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 values per column, the total valid values, and the percentage of null values from the total observations.

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 that fall into either a string, if they are part of an 'object' column, or to a 0 if they're numerical. In a small number of cases, the entire row had to be deleted.

'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

The right way to go here is to remove those observations from the dataset (remove the whole row).

This has to be done before converting the NaN values, as in these 3 cases NaN is not equivalent to an actual observation, but simply a lack of data for that particular observation.

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’.

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 where 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 warnings come up from the results of the profiling package.

Further EDA will be done in the next submission of the capstone project.