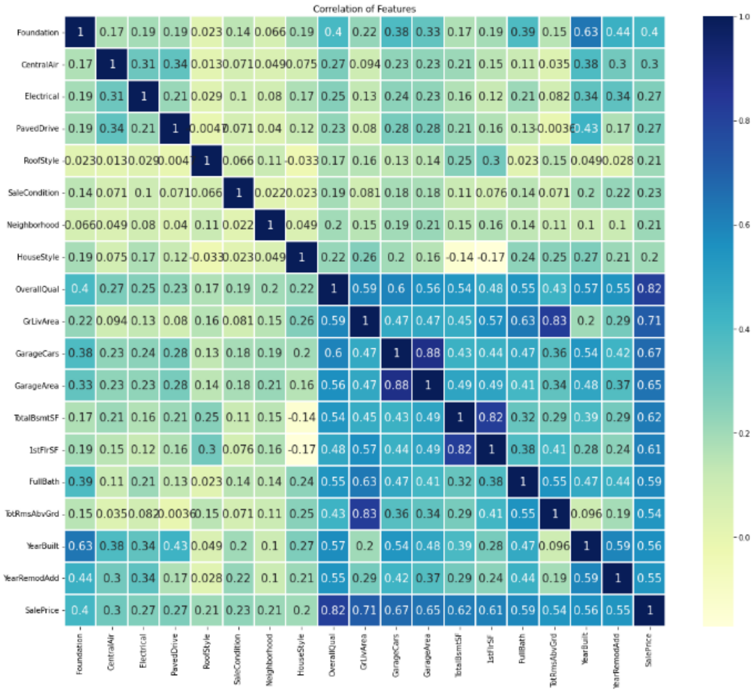
```
print('skewness: {:.3f}'.format(df['SalePrice'].skew()))
print('kurtosis: {:.3f}'.format(df['SalePrice'].kurt()))
```

```
skewness: 0.943
kurtosis: 1.958
```

Both kurtosis and skewness are greatly reduced

Data Exploratory Analysis – Extension 3: Correlation Heatmap

Heatmaps are incredibly useful because they allow us to compare across multiple variables to check for correlation. We encoded our categorical data from earlier and created a heat map to see what we could find.

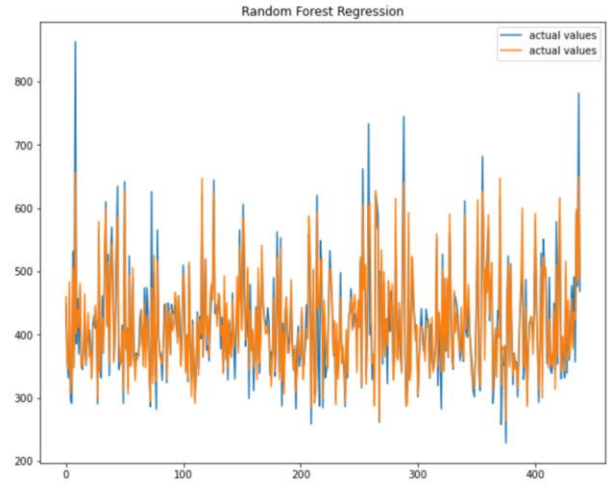


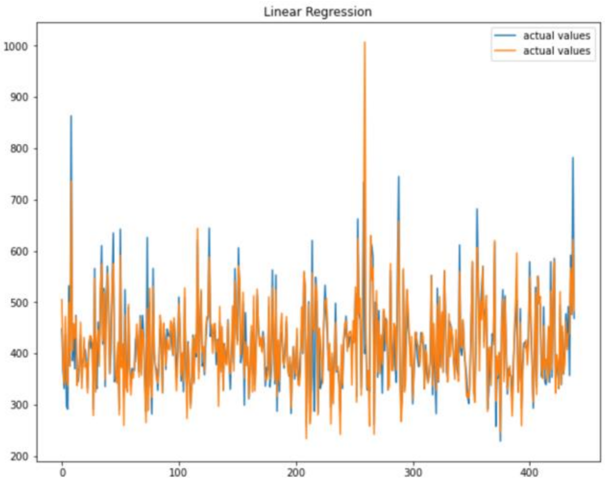
The darker the square, the greater the correlation between the two variables connecting it. For example, we can see again that the highest correlation to Sale Price is OverallQual.

Regression Analysis and Comparisons

Our final step was to run our linear regression using the built in functionality sklearn provides. In order to find the best result possible, we chose to not only run a linear regression, but also Random Forest regression (another sklearn tool recommended by some online sources). We chose to run each regression, and report upon which method had higher prediction ability.

Results:





```
Linear regression model:
RMSE:  42.77243000377719
MSE:  1829.4807684280192
R-squared:  0.7568604062905525

Random forest regression model:
RMSE:  32.810546323354195
MSE:  1076.5319500369717
R-squared:  0.8569279625868191
```

Since the Random forest regression model has a larger R-squared value, the greater proportion of the variance in Sales Price can be explained by the independent variables in the model.

Conclusion

Throughout this process we have learned that python has many tools that can be used effectively in the transformation and analysis of data. Additionally, we learned some of the best practices for running a data regression analysis. We ultimately concluded that housing prices are determined by a number of factors, and cannot be explained by a few simple variables. However, using the variables that we determined had the highest correlation, we were able to determine that a random forest regression model was most effective in predicting housing prices.