Adrian Halarewicz: '*Wrangling U.S. Student Loan Data*'

Link to project file: github.com

The data for this project came from a variety of different sources pertinent to Student Loan data in the US. The raw CSV files include data for mean income by age, median income by age, number of loans distributed, loan balance by age, earnings by occupation, and cost of living information. The data regarding occupation income data by age was produced from US Census data, while loan specific data came from the federal student aid database and data.world. The cost of living index data was obtained from kaggle.

Many of the files were imported in an already clean format, but the census data required a significant amount of wrangling to construct a '*tidy*' (Hadley Wickham, *Tidy Data*) DataFrame.

The mean_income and median_income by age data were estimated from Census data by PK in his article on dqydj.com. The columns containing mean and median income data were converted to floating-point numbers after removing monetary formatting characters such as commas and dollar signs. The mean and median tables were merged by aligning data based on age.

Data regarding the amount of debt distributed to borrowers in 2014 was imported as a CSV file from studentaid.ed.gov. Similar steps were applied to remove formatting and convert the total number of borrowers for each partitioned loan amount from a string type to integer. Columns in the DataFrame were created for the total number of borrowers in the table, and the percentage of borrowers in each partitioned group was calculated in a new row.

Student loan balance by age data was downloaded from data.world where each row represented data for a singular year between 2014 and 2014. The column variables contained data for the Loan Balance of each ten-year age range between 30 and 60. The goal was to wrangle the DataFrame to have rows for each age between 18 and 80 with columns containing Loan Balances for the age over the years 2004 to 2014. To do this, the matrix was first **transposed** and the columns were appropriately renamed. Rows containing unnecessary information and unknown data were stripped away. The index for each row was set to the starting age of the partitioned age group, allowing a **reindex** to create empty rows for each missing age in the group. The empty rows were filled using a **forward fill** to drag down the data at the start of the age partitioned group to fill the values in the following rows in the group. A separate column age_group was created to identify the origin of the data used to fill the row. A simple **plot** of Loan Balance by age_group was constructed to inspect the DataFrame values using the **pandas** plot function built on **matplotlib**.

Adrian Halarewicz: '*Wrangling U.S. Student Loan Data*'

Link to project file: github.com

To create the final version of the loan_data DataFrame, the income_by_age was **joined** with the loan_balance_by_age table **on** the mutual 'age' column.

Building the DataFrame containing median earning by occupation was the most labor-intensive part of the Data Wrangling process. Occupational data came in nine different files from data.census.gov where each file contained median income by occupation for each year between 2010 and 2018. The files were saved in a separate sub-directory in the data folder so the **glob** function could be used to create a list of filenames to be read into DataFrames. A simple for loop iterated through the list of filenames and constructed a list of DataFrames that were **concatenated** after the loop. After inspecting the DataFrame, rows indexed with zero contained strings for each column name in a hard to read format. Rows indexed with 1 contained data for each year. Because pd.concat does not change the index of any rows, it was east to use the **.loc** method to remove all rows indexed with 0 and keep the rows of data indexed with 1, before **resetting** the index. A column in the DataFrame was created to hold the filename corresponding to each row of data because the file was the only source for obtaining the year which the data corresponded to. The year was sliced from the filename string, a new column for the year was created, and the filename column was **dropped**. Each column of median income by occupation data had a corresponding Margin of Error column that needed to be removed. To do so, a simple **Regular Expression** was used to identify row labels containing the string 'Margin of Error' and create a corresponding list of booleans. The matrix was transposed and the boolean series was used to **filter** out the rows corresponding to Margin of Error data. Additionally, rows with data not relevant to the occupational incomes needed to be removed. The column containing the label for the row of data was **split** and **sliced** to create columns for income type, gender, and occupation. Filtering was used to remove all rows that did not correspond to Median Earnings for Males or Females. The table also contained race and ethnicity-specific median income data. To remove these rows, another **RegEx** was used to identify rows containing, 'Race', 'race', or 'races' in order to filter them out of the DataFrame. Finally, rows with any missing values were dropped using the argument **how='any'**. The DataFrame was **Multi-indexed** and sorted by occupation and gender to create the final version. A basic plot of Median Income by year was built to visually inspect the results.

The cost of living data from kaggle.com was imported using a **python script** using the **kaggle API**. Using a simple **lambda function**, the city and country were split into separate columns, allowing the data to be filtered by a desired location.

The final data frames were written as CSV files to a separate file designated for the clean data. These wrangled and cleaned files are ready to be read and used for analysis.