By Alex A. Garfias, and Brandon Leal

ETL of Credit Default data from The United States, Germany, and Taiwan

Took data for credit default information from United States of America, Germany, Taiwan. We picked these countries because this data was very hard to come by and not every country wanted to share, or document.

United State: <https://fred.stlouisfed.org/series/DRCCLACBS>

German: <https://archive.ics.uci.edu/ml/datasets/statlog+%28german+credit+data%29>

Taiwan: <https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset>

For United States data, started by downloading 4 different CSV flies, then setting the index to ‘DATE’. After the index was set, we casted the index to a datetime “<M8[ns]”; then passed each data frames into a list. We proceeded to import “functools.reduce” to pass a lambda function onto a list of data frames, to merge them all together.

­

For the German data set we began by taking the “german.data” file and reading it in as a CSV file, but passing the delimiter to a blank string such as: ‘ ‘. The data was stored as keys, that needed to be converted over to become usable. The key list was described where you downloaded the data set. We placed the keys into a dictionary that we then passed into a data frame. Using pandas “df.replace” method to iterate over the rows, casting the data to make it readable.

In the Taiwanese data set it started with an unformatted multilayered index. We cut the undescriptive column names and replace it with the lower tiered column names. Next began the task of converting the ambiguous row information. Using loc we parsed certain columns, and casted the correct value over the ambiguous data.

Due to the way the information varied from country to county we decided that a non-relational database would be the best fit for our data. After instantiating our database, we proceeded to parse the information. In order to accomplish this, we had to get creative because there is no build in method for a MongoDB import from Pandas. Our way of accomplishing this task was to take our data frames, transpose them, cast them to json, and extract the values. Into a “json.load” method. Taking those records, we used PyMongo to “import\_many” into our database.