**1. Missing values**

**1.1 There are missing values in the provided dataset for both training and test set. Will the missing values affect the model training?**

Yes, this will affect the model training. In general, machine learning models require data sets to be adequately prepared and transformed if it’s to have any hope of offering potentially meaningful predictive models. Missing values can bias the training of models or prevent them from running altogether.

Specifically, the missing values in train.csv and test.csv are strings which is a different data altogether to the rest of the column. Linear regression model training can’t handle string data types and require numerical data types only. Naively running the raw train\_data and test\_data into an ordinary least squares regression function results in a runtime error: “*ValueError: could not convert string to float”.*

# Naively calling a regression function with raw data

run\_lin\_regression(train\_data.drop(['SalePrice'], axis=1), train\_data['SalePrice'], test\_data.drop(['SalePrice'], axis=1), test\_data['SalePrice'])

**1.2 How do you solve the issue? Do research on the possible solutions, choose two solutions and justify the reason you choose them.**

To solve the issue I first manually inspected the data to determine what ‘missing’ means: NaNs, Infs, empty cells, corrupted strings? In practical machine learning problems the data set will have any combination of these.

To evaluate this data set, I imported test.csv and train.csv as a panda data frame and manually inspected to get an initial impression. I could clearly see the string ‘Missing’ in columns. I confirmed this by checking the datatype of a cell with ‘Missing’ which returned a dtype(‘O’), which is a panda string type.

I then created a function to find all the columns that contain the string ‘Missing’. The test.csv is missing values in the columns ['OverallCond' 'OverallQual' 'YearBuilt']. The train.csv is missing values in the columns ['OverallCond' 'OverallQual' 'YearBuilt']. These columns hold numerical data. I also checked for NaNs, of which there were none.

To get rid of the Missing values I used two solutions.

1. The first is to simply delete all the rows with missing data. This is a viable option because only 5.7% of the training data rows are discarded and 5.2% of the rows in test.csv. This percentage is acceptable to discard based on heuristics.
   1. In train.csv, there are 66 rows with ‘Missing’ data, which is 66/1168 (5.7%)
   2. In test.csv, there are 15 rows with ‘Missing’ data, which is 15/292 (5.2%)
2. The second was to replace the missing data with the mean value of the column. I quickly discovered this didn’t work because the columns had mismatched data types: ‘Missing’ is a string while the numeric values are floats. I then replaced all instances of ‘Missing’ with NaNs, the I cast the entire column as floats, and replaced NaNs with the mean.

**1.3 Write your own code to apply the two solutions and compare the regression model’s performances on the datasets that apply the two solutions. Section 5 gives reference about handling missing values. (6 Marks)**

The two solutions to handle the ‘Missing’ data are applied after additional cleaning operations\*. Dropping the rows with ‘Missing’ data resulted in lower coefficient of determination vs. replacing ‘Missing’ with the mean of the column. The coefficient of determination decreased approx. 0.05. Losing the number of unique data ‘points’ (rows in this case), has a noticeable impact on the model.

\* Notes on the additional cleaning operations. Please see the code.

* No features were dropped.
* One shot encoding added more features by creating columns for text categoricals.
* No scaling was applied
* No normalisation was applied
* Data types were converted to integers

|  |  |
| --- | --- |
| Replacing ‘Missing’ with the mean of the column | Dropping rows if any column contains ‘Missing |
|  |  |