MET CS 521 Final Project:
Names: Daniela Demaestri and Alex Mao
GitHub usernames: ddemaest-bu and alexmao0501

**Final Project: Predicting Housing Prices (Regression)**

## Objective

For this final project, we have selected to work with a large housing dataset. The dataset consists of 81 columns and 2919 rows. We started by cleaning and handling the missing data. Once our data was cleaned, we performed some exploratory data analysis to better understand the data and to help better select some relevant variables for modeling. Finally we fitted our data to different regression models in order to predict SalePrice based on different housing features.

Achieving our objectives was split into three phases:

1. Cleaning and Processing the Data: In this phase we handled nulls, encoded some categorical variables, and removed some features to reduce the complexity.
2. Exploratory Data Analysis (EDA): Our main goal in this phase was to have a better understanding of the features involved in our data. This consisted of a non-exhaustive analysis focused on determining the features that have the highest correlation with SalePrice in the hopes that it could later help us produce a model to predict SalePrice.
3. Regression Modeling: Lastly, we implemented regression models to predict SalePrice (target variable) using different housing features.

In order to reproduce results, we have included a script, Script_Download_Data.ipynb, that will download and write a csv file to your working directory that will contain the housing data we utilized for our project. The finalproject.ipynb script will read the csv file into a dataframe using a popular import tool, pandas. The script should be run sequentially in order to reproduce the same results we will be explaining throughout this paper.

## Cleaning and Processing the Data

***Handling null values:*** Our initial focus was to handle null values within the dataset. Not only is this an important step to get a better understanding of the dataset, but also because many machine learning algorithms and models fail when the dataset contains missing values.

We started looking at the null values within our target variable, SalePrice, because our main objective was to predict SalePrice and identify important housing features in relation to SalePrice. There were 1459 null values in SalePrice (49.98% of our data). Given our objective, we decided to drop these nulls.

After removing the rows with null values in SalePrice, we were left with 1460 rows. Now we were interested in seeing how many nulls there were in the remainder of the dataset, shown below:

| | |
|---|---|
| LotFrontage | 259 |
| Alley | 1369 |
| MasVnrType | 8 |
| MasVnrArea | 8 |
| BsmtQual | 37 |
| BsmtCond | 37 |
| BsmtExposure | 38 |
| BsmtFinType1 | 37 |
| BsmtFinType2 | 38 |
| Electrical | 1 |
| FireplaceQu | 690 |
| GarageType | 81 |
| GarageYrBlt | 81 |
| GarageFinish | 81 |
| GarageQual | 81 |
| GarageCond | 81 |
| PoolQC | 1453 |
| Fence | 1179 |
| MiscFeature | 1406 |

We decided to drop the columns that didn't seem interesting or relevant given the high amount of null values. The features we removed included: PoolQC, MiscFeature, Alley, Fence, FireplaceQu, and MiscVal. We also deduced that the reason behind the high quantity of nulls for these columns is due to the fact these features may not be present in all houses. In other words not all houses have pools, alleys, fences, and fireplaces.

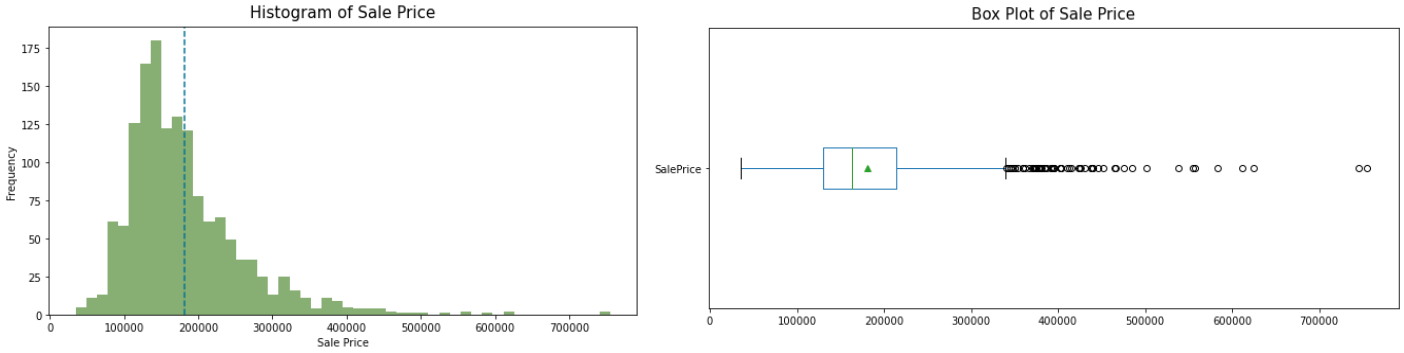
Looking at the data description we also noticed that there were nine features that described and/or rated basements. To limit the scope for basement features, we decided to keep BsmtQual, BsmtCond, BsmtExposure, and TotalBsmtSF as they seemed to indicate a more generalized description of the basement, and decided to drop the other basement features.

Then we handled the remaining null values in the dataset by replacing nulls with the mode for Electrical and MasVnrType, and by replacing the nulls with 0 for MasVnrArea (if MasVnrType was None) and LotFrontage.

***Encoding categorical variables:*** Looking at the data description, a few categorical variables had an implicit order. Hence, we decided to encode these categorical variables as ordered numbers (similar to how OverallCond was already encoded). The variables we manually encoded numerically were: LotShape, Utilities, AllPub, LandSlope, ExterQual, ExterCond, BsmtQual, BsmtCond, HeatingQC, KitchenQual, FireplaceQu, GarageFinish, GarageQual, and GarageCond. Just to showcase one example, we encoded BsmtCond as follows: 5 - Ex, 4 - Gd, 3 - TA, 2 - Fa, 1 - Po, 0 - NA.

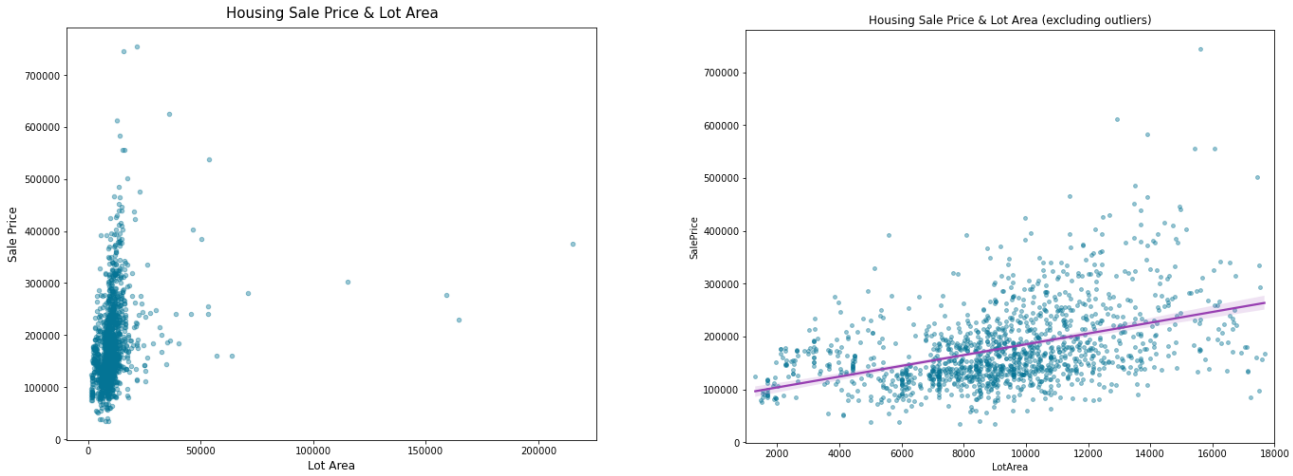
## Exploratory Data Analysis (EDA)

***Taking a closer look at the target variable - SalePrice:*** Looking at the distribution of the target variable, SalePrice, is important especially when it comes to Linear Regression because of the assumptions the model makes: linearity, homoscedasticity, independence, and normal distribution. Below we observe the distribution of SalePrice:




Looking at the distribution of SalePrice (either looking at the histogram or boxplot), we can see the graph is right-skewed.

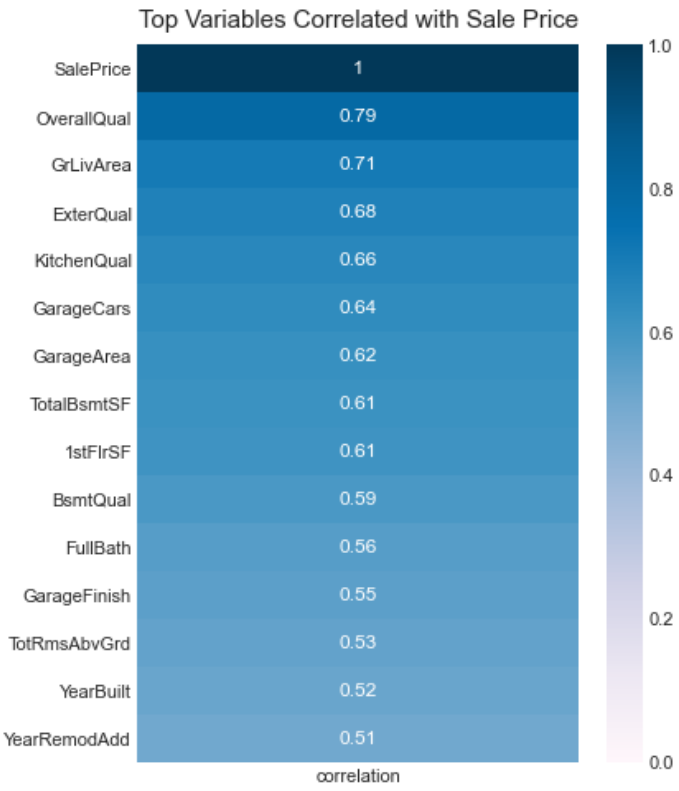
***SalePrice vs. LotArea:*** The next question we wanted to explore was: What is the relationship between SalePrice and LotArea? The reasoning behind our question is due to our assumption that houses on larger lots tend to have a higher price or property value than similar houses on smaller lots. We explore this relationship below:

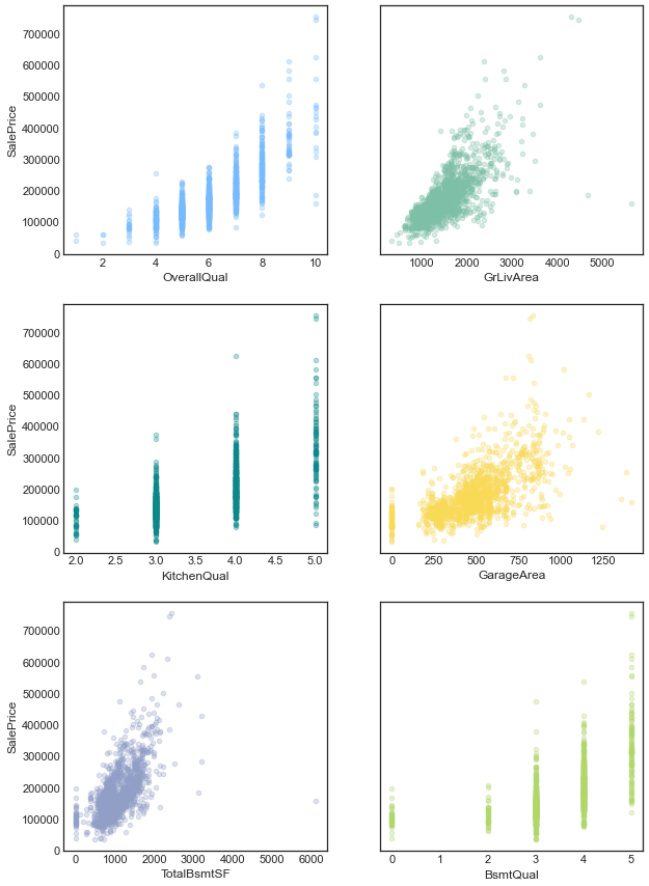



In the left graph, we observe quite a few outliers when it comes to the LotArea. By removing the outliers (right graph), we can take a better look at the relationship between SalePrice and LotArea. To determine the outliers, we used the interquartile range (IQR) approach where we find the upper and lower bound. Outliers are values that are above and below the dataset's normal range (above the upper bound and below the lower bound). When looking at the graph without the outliers, we can observe a positive correlation between SalePrice and LotArea.

**Extension 1: Exploring Correlation**

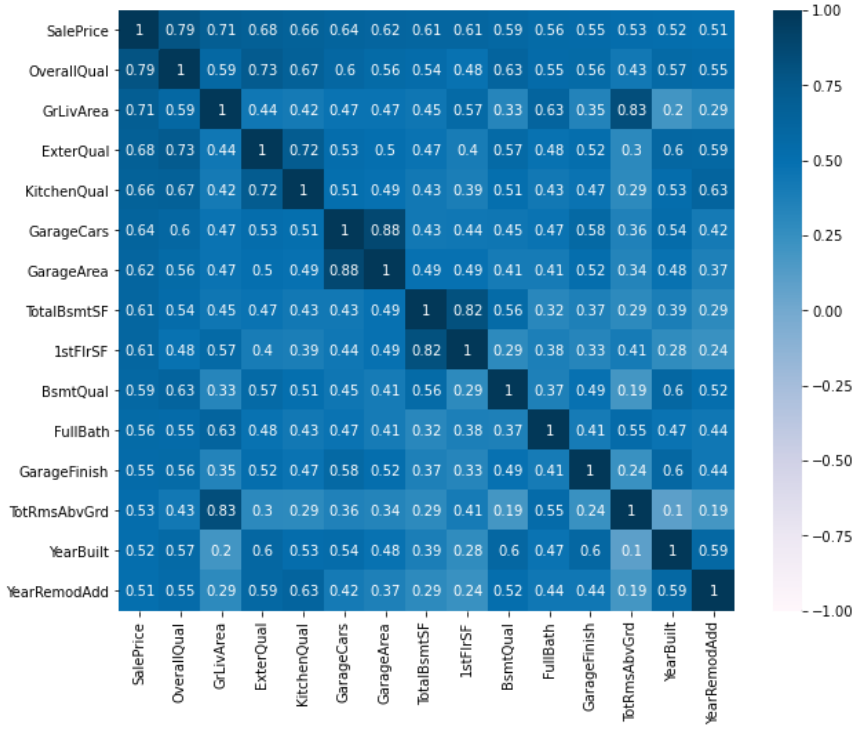
***Which are the top (numerical) housing features that are correlated with SalePrice?*** In order to visualize the top correlated values (left graph) we used seaborn.heatmap and filtered to only visualize the correlation between features and SalePrice with correlation values greater than the absolute value of 0.50 (to include features with both positive and negative correlations). We will use a few of these correlated features in our regression model later in our project. The right graph further visualizes the relationship between some of the top correlated features with SalePrice.




***Looking at multicollinearity between features:***

Looking at only the top correlated features, some housing features seem to be highly correlated to each other (see heatmap on the right). For example: GarageArea and GarageCars (0.88), TotRmsAbvGrd and GrLivArea (0.83), 1stFlrSF and TotalBsmtSF (0.82), ExterQual and OverallQual (0.73), and KitchenQual and ExterQual (0.72). Hence, multicollinearity exists, which may be a problem when analyzing multiple regression models because multicollinearity undermines the statistical significance of an independent variable. One solution is to try removing highly correlated independent variables from models. Another solution is to combine the highly correlated independent variables into new variables. We focused on the former for this project and decided to exclude ExterQual, GarageCars, and 1stFlrSF from



one of our models where we only use a few of the top correlated features.

### Regression Models

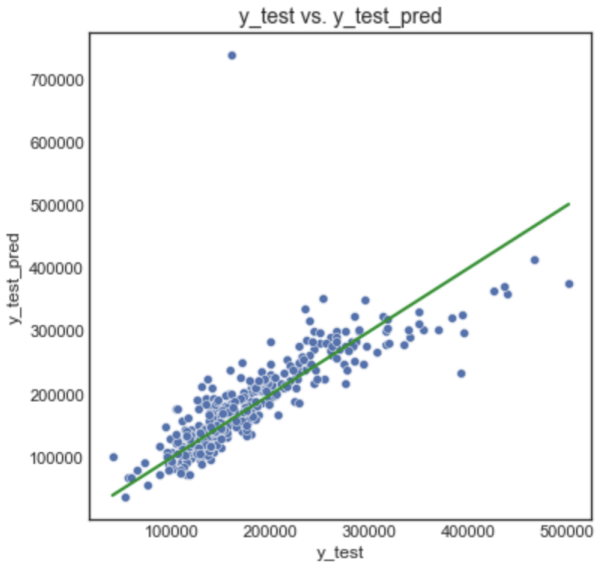
Linear regression is a technique for estimating linear relationships between various features and a continuous target variable. As part of our **Second Extension**, we will be performing a few different linear regression models in order to compare their ability to predict SalePrice.

**Linear Regression (using top correlated variables)**

***Using train/test splits:*** We performed a linear regression on the dataset, using some of the most correlated features with SalePrice for the model. We chose the following features: OverallQual, GrLivArea, KitchenQual, GarageArea, TotalBsmtSF, and BsmtQual. We decided to exclude ExterQual, GarageCars, and 1stFlrSF because of their high multicollinearity which other variables.

We split the data into training and testing sets using scikit-learn train_test_split function. We used 75% of the data for training and 25% for testing. After fitting the data to the model, we returned and printed a few testing metrics.

The testing metrics we focused on were RMSE (root mean squared error) and R-squared. The R-squared score metric describes the percentage of variance explained by the model and ranges between 0 and 1. The RMSE is the standard deviation of the residuals and tells you how concentrated the data is around the line of best fit. This model resulted in a RMSE score of 45566.93 and an R-squared of 0.703. Hence, 70.3% of the variability observed in SalePrice is explained by the Train/Test model using the previously mentioned features (OverallQual, GrLivArea, KitchenQual, GarageArea, TotalBsmtSF, and BsmtQual).




The graph to the right further provides a visualization of the goodness of fit of our model. Ideally you want the predicted values to be close to the actual values (the scatter plot should resemble the straight line). Visually, the model doesn't seem to be doing that bad of a job.

***Using Kfolds:*** Next we performed a linear regression model including the same housing features as the train/test split (above), but decided to use sklearn.mode_selectiong.Kfold in order to better evaluate the model. This approach is a great way to ensure you don't get stuck with a split that inaccurately reflects the data when only doing one train/test split. The Kfold approach involves randomly dividing the data into kfolds (or k groups/sets) of approximately equal sizes; then each fold is given the opportunity to be used as the test set and used to train the model k-1 times. The testing metrics are the mean scores of all k-folds. We used 10 kfolds (as it is one of the most commonly used) for our model which resulted in the following testing and training metrics.

```
SUMMARY OF CROSS VALIDATION: 10
test metrics:
--------------
Mean of RMSE for all folds: 38012.32809499278
Mean of R2 for all folds: 0.8064877206536138

train metric:
---------------
Mean of RMSE for all folds: 37472.8021943238
```

We can see that on average 80.65% of the variability observed in SalePrice is explained when using the Kfolds model, which is an improvement to the train/test R-squared metric (0.703). Additionally, looking at both the train and test RMSE scores, we can see that it doesn't look like the model is overfitting the data as they are relatively similar (if the test RMSE score was significantly higher than the train metric we would be concerned about overfitting).

**Linear regression (using Kfolds and all numerical variables)**

To compare, we also performed a linear regression using the Kfolds method using all the numerical variables. The train and testing metrics for this model resulted with a mean R-squared score of 0.824 and mean RMSE of 36242.652 and a training mean RMSE of 32899.913. Although this model resulted in a higher R-squared  than the previous model (0.806), something to note is the difference between the train and test RMSE metrics for this model compared to the previous model. This may indicate that including additional variables to the model may be adding noise and making the model too complex, causing the model to overfit the train data.

## Conclusion

Throughout this final project, we were able to use what we have learned throughout the course to work with a large dataset. We were able to clean and process data, explore the dataset and perform exploratory data analysis, use correlation in order to identify important variables related to the target variable, SalePrice, and then perform a few regression models. We were also able to identify, by comparing different models, that simply including all numerical variables into a model doesn't always necessarily improve the model and may actually cause a model to overfit the training data. Future work we would like to explore include working with feature transformations and feature engineering, utilizing machine learning techniques to utilize a few of the categorical variables that we excluded and didn't manually encode, and exploring other machine learning regression techniques such as decision tree, random forest, knn, and other regression models to improve the predictive metrics and to explore other models that may perform better.

## Sources


[1] https://seaborn.pydata.org/generated/seaborn.heatmap.html
[2] https://en.wikipedia.org/wiki/Coefficient_of_determination
[3] https://en.wikipedia.org/wiki/Root-mean-square_deviation
[4] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html