Project Report

Title - Resume Tracking System

Class: CECS 590

Team Members:
Rajan Patel
Suraj Nair
Adheep Shetty

Abstract

Resume analyser is a dashboard consisting of visualizations which have been made to fulfil the need of recruiters or companies who are looking for right candidate which are perfectly fit in job description. It helps the recruiters to get exact count of people who are currently working for a particular company. If the user wants to hire applicants only from a certain university they can have a look at count of how many graduated from that university and can decide accordingly what percentage of students can be picked from that university.

While creating visualizations we have taken into consideration all the possible cases that could prove helpful to a recruiter. Thus this dashboard proves to be a great source of information for the recruiters and serves its purpose.

However, while scraping the data it was found to contain certain anomalies and not uniform. So this scraped data from indeed.com it would not be of any use if it would not have been manipulated or analysed by us. So we made sure that data is uniform before we start using it, to get most out of data.

The main idea of this project is to keep track large number of resumes, here “large” means not in hundreds or thousands but in millions of resumes. To handle such a large amount of data we will be using new emerging technologies like Mongo and Kafka which will provide an effective and faster way to store and stream data and at the final result of the project, we have provided best data visualization user interface to track and easy search control over information.

Index

Page No.

1. Introduction 3

2. Project system design. 4

3. Data collection (Web Scraping). 4

4. Data streaming (Kafka). 6

5. Data storage (Mongodb). 9

6. Data visualization (Tableau). 10

7. Project results. 15

8. References. 15

1. Introduction

About project.

We have developed resume tracking system for recruiters, which keep track of large amount of applicants information and provided better, faster and efficient way to store and analyze informations using advance technology.

About data set.

We are performing web scraping to collect data from “Indeed website”. The reason to do so is resume information keeps updating after certain period of time, it do not make sense to store just data and make analysis on static information. It is important that data keep on updating which can only possible by performing web scraping on platforms were updated information about job seeker regularly updated.

On Indeed website, there are over 12 million resume are present and all the candidates information being updated and maintain on regular basis. Taking analysis of such large information provide huge benefit to recruiters, can target right applications for the right job position.

Technology used.

We have research many different and most latest technologies, and choose right one which give maximum benefit for this project. Technologies we used to build this project are.

- Data collection - Web scraping using python.
  - Beautifulsoup.
  - Selenium.
  - Threading.
- Data Streaming - Apache Kafka.
- Data Storage - Mongodb.
- Data visualization - Tableau.

Benefit of project over existing system.

- Over system can store information efficiently as compared to traditional system.
- The response time of data visualization is extremely minimal.

2. Project system design.

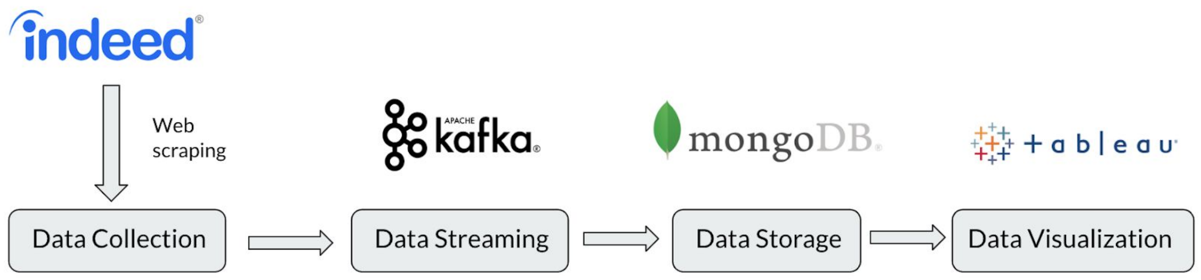
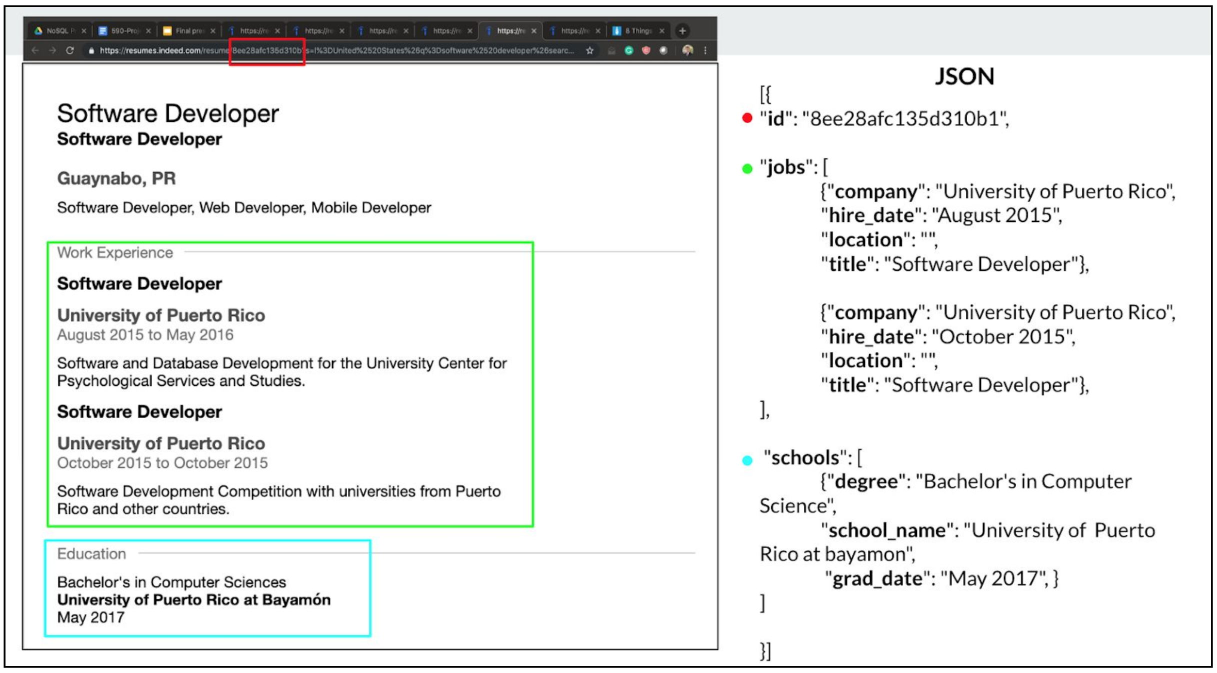


Fig. 1 System design

3. Data collection (Web Scraping).


Software Developer
Software Developer

Guaynabo, PR

Software Developer, Web Developer, Mobile Developer

Work Experience

Software Developer

University of Puerto Rico
August 2015 to May 2016

Software and Database Development for the University Center for Psychological Services and Studies.

Software Developer

University of Puerto Rico
October 2015 to October 2015

Software Development Competition with universities from Puerto Rico and other countries.

Education

Bachelor's in Computer Sciences
University of Puerto Rico at Bayamón
May 2017

JSON

```
[{
● "id": "8ee28afc135d310b1",

● "jobs": [
        {"company": "University of Puerto Rico",
        "hire_date": "August 2015",
        "location": "",
        "title": "Software Developer"},

        {"company": "University of Puerto Rico",
        "hire_date": "October 2015",
        "location": "",
        "title": "Software Developer"},
],

● "schools": [
        {"degree": "Bachelor's in Computer
Science",
        "school_name": "University of  Puerto
Rico at bayamon",
         "grad_date": "May 2017", }
]

}]
```

Fig. 2 Web scraping example.

We perform web scraping on individual resume page and collected information like resume URL to id, work experience - jobs and Education - schools. Over here “id” are unique for each resume page we have used as key. After extracting data from web it is stored in JSON format as shown in Fig. 2. This all process of extracting information and storing in JSON format is done in parallel using threads. Using multi-threads give us advantage to extract data 10 time faster compared to single page extraction. This factor of speed can increase if high processing power is used.

Data uniform issue.

After extracting data and analysing it, we found that data was not uniform. For example school degree were represent in different manner - “Master in Computer Science”, “MS in computer science”, and “MSCS”. This issue was also encountered in many different fields like college name, company name and job title.

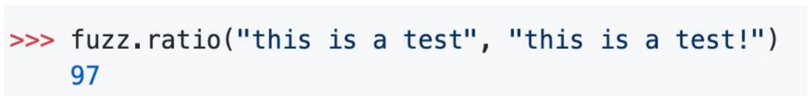
To overcome this issue we have perform pre-processing on extracted data to make data uniform and consistent. To do so we have perform following steps.

1. We replace all mnemonic with actual words (using mnemonic data set).

   MS - Master in Science.
   CS- Computer Science.
   ….

2. Used “best string matching” algorithm to map all data to its respective category.

   For example - We used Fuzzywuzzy - best string matching python library to get match score.



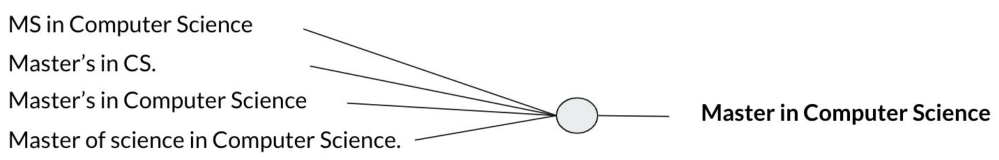
```
>>> fuzz.ratio("this is a test", "this is a test!")
    97
```

Code:

```
from fuzzywuzzy import fuzz
max=0
for i in degree_name:
     val=fuzz.ratio(degree, i)  #return best score
     if val>max:
          max=val
          str1=i

if max>50:  # 50 is hyperparameter to set data uniform level.
     #replace with best classified string.
```

Result: Data is uniform.


MS in Computer Science
Master’s in CS.
Master’s in Computer Science
Master of science in Computer Science.
Master in Computer Science

4. Data streaming.

Here are some real time data streaming tools and technologies.
There are several options for data streaming. Some of the data Streaming options are mentioned below:

1. Flink.
2. Storm.
3. Kinesis.
4. Samza.
5. Kafka.

Reason to used kafka.

Data Streaming technologies we have used Apache Kafka because of the following major advantages:

- **Acts as a Safety Buffer**:

  Now you might be thinking why we have called it particularly a “safety” buffer. We have called Apache Kafka as Safety buffer since there may be a possibility that Producer may keep producing but the consumer may be busy with other task. Thus, this might lead to an imbalance. However, Apache Kafka makes sure to store the data that is being produced into the messaging queue and the pointer of the queue does not move from the last point from where the consumer has consumed. Thus, Apache Kafka makes sure that the pipeline created for streaming is safe and does consider these anomalies.

- **Highly Scalable:**

  In out project we are currently using indeed.com as source of data and feeding the data into Apache Kafka but Apache Kafka could be potentially scaled to many more producers of data, which could be pushed into the messaging queue. Thus, Apache Kafka is highly scalable since multiple independent producer can be added in Apache Kafka.

About Kafka.

- Kafka is a distributed publish-subscribe messaging system which integrates applications/data streams. It was originally developed at Linkedin Corporation and later became a part of Apache project.

- Let’s look at some basic messaging terminology:

1. Topics: A list of data where producers add data to one end (back in this case) and consumers read from the other end.

2. Producers: Producers are processes that publish data (push messages) into Kafka topics within the broker.

3. Consumers: A consumer of topics pulls messages off a Kafka topic.

4. Broker: Kafka is run as a cluster comprised of one or more servers each of which is called a *broker*.

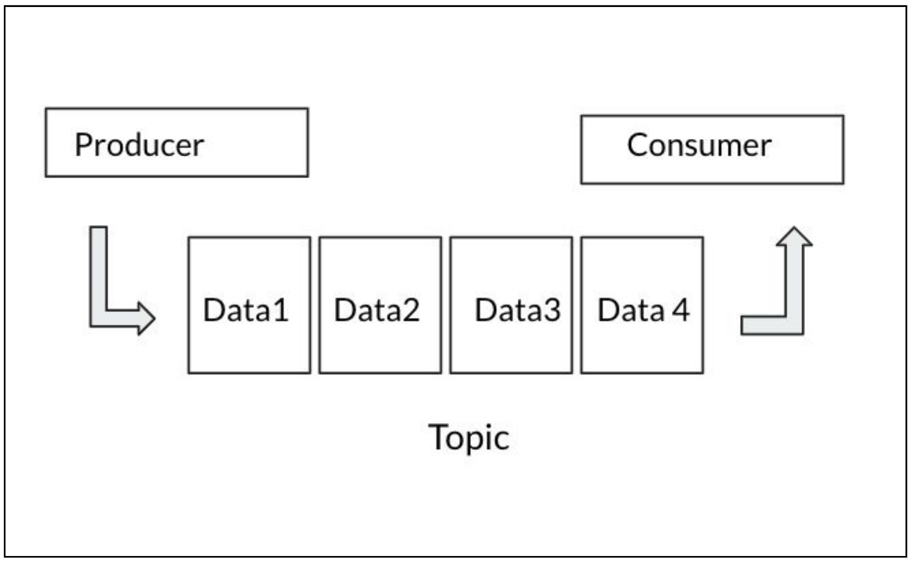


Fig. 3 Data representation in kafka.

Use of kafka in this project.

The data that is being scrapped is made uniform. This uniform data in JSON format is pushed into the messaging queue by the Producer. The messaging queue has topic name “test” in which data is stored. All the messages or the data is pulled by Consumer and sent over to a Data Storage. Thus Kafka acts as a pipeline between the Json formatted source and Data storage sink and plays a major role to regulate this continuous flow of data.

Now let see the code for Producer and Consumer that how data is streaming in json format:

Code (Producer.py):

```
from kafka import KafkaProducer
import json

#created a producer object which stores json formatted data which is being sent to brokers which has localhost address
producer = KafkaProducer(value_serializer=lambda v: json.dumps(v).encode('utf-8'),bootstrap_servers=['localhost:9092'])

# producer sending the data into the topic named ‘test’
producer.send('test', #data)
```

Code (Consumer.py):

```
from kafka import KafkaConsumer
import json

# Kafka consumer object has been created which has a topic name ‘test’ with group_id equal to ‘my-group’ which shows that this consumer belongs to my-group. The consumer is retrieving data stored in broker on a localhost server.
consumer = KafkaConsumer('test', group_id='my-group', bootstrap_servers=['localhost:9092'])

KafkaConsumer(auto_offset_reset='latest',value_deserializer=lambda m: json.loads(m.decode('ascii')))

# each message in consumer is sent to a data storage(mongo) which has certain unique id.
for message in consumer:
    try:
      #Message #(single resume data)
  except pymongo.errors.DuplicateKeyError:
      pass
    except:
      print("Error")
      Break
```

5. Data storage (MongoDB)

MongoDB is an object-oriented, simple, dynamic, and scalable NoSQL database. It is based on the NoSQL document store model. The data objects are stored as separate documents inside a collection — instead of storing the data into the columns and rows of a traditional relational database. The motivation of the MongoDB language is to implement a data store that provides high performance, high availability, and automatic scaling. MongoDB is extremely simple to install and implement. MongoDB uses JSON or BSON documents to store data. General distributions for MongoDB support Windows, Linux, Mac OS X, and Solaris.

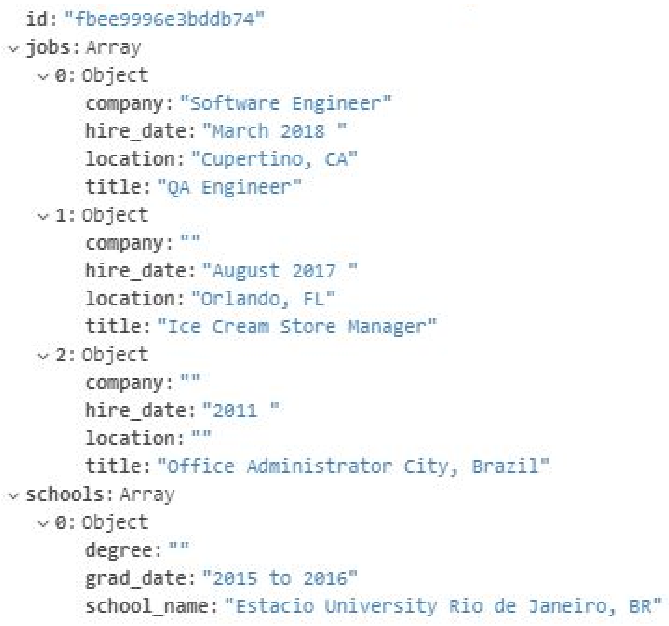
MongoDB is one of the best NoSQL database we currently have and it is the best fit for our project owning to the nature of the incoming Resume data which is semi-structured and to handle such bulk incoming semi-structured data we required a mature NoSQL database like MongoDB.

```
from kafka import KafkaConsumer
import json
import pymongo
from pymongo import MongoClient

# Below we created connectors to connect to the MongoClient, database “project” ,
collection “myCollection”
client = MongoClient()
db = client.project
collection = db.myCollection
consumer = KafkaConsumer('test', group_id='my-group',
bootstrap_servers=['localhost:9092'])
KafkaConsumer(auto_offset_reset='latest',value_deserializ
er=lambda m: json.loads(m.decode('ascii')))

for message in consumer:
 try:
  post_id =
collection.insert_one(json.loads(message.value)).inserted
_id
#Here we are importing the data extracted via KafkaConsumer into mongodb
    except pymongo.errors.DuplicateKeyError:
     pass
    except:
     print("Error")
Break
```

We have successfully imported data to the MongoDB database , following is data format:



```
  id: "fbee9996e3bddb74"
v jobs: Array
  v 0: Object
      company: "Software Engineer"
      hire_date: "March 2018 "
      location: "Cupertino, CA"
      title: "QA Engineer"
  v 1: Object
      company: ""
      hire_date: "August 2017 "
      location: "Orlando, FL"
      title: "Ice Cream Store Manager"
  v 2: Object
      company: ""
      hire_date: "2011 "
      location: ""
      title: "Office Administrator City, Brazil"
v schools: Array
  v 0: Object
      degree: ""
      grad_date: "2015 to 2016"
      school_name: "Estacio University Rio de Janeiro, BR"
```

Here we can see how in the collection all the attributes have been filtered out.

1) ObjectId : It is default created by MongoDB.
2) id : The resume id which we received from the URL earlier while web scraping.
3) jobs : Here all the job information of the applicant has been stored.
4) schools : Here all the education information of the applicant has been stored.

6. Data visualization (Tableau)

Reason to use tableau.

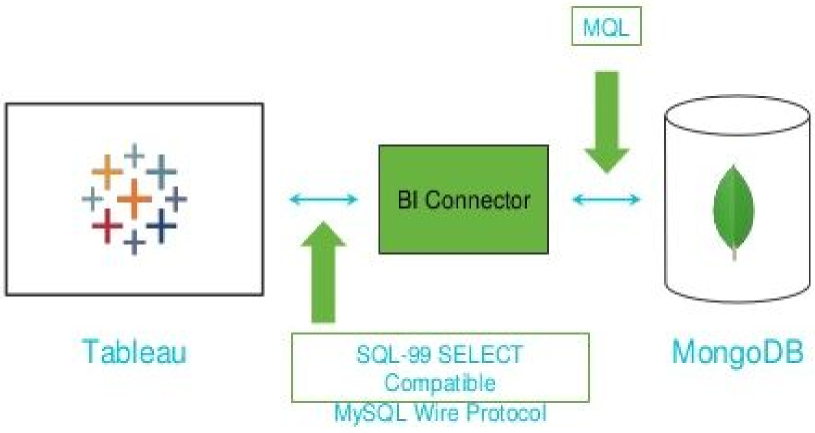
1. Tableau is a powerful and fastest growing data visualization tool used in the Business Intelligence Industry. It helps in simplifying raw data into the very easily understandable format.

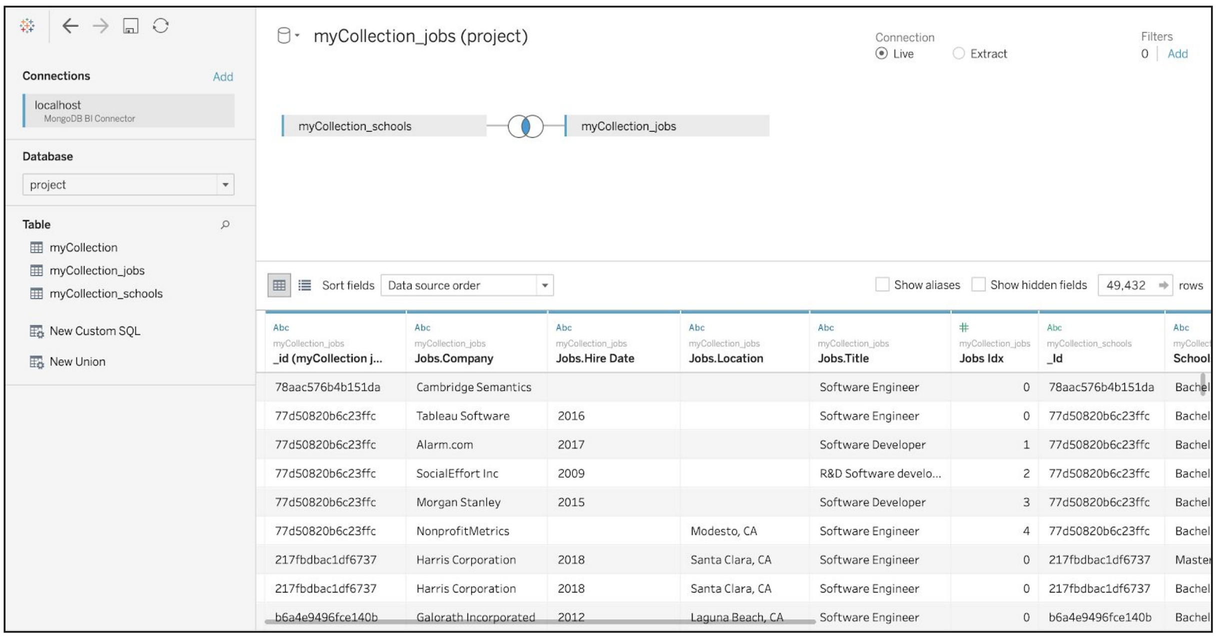
2. Data analysis is very fast with Tableau and the visualizations created are in the form of dashboards and worksheets. The data that is created using Tableau can be understood by professional at any level in an organization. It even allows a non-technical user to create a customized dashboard.

3. The best features of Tableau are
    - Data Blending
    - Real time analysis
    - Collaboration of data

Connecting tableau with data storage.

We connect MongoDB to Tableau using its BI Connector.





_id (myCollection j...	Jobs.Company	Jobs.Hire Date	Jobs.Location	Jobs.Title	Jobs Idx	_Id	School
78aac576b4b151da	Cambridge Semantics			Software Engineer	0	78aac576b4b151da	Bachel
77d50820b6c23ffc	Tableau Software	2016		Software Engineer	0	77d50820b6c23ffc	Bachel
77d50820b6c23ffc	Alarm.com	2017		Software Developer	1	77d50820b6c23ffc	Bachel
77d50820b6c23ffc	SocialEffort Inc	2009		R&D Software develo...	2	77d50820b6c23ffc	Bachel
77d50820b6c23ffc	Morgan Stanley	2015		Software Developer	3	77d50820b6c23ffc	Bachel
77d50820b6c23ffc	NonprofitMetrics		Modesto, CA	Software Engineer	4	77d50820b6c23ffc	Bachel
217fbdbac1df6737	Harris Corporation	2018	Santa Clara, CA	Software Engineer	0	217fbdbac1df6737	Maste
217fbdbac1df6737	Harris Corporation	2018	Santa Clara, CA	Software Engineer	0	217fbdbac1df6737	Bachel
b6a4e9496fce140b	Galorath Incorporated	2012	Laguna Beach, CA	Software Engineer	0	b6a4e9496fce140b	Bachel

The above image confirms the Tableau connection to MongoDB and it represents how we can extract required data by connecting the tables.

- Select the file, database, or schema, and then double-click or drag a table to the canvas.
- Double-click or drag another table to the canvas, and then click the join relationship to add join clauses and select your join type.
- Add one or more join clauses by selecting a field from one of the available tables used in the data source, a join operator, and a field from the added

table. Inspect the join clause to make sure it reflects how you want to connect the tables.

Tableau functional

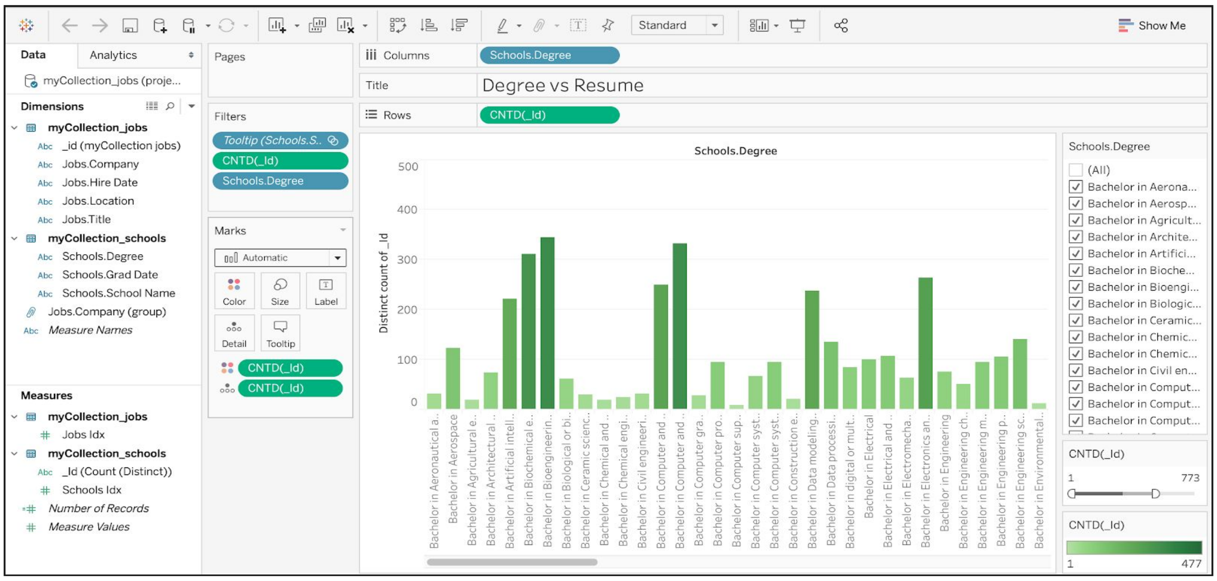
In Tableau, a user can use different types of built-in functions which can be applied to numbers,strings,data and aggregation functions. Here we limit ourselves to the functions we implied in our Tableau Sheets.

1) **MIN:** Returns the minimum value of an expression across all records
2) **MAX:** Returns the maximum value of an expression across all records
3) **DATE**: Returns a date given a number, string or date expression
4) **FLOAT**: It returns the floating number from the given expression of any type
5) **INT**: Returns an integer given an expression
6) **STR**: Returns a string given an expression
7) **AVG:** It returns the average value of the given expression or array of values
8) **COUNT**: It returns the count of items in the group. NULL values are not counted

Sample data visualization illustration

We can view in the top the current total count of resume that has been extracted.

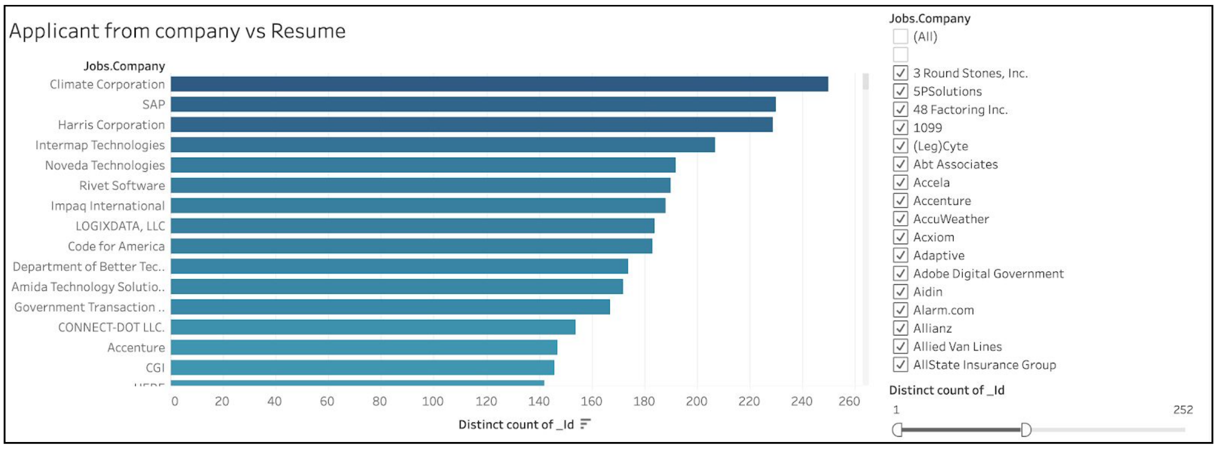
1) The first sheet showcases a barchart containing information of distinct count of the resume which has been differentiated on the basis of their degree earned by them.



Here we used the Tableau 'COUNT' function to get the resume count.

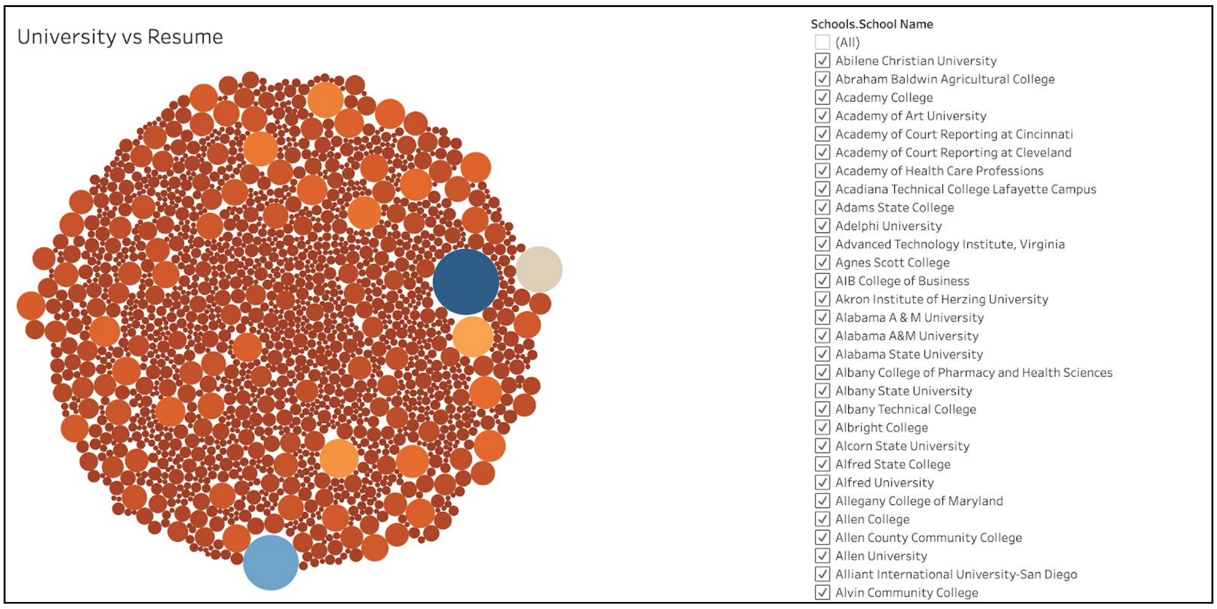
We can use the filter to navigate and search in for any specific degree information which we might require.

2) The second sheet showcases a barchart containing information of distinct count of the resume which has been differentiated on the company work history of all the applicants.



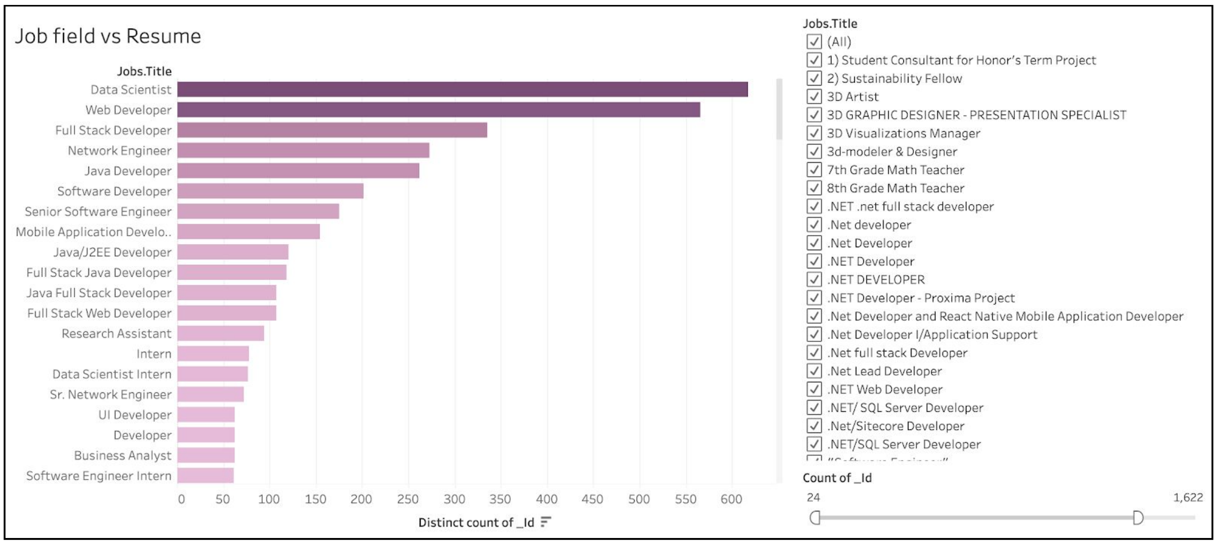
Here we used the Tableau ‘COUNT’ function to get the resume count.
We can use the filter to navigate and search in for any specific company information which we might require and also filter out on the basis of the distinct count.

3) The third sheet shows a bubble graph containing information of distinct count of the resume which has been differentiated on the basis of their university.



We can also use the filter to navigate and search in for any specific university information which we might require.

4) The fourth sheet showcases a barchart containing information of distinct count of the resume which has been differentiated on the basis of the job title of the applicants.



Jobs.Title

- ☑ (All)
- ☑ 1) Student Consultant for Honor's Term Project
- ☑ 2) Sustainability Fellow
- ☑ 3D Artist
- ☑ 3D GRAPHIC DESIGNER - PRESENTATION SPECIALIST
- ☑ 3D Visualizations Manager
- ☑ 3d-modeler & Designer
- ☑ 7th Grade Math Teacher
- ☑ 8th Grade Math Teacher
- ☑ .NET .net full stack developer
- ☑ .Net developer
- ☑ .Net Developer
- ☑ .NET Developer
- ☑ .NET DEVELOPER
- ☑ .NET Developer - Proxima Project
- ☑ .Net Developer and React Native Mobile Application Developer
- ☑ .Net Developer I/Application Support
- ☑ .Net full stack Developer
- ☑ .Net Lead Developer
- ☑ .NET Web Developer
- ☑ .NET/ SQL Server Developer
- ☑ .Net/Sitecore Developer
- ☑ .NET/SQL Server Developer

Here we used the Tableau 'COUNT' function to get the resume count.
We can use the filter to navigate and search in for any specific job title information which we might require.

Note : We can also extract the data we require after we have chosen a specific information to retrieve from the mentioned sheets.

This is interactive graph which will be automatically updated as when when any data is added or deleted at data storage. Also we can choose to navigate onto the sheets in the dashboard and pick and choose to explicitly keep if we need something in specific from the available sheets.

7. Project results.

A dashboard is a culmination of these sheets and we have created a dashboard of our sheets and published them on the Tableau Public Server wherein anyone can view and check out the results .

Dashboard view:
https://public.tableau.com/profile/suraj.nair1535#!/vizhome/Resumeactivitytracker/Dashboard1?publish=yes

8. References

- FuzzyWuzzy (best string match algorithm)
  https://github.com/seatgeek/fuzzywuzzy
- Technologies used for data streaming
  https://www.algoworks.com/blog/real-time-data-streaming-tools-and-technologies/
- Explaining why we used MongoDB in our Project
  https://www.sqlservercentral.com/blogs/why-mongodb
- Explaining why we used Tableau in our Project
  https://intellipaat.com/blog/what-is-tableau/