Nick Stanford
Springboard
Data Science Career Track

# Capstone Project 1: Data Wrangling

The data for this project is split into two files, a training set and a test set with the target variable 'trip_type' withheld, each consisting of roughly 800,000 records. The features for each record are 'trip_type', 'visit_number', 'weekday', 'upc', 'scan_count', 'department_description', and 'fineline_number'.

The initial step was to load the data into Python as a data frame. The column names were replaced with ones that have proper formatting: all lower case letters with underscores for spaces.

Once the data was loaded, the first step in the cleaning process was to check the data type of each of the variables and convert it to a more appropriate type if necessary. Both 'weekday' and 'department_description' were initially of type object, as they are text descriptions. Instead, we convert them to categorical variables, as this reduced the amount of memory used by the file and will make the machine learning process easier later. It should be noted that in changing 'department_description' to a categorical variable, we have eliminated the possibility of performing a text analysis on it. However, naively it does not seem likely that many inter-departmental relations would have arisen between brief 1-3 word descriptions. The variables 'trip_type' and 'fineline_number' were also converted to categorical variables, as their numeric values have no ordinal meaning.

Although 'upc' and 'visit_number' are also categorical in that they have no intrinsic ordering, they were left as float64 and int64 respectively. There are so many unique product codes that converting them to categorical would actually increase memory usage. It does not make sense to treat 'visit_number' as a category because one would not expect the visit numbers in the test set to be the same as those in the training set. Indeed, 'visit_number' is not a feature with any meaning at all to directly contribute to 'trip_type'. It instead serves as a means of grouping records within the data set, and it is those groupings that jointly determine a single 'trip_type'.

There are no outliers in the dataset, but sometimes the 'scan_count' for a record is negative. This is used to indicate that an item has been returned. A return, however, is distinctly different from a purchase: returning one box of spaghetti is not “almost the same” as buying one box despite the numbers being close. In order to get the machine learning model to recognize this difference, I have split ‘scan_count’ into two new int64 variables: ‘purchase_count’ and ‘return_count’.

In a small subset (under 1%) of the data, the ‘upc’, ‘department_description’, and ‘fineline_number’ have missing values. It turns out that the records for which ‘upc’ and ‘fineline_number’ are null correspond perfectly. Further, the records for which ‘department_description’ is null is a subset of the previous set. Interestingly, the complementary subset, for which there is no ‘upc’ or ‘fineline_number’ but ‘department_description’ is not null, all have ‘department_description’ = ‘PHARMACY RX’. Because the data is categorical, we cannot fill in the missing values with a statistic such as the mean, nor can we back or forward fill since the data is not from a time series. The data could be filled randomly based on the distributions of the variables for their non-null values, but this would lead to erroneous relationships. Because so little of the data has this problem, we could simply drop it. However, the test data set also exhibits this phenomenon, so it must be accounted for. It is best to merely replace the null values with a single new unique value for each of these variables.

It seems reasonable to suspect that there might be a hierarchical relationship between the upc, fineline, and department and that this information should be encoded into the dataset. However, such a relationship does not strictly exist. While each upc comes from a single department, a few upcs come from two finelines. Furthers, roughly half of the finelines have instances of coming from different departments. There is no action to take here.

The last step I took was to split the training data set into two variables: one containing the target variable and the other containing the feature variables. At this point the data is ready for further analysis.