# Data Wrangling - Capstone Project

## Project Review

This project uses the data from a Kaggle competition. The goal is to predict the value of a house, given a set of features of that property.

## Overview

In general, this Capstone Project doesn't require too much cleaning and data wrangling, as it's part of a Kaggle competition and the data is already clean and organized.

However, some steps are still necessary. In particular, investigating missing values, looking at potential outliers, and doing a first general description of the data.

## Missing Values

Data is available in the related Notebook in this same folder.

For a first review, I created a new DF with the total number of null values per column, the total valid values, and the percentage of null values from the total observations.

See the dataframe below. 'Null Values' is a count of NaNs, 'Valid Values' is a count of not null values, and percentage is calculated comparing null values with total count.

| | Null Values | Valid Values | Percentage Null Values |
|---|---|---|---|
| **PoolQC** | 1453 | 7 | 99.52 |
| **MiscFeature** | 1406 | 54 | 96.30 |
| **Alley** | 1369 | 91 | 93.77 |
| **Fence** | 1179 | 281 | 80.75 |
| **FireplaceQu** | 690 | 770 | 47.26 |
| **LotFrontage** | 259 | 1201 | 17.74 |
| **GarageCond** | 81 | 1379 | 5.55 |
| **GarageType** | 81 | 1379 | 5.55 |
| **GarageQual** | 81 | 1379 | 5.55 |
| **GarageFinish** | 81 | 1379 | 5.55 |
| **GarageYrBlt** | 81 | 1379 | 5.55 |
| **BsmtExposure** | 38 | 1422 | 2.60 |
| **BsmtFinType2** | 38 | 1422 | 2.60 |
| **BsmtCond** | 37 | 1423 | 2.53 |
| **BsmtQual** | 37 | 1423 | 2.53 |
| **BsmtFinType1** | 37 | 1423 | 2.53 |
| **MasVnrArea** | 8 | 1452 | 0.55 |
| **MasVnrType** | 8 | 1452 | 0.55 |
| **Electrical** | 1 | 1459 | 0.07 |
| **FullBath** | 0 | 1460 | 0.00 |
| **TotRmsAbvGrd** | 0 | 1460 | 0.00 |
| **KitchenQual** | 0 | 1460 | 0.00 |

Then, I looked at each of the columns to see which type of NaN we have. In some of the columns, a NaN is actually a meaningful observation - mostly the absence of the observed feature. For example, in the ‘PoolCQ’ column (‘Pool quality’), the NaN values denote the absence of a pool whatsoever. This is also confirmed by the ‘data_description.txt’ file that comes with the dataset. For this same column, the possible values are

*Ex* *Excellent*
*Gd* *Good*
*TA* *Average/Typical*
*Fa* *Fair*
*NA* *No Pool*

Since NaN values are not computable, I transformed all null values to either a string, if they are part of an 'object' column, or to a 0 if they're numerical. In the 3 following cases, however, the NaN values were inputted with the most common value for that variable.

### 'True' Missing Values

These are missing observations in the dataset - values that are expected but they didn't appear. We can assume these are MCAR types of missing values.

They are not very frequent, here are the cases:
*Electrical* - 1 missing value - index 1379
*BsmtExposure* - 1 missing value - index 948
*BsmtFinType2* - 1 missing value - index 332

For these 3 cases, I used imputation to fill the observations with the most common value for that variable. It can be seen in the notebook, cell 211. Here are the most common values for each variable that were imputed.

*Electrical* - 'SBrkr' (Standard)
*BsmtExposure* - 'Av' (Average)
*BsmtFinType2* -'Unf' (Unfinished)

### Converting to categorical values

Convert NaN values to categorical text values 'None'. The columns affected are:
*PoolQC*
*MiscFeature*
*Alley*
*Fence*
*FireplaceQu*
*GarageCond*
*GarageType*
*GarageQual*
*GarageFinish*
*GarageYrBlt*
*MasVnrType*
*BsmtExposure*
*BsmtFinType2*
*BsmtCond*
*BsmtQual*
*BsmtFinType1*

### Converting to numerical values 0

Some other NaN values should be converted to 0, as they are part of numerical observations. For example, the NaN values in the column LotFrontage ('Linear feet of street connected to property') are equivalent to 0 feet of street connected to property.
Here are the columns with NaN values converted to 0:
*LotFrontage*
*MasVnrArea*

At the end of all these cleaning steps, there's a new dataframe ready to be consumed with the name 'filled'.

### Other notes on data

By exploring all the unique values for the object dtype columns, looks like no typos are present in the categorical values.

The Year remodeled variable is the same as the YearBuilt in 764 cases. That means in 764 houses there was no remodelation done after buying.

## Description / Exploration

An additional step was conducted, to do a first exploring pass where we can compare basic statistics like mean, median, max and min values, etc.

This was done using the pandas .describe() method, as well as the pandas profiling package.

Only the relevant columns were included in the .describe() method. For example, the ID column or the MSZoning column were excluded, because they're categorical even though they're numerical.

It all looks good using the describe() method. No apparent outliers come up from the basic statistical methods performed.

Likewise, no actionable warnings come up from the results of the profiling package.

## Notes for next steps

Further EDA will be done in the next submission of the capstone project. Some ideas to explore are the following:

- Explore the possibility of aggregating all binary features in one column (Garage/No Garage, Pool/No Pool, Alley/No Alley…) to reduce variables.