**An Observation of the Data Science Market Using an Extract, Transform, Load (ETL) Approach**

**By**

**Desmond Walker**

**Lito Biala**

**Chike Uduku**

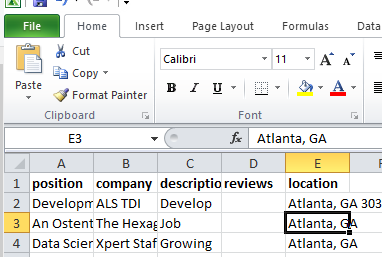
**Motivation**

Due to advances in technology, data is readily available at an amount greater than any time in modern history. The breakneck pace at which more and more volume of data becomes readily accessible does not look to be slowing down anytime soon. As a result, there has been a huge spike in demand for experts who can adequately access, process and interpret this data. Our team has decided to embark on a simple exploration of the Data Science job market to find out information such as what type of jobs are available, what languages are used to aid the processing and visualization of obtained data, in-demand methods used to model data, etc. We will employ an ETL approach. Further details about each step of our approach are given below:

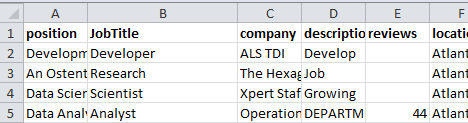
**Method**

**Extract:** For our original data, we primarily made use of two csv files sourced from <https://www.kaggle.com> and <https://data.world/> respectively. The first file contained amongst other things, job positions and descriptions for data science jobs across the United States. The second csv files contained data science job titles as well as potential employers, aggregated learning platforms and so on.

**Transform:** The first part of transformation done on our data was actually done in Microsoft Excel, as this was more convenient. Most of the transformational work involved normalizing data fields such as job titles. For example, some job titles were posted as “Analyst” in some places and “Data Analyst” in other places. Other job titles had a variation of Developer such as “Developer: App code(C#, JS) “ and “Developer: T-SQL”. To normalize this data, a separate column called “Job Categories” was created. We then went through the list of job titles to settle on the following categories: Analyst, Data Scientist, Engineer, Systems Administrator, Research, Developer, Sales, DBA (Database Administrator), Architect and other. Each job title was put in one of these category bins.



*Fig 1: Data prior to normalizing*

**

*Fig 2: Post normalized data*

It should be noted that we originally had over sixteen thousand records, but normalizing this in the time span given for this project would have been a bit tedious, so we reduced the data to six hundred records for this project.

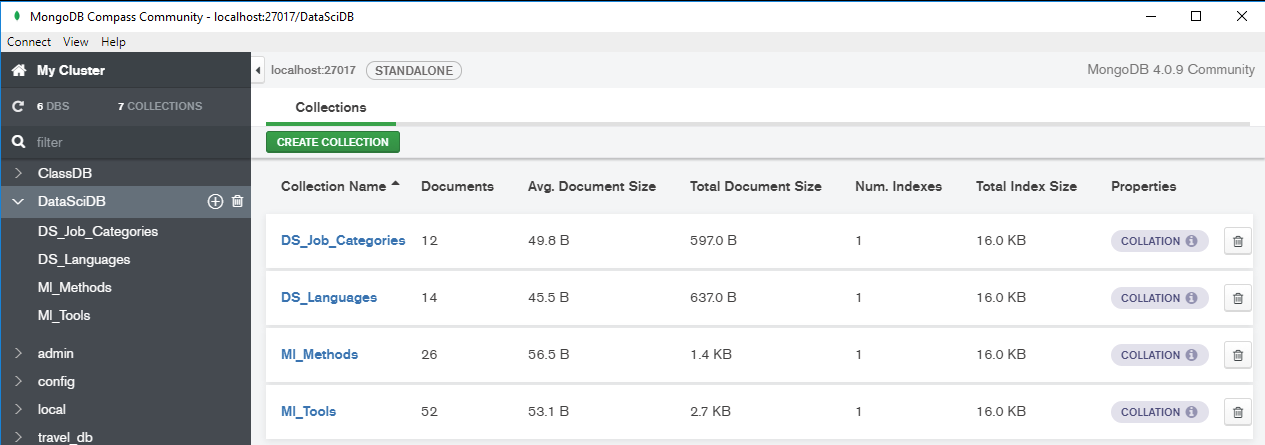
The second part of transformation done on our data was done in Jupyter Notebook. To achieve this, our partially transformed data was read into the Jupyter environment and stored in data frames. We then proceeded to use pandas methods to obtain a unique series for the data that we were interested in. More details about this part are available in the supplemental Jupyter notebook included with this report.

**Load:** For the load part of our project, we first had to determine what kind of database we were going to use. Our goal was to obtain information about the data science job market and store that information in a database. There isn’t necessarily a strong structure between each set of data. As a result, we decided that a NO-SQL database like MongoDB would be best suited to our type of data.

To achieve successful loading of our database, we had to do the following:

* Create a database called “DataSciDB in the MongoDB environment.
* Create a collection called “DS\_Job\_Categories” and insert data reflecting data science job categories into this collection.
* Create a collection called “DS\_Languages” and insert data reflecting languages used in the procuring and processing of data into this collection.
* Create a collection called “Ml\_Tools” and insert data reflecting current tools used for machine learning into this collection.
* Create a collection called “Ml\_Methods and insert data reflecting most widely used machine learning models for mapping and modelling of data into this collection.

More details of this process are provided in the supplemental Jupyter Notebook.



*Fig 3: New database with collections of loaded data*