Capstone Project Report

Author:
Sarvesh Shah, tuj80000@temple.edu

Problem Description:
SEPTA uses a legacy system to generate periodic reports related to financial data. These reports are exported in text format and often do not have a consistent format. For some files, it is possible to use simple excel tools like 'text to columns' but for most files, it requires some additional data wrangling and cleaning. Until now, due to difficulties in data extractions from old files, SEPTA has not been able to make use of these generated reports. Some of these reports included files like employee pay stubs. Once the files could be mined, they could optimize SEPTA finances and generate key insights like spending on raw materials, salary information, budget allocations, etc.

Objective:
The objective of this project is to implement a system and establish a workflow to generate reports from these raw text files. The steps are as follows:

1. Write scripts to mine files
2. Establish a data dictionary and establish a database
3. Generate dashboards for every possible aggregation using a BI tool like Tableau.
4. Generate insights from these reports.

Approach/Methodology:
We plan to start this project with developing the system to mine the data. The language we plan to mine the data is Python, with Pandas and Numpy libraries. We plan to use regex and other Python libraries and functions to develop logic for mining. The next phase of the project also involves Python to use it's dataframes and high order aggregation functions to create summarized outputs and analysis from the data. Once we have these data outputs, we plan to develop a SQL workflow to create a database and a data dictionary. Followed by automated dashboards which will process and present insights. The long term goal is to automate this entire workflow using 'Windows Task Scheduler' which will auto update these dashboards biweekly.

Data Description:
The financial data of SEPTA comprises of 35 different textual unstructured files ranging from transactions amongst companies that SEPTA deals with and employee hours and wages information. This data is pumped out on a bi-weekly basis.

Data Issues and Solution Approach:
File Structure:

1. I encountered some serious issues when I was mining certain data files. For instance, there were interesting variations (header changes, structural changes due to different parts of the report) in the files that required me to write additional logic to isolate these cases and then systematically merge these isolated columns in the main files, preserving the order was of utmost importance in order to maintain the integrity of data.
2. Another challenge that we faced was that in certain files, the output file had just one column populated for the category stored in the previous column along with numeric data and I backtracked my code to aggregate and then roll up all those columns in a single row (where the pointer resided).

File Format and Name:

1. Since we assume that a user can pass any file as an input to the system, I wrote a file-validation code that ensures that the user selects a text file and the appropriate file from the file list that can be mined by the system. The user can also have files with strange alphanumeric characters in them like 'Filename - 1' or 'Filename (1)', etc., so I first cleaned the file names and directories and showed the user the core type of file before the mining script is executed.

File Version:

1. As the system pipes out data on a biweekly basis, there was a need of a time tracker that could tell the user the period that this file belongs. In order to achieve this, I wrote a small date extractor function that uses regex to specifically extract the report end date from the entire text file and add it in the output file name as an extra column in the final excel file.

Filling missing data:

1. The reports were not designed with a CSV or a spreadsheet format in mind and thus, in a lot of instances, the column will have a single cell with some data followed by empty cells until the value of that cell changes. In cases like this, firstly I had to identify these missing cells, mark them and use forward fill, backward fill or a combination of these functions to accomplish the task.

Data Cleaning:

1. As the entire unstructured file is essentially treated as a string, the structured version of the file required a check for data type and reassigning data type. A lot of columns, especially those with currency values, stored them in "$##,###" format. In order for any numeric aggregation to be possible, I had to clean the '$' and ',' or '.' characters and convert them into numeric formats.
2. The date columns also had the same problem with most of them being a text and not following a standard format. Thus, I had to convert date strings into standard date-time objects for further slicing and dicing.

Code Workflow:

The code comprises of 2 modules - file finder, and file miner.

- File finder guides users to use a GUI of a file dialog box to open any file they chose to mine. Finder also generates a token that the miner uses to understand which script is to be executed for mining the given file.
- File miner once invoked, calls the appropriate function. Each function is conveniently named as token itself to ease the nomenclature and execution process. The function then does all the cleaning, mining parts, aggregations if required, and handles the exporting. An excel writer object is invoked and it uses the file path and the data frame to export the cleaned file as a .xlsx file with the cleaned name and date attached to it.