Unemployment and Financial Crisis In Review

Data Engineering Platforms for Analytics

Aakash Pahuja, Adrienne Wang, Brandon Allen, Prasanth Chinta

June 12, 2020

Our Team

Aakash Pahuja

MScA Candidate at The University of Chicago

RFID Solution Architect

Adrienne Wang

MScA Candidate at The University of Chicago

Brandon Allen

MBA Candidate at The University of Chicago

Senior Associate Director of Analytics at The University of Chicago

Prasanth Chinta

MScA Candidate at The University of Chicago

Senior Product Manager at a Financial Firm

Table of Contents

Does history repeat itself?

Extrapolating implications from the 2008 Financial Crisis for a COVID-19 world.

The 2008 Financial Crisis was one of the most severe global economic crises in history. Its impact was felt across all industries, and it continues to have a material impact on countless lives. Learning from the past could allow policy-makers, civic leaders, and industry entities to more adequately prepare for future economic recessions.

By understanding how the 2008 crisis impacted employment and stock performance, we seek to identify individuals and industries that are most vulnerable to economic downturns. In doing so, we hope to highlight opportunities for support during the next economic crisis.

Business Case

Financial Crisis Analysis

- Demographic Analysis
- Stock Market Analysis
- Employment Analysis

Data Structure Design

- Underlying Data Structure Design
- Data Source Clarification

Verification

- Result Verification

Objective

- Develop analytics framework to carry out the analysis of the Financial Crisis on unemployment rate
- Detail impact on stock market and workforce deployment
- Identify individuals' characteristics and industries that are most at-risk
- Design One-Page Dashboard to interactively visualize current and historical KPI, filtered by selected dimensions

Data Engineering Tools

Data Collection → Data Processing → Data Warehouse → Data Visualization

Data Collection

Web-scraping

Public APIs

.csv Downloads

Our Datasets

Data Profile	Description	Total Size	Sources #	# of Rows	Columns #	Structure	Source
Stock Market Indexes	**S&P 500, Dow Jones IA, and NASDAQ Indexes**	**16KB**	**1**	**149**	**4**	**Structured**	Yahoo Finance
Employment and Labor Force Participation Data for Education Attainment	**Labor force, employment, and unemployment statistics by educational attainment for persons age 25 and older**	**22KB**	**1**	**360**	**2**	**Structured**	Bls.gov Website and Public API FRED
Currency Exchange Rate	**US to EURO Currency Exchange Rate**	**10KB**	**1**	**149**	**5**	**Structured**	FRED
Unemployment Duration	**Unemployment Duration**	**64KB**	**1**	**588**	**5**	**Structured**	Bls.gov Website and Public API
Unemployment Rate and Volume by Demographics	**Unemployed jobseekers by sex, reason for unemployment, and active job search methods used**	**193KB**	**1**	**5,000**	**6**	**Structured (JSON)**	Bls.gov Website and Public API
Stocks Return by Industries	**Annual returns for the ten stock market sectors against the S&P 500**	**112KB**	**1**	**1872**	**5**	**Structured**	Novel Investor

Data **Quality**

Completeness	There are no missing values in our datasets as we ensured that ETL process has necessary checkups
Validity	Our data passes format check, length check, lookup table, presence check, range check and spell check
Uniqueness	Our datasets have duplicated monthly data to keep all datasets at the same granularity level (No interference with analysis)
Consistency	Data format is consistent across all tables
Timeliness	Our data except industry returns is up to date (May 2020)
Accuracy	Accuracy is verified by our data providers

Data Profile **After**

- **dim_master_codes**: “Data about races, education, marital status etc.”

    57 rows and 9 columns

Columns:

stg_id smallint(5) UN AI PK

code varchar(45)

sexes varchar(4)

races varchar(10)

age varchar(10)

education varchar(255)

marital_status varchar(45)

bls_type varchar(45)

ts timestamp

Data Profile After

- dim_period: “data about month, year”
  156 rows and 4 columns

Columns:
period_id smallint(6) PK
period_month varchar(45)
period_month_number varchar(40)
period_year year(4)

- fact_econ_data: “Data type about Dow_Jones_Average_Close, Nasdaq_Average_Close, SP500_Average_Close, employment ratio and rate, unemployment ratio and rate”
  6435 rows and 6 columns

Columns:
bls_id smallint(5) UN AI PK
period_id smallint(6)
bls_type_id smallint(6) UN
bls_value decimal(10,2)
last_update timestamp

Data Profile After

- **industry_returnRate_bridge:** “Industry name being affected in the financial crisis”

  12 rows and 2 columns

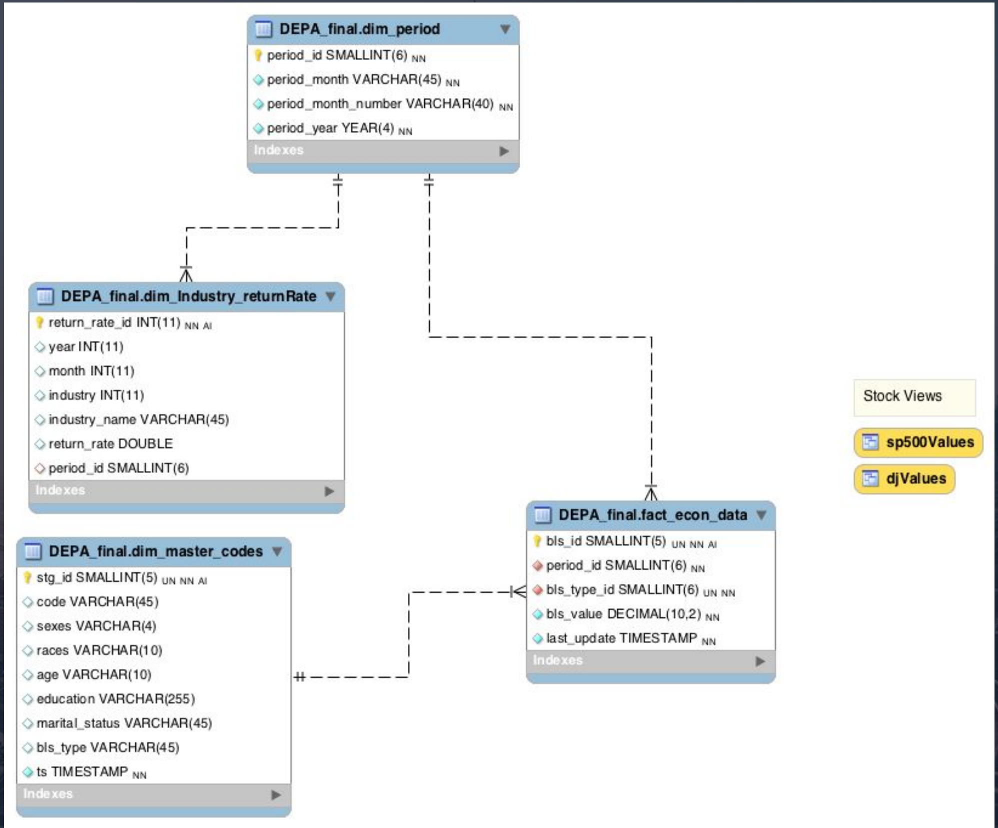
Columns:
-industry_id int(11) AI PK
-industry_name varchar(45)

- **t_backup_bls_Unemployed_data:**

  5297 rows and 7 columns

Columns:
-stg_id smallint(5) UN AI PK
-year varchar(45)
-period varchar(45)
-periodName varchar(45)
-latest varchar(10)
-value double
-footnotes varchar(45)
-seriesID varchar(45)
-ts timestamp

Our Snowflake Dimensional Table



Data Ingestion

```python
import pandas as pd
import json
import requests

def get_bls_data(series, start, end):
    headers = {'Content-Type': 'application/json'}
    data = json.dumps({"seriesid": series,"startyear":"%d" % (start), "endyear":"%d" % (end)})
    p = requests.post('https://api.bls.gov/publicAPI/v2/timeseries/data/?registrationkey=60256a471f4a427c813300a92445943c&catalog=false&startyear=2010&endyear=2021', data=data, headers=headers)
    json_data = json.loads(p.text)
    try:
        df = pd.DataFrame()
        for series in json_data['Results']['series']:
            df_initial = pd.DataFrame(series)
            series_col = df_initial['seriesID'][0]
            for i in range(0, len(df_initial) - 1):
                df_row = pd.DataFrame(df_initial['data'][i])
                df_row['seriesID'] = series_col
                if 'code' not in str(df_row['footnotes']):
                    df_row['footnotes'] = ''
                else:
                    df_row['footnotes'] = str(df_row['footnotes']).split("'code': '",1)[1][:1]
                df = df.append(df_row, ignore_index=True)
        return df
    except:
        json_data['status'] == 'REQUEST_NOT_PROCESSED'
        print('BLS API has given the following Response:', json_data['status'])
        print('Reason:', json_data['message'])
```

We are connected to the bls api to gather information for:

- unemployment_duration
- unemployment_rate and employment_rate
- consumer price index

Data Ingestion Example

Tool Utilized

Jupyter Notebook (package csv, and BeautifulSoup to open and scrape web data and write to CSV)

Web Scraping Code

```
import requests
from bs4 import BeautifulSoup
import json
import csv
from IPython.display import HTML

URL = 'https://novelinvestor.com/sector-performance/'
page = requests.get(URL)
soup = BeautifulSoup(page.content, 'html.parser')
```

```
table = soup.find('tbody')
tmpRow = (table.findAll('tr')[1:])
```

```
list_of_rows = []

try:
    outfile = open("./SP.csv", "w")
    writer = csv.writer(outfile)
    writer.writerow(["2007", "2008", "2009", "2010", "2011", "2012", "2013", "2014", "2015","
    for row in table.findAll('tr'):
        list_of_cells = []
        for cell in row.findAll("td"):
            text = cell.text.replace(' ', '')
            list_of_cells.append(text)
        arrLength = len(list_of_cells)
        writer.writerow(list_of_cells)

finally:outfile.close()
```

Output

	2007	2008	2009	2010	2011	2012	2013	2014	
0	ENRS\n34.4%	CONS\n-15.4%	INFT\n61.7%	REAL\n32.3%	UTIL\n19.9%	FINL\n28.8%	COND\n43.1%	REAL\n30.2%	COND\n
1	MATR\n22.5%	HLTH\n-22.8%	MATR\n48.6%	COND\n27.7%	CONS\n14.0%	COND\n23.9%	HLTH\n41.5%	UTIL\n29.0%	HLTH\
2	UTIL\n19.4%	UTIL\n-29.0%	COND\n41.3%	INDU\n26.7%	HLTH\n12.7%	REAL\n19.7%	INDU\n40.7%	HLTH\n25.3%	CONS\
3	INFT\n16.3%	TELS\n-30.5%	REAL\n27.1%	MATR\n22.2%	REAL\n11.4%	TELS\n18.3%	FINL\n35.6%	INFT\n20.1%	INFT\
4	CONS\n14.2%	COND\n-33.5%	S&P\n26.5%	ENRS\n20.5%	TELS\n6.3%	HLTH\n17.9%	S&P\n32.4%	CONS\n16.0%	REAL\
5	INDU\n12.0%	ENRS\n-34.9%	INDU\n20.9%	TELS\n19.0%	COND\n6.1%	S&P\n16.0%	INFT\n28.4%	FINL\n15.2%	TELS\
6	TELS\n11.9%	S&P\n-37.0%	HLTH\n19.7%	S&P\n15.1%	ENRS\n4.7%	INDU\n15.4%	CONS\n26.1%	S&P\n13.7%	S&P\
7	HLTH\n7.2%	INDU\n-39.9%	FINL\n17.2%	CONS\n14.1%	INFT\n2.4%	MATR\n15.0%	MATR\n25.6%	INDU\n9.8%	FINL\n
8	S&P\n5.5%	REAL\n-42.3%	CONS\n14.9%	FINL\n12.1%	S&P\n2.1%	INFT\n14.8%	ENRS\n25.1%	COND\n9.7%	INDU\n
9	COND\n-13.2%	INFT\n-43.1%	ENRS\n13.8%	INFT\n10.2%	INDU\n-0.6%	CONS\n10.8%	UTIL\n13.2%	MATR\n6.9%	UTIL\n

Data Cleaning

```
[ ] #li[26:41]
    #df_final = df_final.append(df)
    #df_final = df
    df_final.count()
```

```
year          4929
period        4929
periodName    4929
latest          39
value         4929
footnotes     4929
seriesID      4929
dtype: int64
```

```
[ ] start = 2008
    end = 2013
    series = li[0:25]
    df3 = get_bls_data(series=series, start=start, en
    series1 = li[26:41]
    df4 = get_bls_data(series=series1, start=start, e
    #df3.count()
    #df4.count()
```

```
BLS API has given the following Response: REQUEST
Reason: ['No Data Available for Series LNU0309222
BLS API has given the following Response: REQUEST
Reason: ['No Data Available for Series LNU0309111
```

```
[ ] # Prepare SQL query.
    sql = "CREATE TABLE t_unemployment ( \
      unemployment_id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT, \
      year VARCHAR(45), \
      period VARCHAR(45),\
      periodName VARCHAR(45), \
      value DOUBLE, \
      ts TIMESTAMP DEFAULT CURRENT_TIMESTAMP, \
      PRIMARY KEY  (unemployment_id) \
    ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;"


    try:
       # Execute the SQL command
        a = cursor.execute(sql)

       # Fetch all the rows in a list of lists.
       #rows = cursor.fetchall()
    except:
        print ("Error: unable to fetch data");

    db.commit()
```

Api Data cleaned using python packages like prettytable in Jupyter notebook

Using pymysql to do DDL and DML in MySQL.

Data Cleaning Example

Output

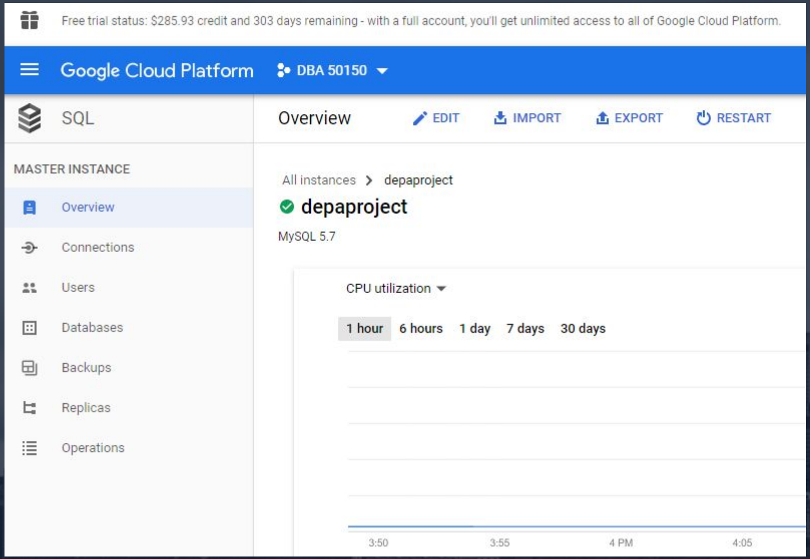
return_rate_id	year	month	industry	return_rate
5	2007	1	5	0.072
6	2007	1	6	0.12
7	2007	1	7	0.163
8	2007	1	8	0.225
9	2007	1	9	-0.179
10	2007	1	10	0.055
11	2007	1	11	0.119
12	2007	1	12	0.194
13	2008	1	1	-0.335
14	2008	1	2	-0.154
15	2008	1	3	-0.349
16	2008	1	4	-0.553
17	2008	1	5	-0.228
18	2008	1	6	-0.399
19	2008	1	7	-0.431
20	2008	1	8	-0.457
21	2008	1	9	-0.423
22	2008	1	10	-0.37
23	2008	1	11	-0.305
24	2008	1	12	-0.29
25	2009	1	1	0.413
26	2009	1	2	0.149
27	2009	1	3	0.138
28	2009	1	4	0.172
29	2009	1	5	0.197
30	2009	1	6	0.209

R Studio was used to replicate yearly data as monthly data

R Code

```
library(readxl)
SP <- read_excel("~/Desktop/MScA-Chicago/Spring2020/Data Engineering Platforms for Analytics/Final_Project_Team3/SP2.0.xlsx")
SP1<-cbind(SP, 'month' = 1)
SP2<-cbind(SP, 'month' = 2)
SP3<-cbind(SP, 'month' = 3)
SP4<-cbind(SP, 'month' = 4)
SP5<-cbind(SP, 'month' = 5)
SP6<-cbind(SP, 'month' = 6)
SP7<-cbind(SP, 'month' = 7)
SP8<-cbind(SP, 'month' = 8)
SP9<-cbind(SP, 'month' = 9)
SP10<-cbind(SP, 'month' = 10)
SP11<-cbind(SP, 'month' = 11)
SP12<-cbind(SP, 'month' = 12)
SP_3<-rbind.data.frame(SP1, SP2, SP3, SP4, SP5, SP6, SP7, SP8, SP9, SP10, SP11, SP12)
```

Data Warehouse through GCP

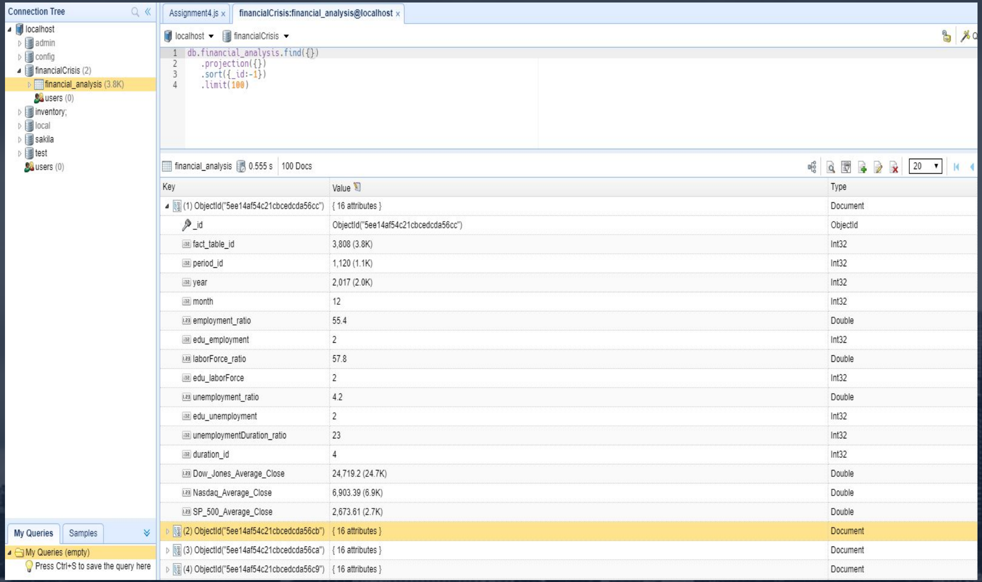


Storage is fast with uncapped bandwidth with strongly consistent listings.

With numerous data breaches and security issues reported in the news almost daily, security is on top of our mind. Hence, GCP offers robust data privacy and security features.

Fully compatible with Jupyter notebook, MySQL, and Tableau with convenient data transfer mechanism.

NoSQL DATABASE



MongoDB is document database, which fits well with our data, as each document contains several attributes (Dow_Jones_Average_Close, Nasdaq_Average_Close, SP_500_Average_Close, employment_ratio, unemployment ratio etc.) for each specific month and year.

Why MongoDB over Neo4j for OLTP?

- We used MongoDB over Neo4js because the analysis we needed was been done using MongoDB. If government officials wanted to add new data for a particular month and year, they could do that easily by just creating a new document in Mongo
- Different categorical variables are only linked through a rate number. Graphical database can only provide minimal insights to future use cases
- Our business case focuses on trend analysis mainly and relational graph provides minimal insight to our use case

Design Considerations

Outliers, Anomalies, and Aggregation

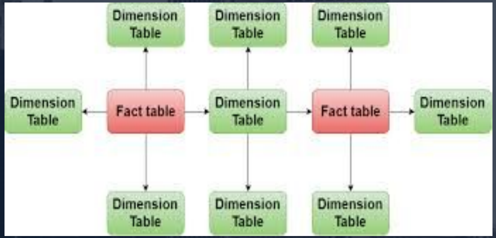
- We didn't encounter outliers or anomalous data points, but we were prepared to treat them if necessary
- Aggregated by time period depending on the native time scale to ensure consistent granularity of time

Data Transformations

- Melted and reformed dim tables to better integrate with fact table
- Reshaped industry data to successfully visualize in Tableau
- Aligned education codes across BLS and Fred datasets
- Standardized period naming conventions

Data Mapping

- We considered a galaxy schema, but decided for the purposes of this project to go with snowflake

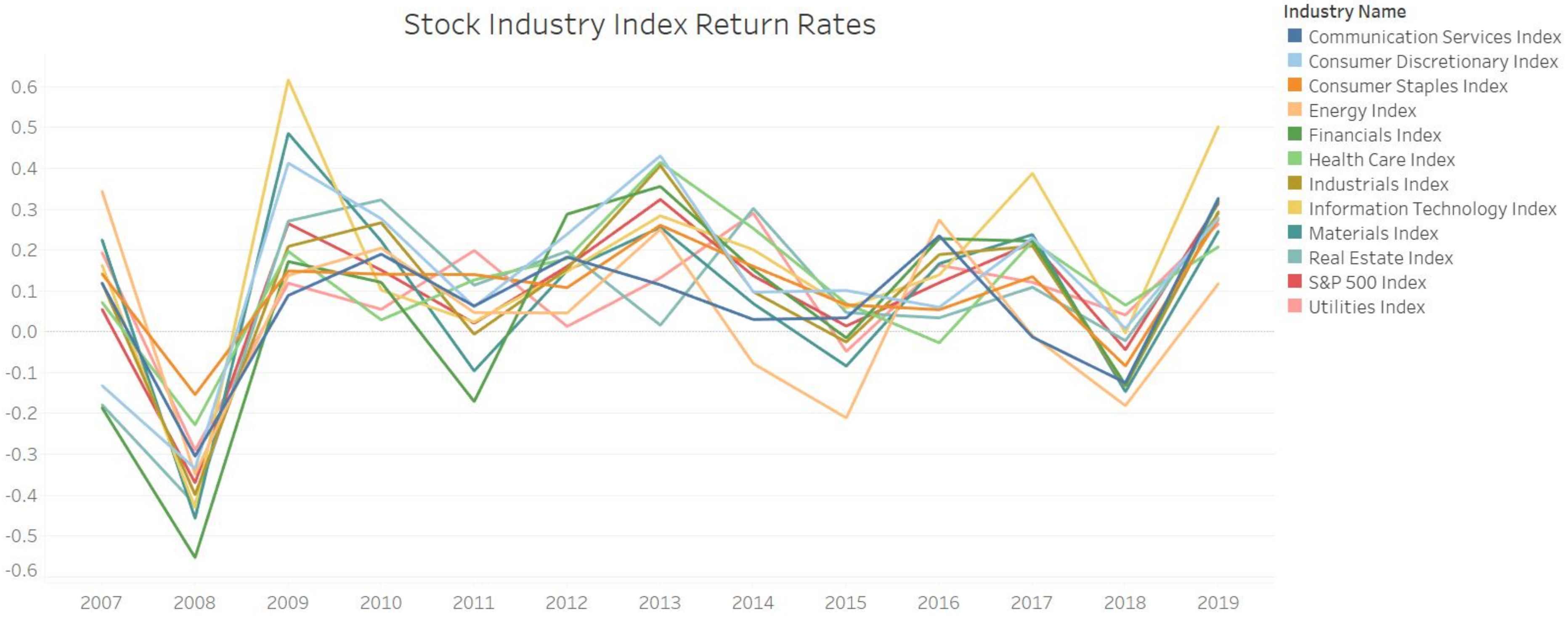


> “After creating reports and building dashboards in Tableau, we noticed that …

The decline in stock index performance during the 2008 Financial Crisis is sufficiently similar to the 2020 COVID decline

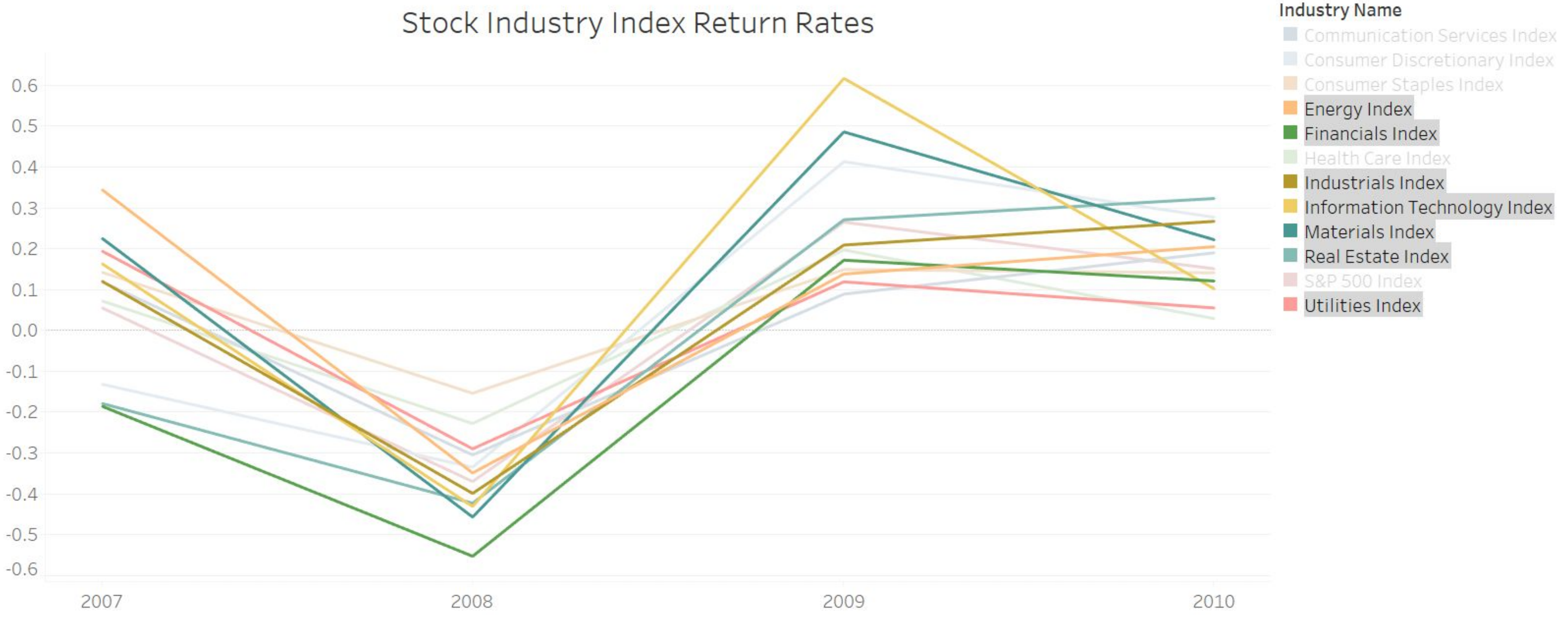
	2008 Financial Crisis Decline	2020 COVID Decline
DOW JONES INDUSTRIAL AVERAGE	Linear Model: -19.06*Month + 766,755 R-Squared: 0.95 p-value: < 0.0001 **Total Decline: 5,575**	Linear Model: -74.38*Month + 3.29e+06 R-Squared: 0.90 p-value: 0.054 **Total Decline: 6,621**
NASDAQ	Linear Model: -4.48*Month + 179,736 R-Squared: 0.917 p-value: < 0.0001 **Total Decline: 1,145**	Linear Model: -24.12*Month + 1.07e+06 R-Squared: 0.98 p-value: 0.083 **Total Decline: 1,451**
S&P 500	Linear Model: -2.44*Month + 98,074 R-Squared: 0.95 p-value: < 0.0001 **Total Decline: 665**	Linear Model: -10.66*Month + 470,484 R-Squared: 0.99 p-value: 0.068 **Total Decline: 641**

Stock industry indexes behaved similarly with a steep decline in 2008



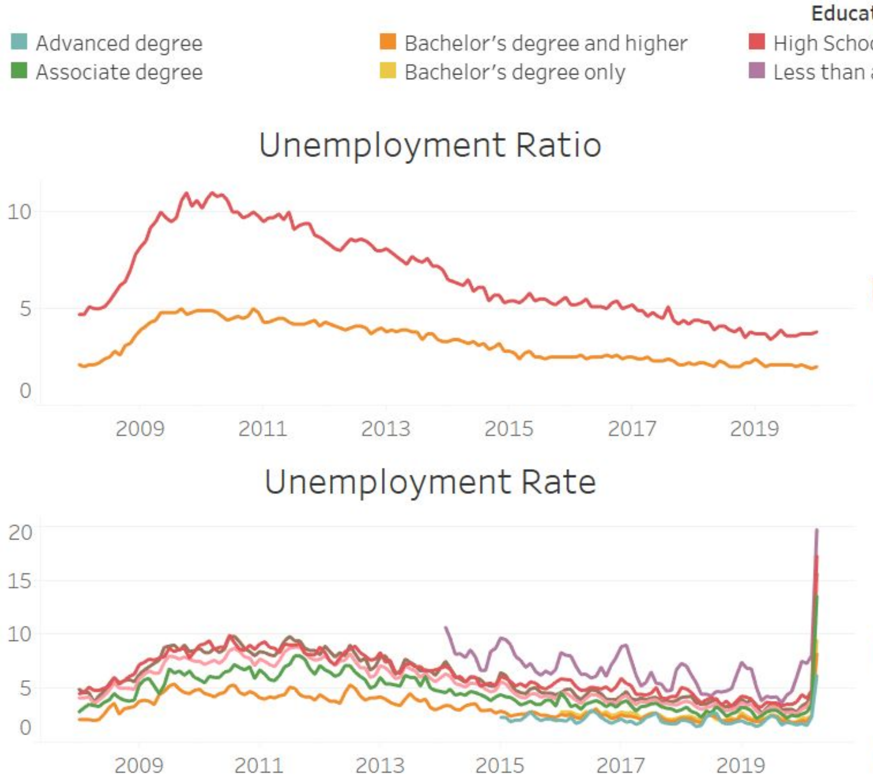
Note: this data captures 2007 to 2019

The Energy, Financials, Industrials, Information Technology, Materials, and Real Estate industries had the lowest average return rates during 2008



Energy and Utilities didn't recover until 2010

Lower education attainment suggests greater risk for unemployment, especially during financial crises



High School Graduates: **2x higher unemployment ratio** than Bachelor’s Degree Holders after 2008 Crisis

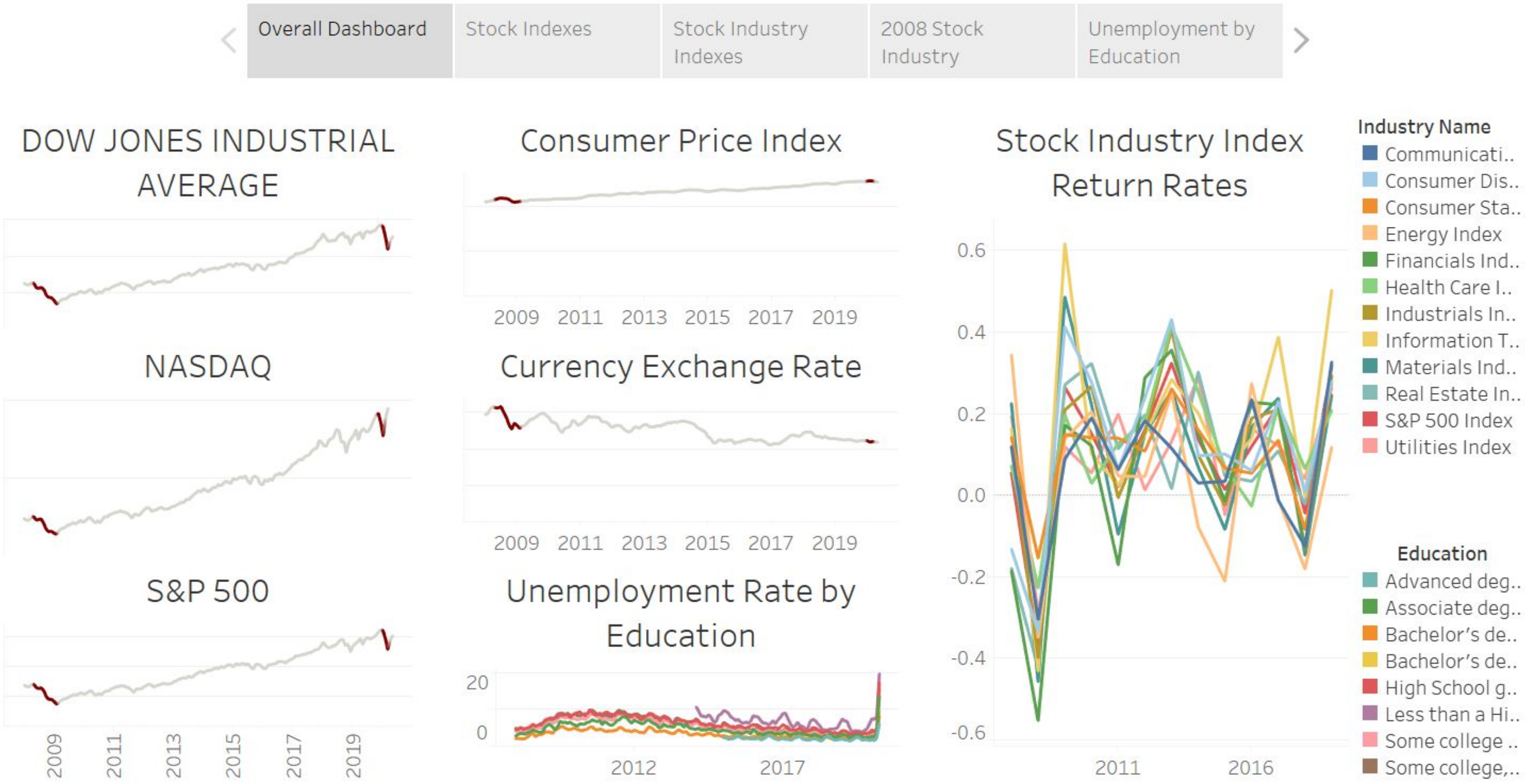
The unemployment ratio dichotomously divided into ”Bachelor’s degree” and ”High School Diploma” suggests that, while high school graduates are typically more likely to be unemployed, the magnitude of increase in the aftermath of the 2008 financial crisis is **over 2 times greater**

Less than a High School Diploma: **3x higher unemployment rate** than Advanced Degree Holders during COVID-19

This distinction by education level is also borne out when a more granular perspective is taken. While data is not available every year for each education level, the unemployment rates corroborate that increased education level leads to a lower unemployment rate. This is especially true when comparing the most educated against the least educated.

Comparisons across other metrics (Labor Force Participation, Consumer Price Index, etc.) and demographics (gender, etc.) did not yield meaningful differences

Our Tableau dashboard consolidates sheets for future research



FUTURE DIRECTIONS

The Energy, Financials, Industrials, Information Technology, Materials, Real Estate, and Utilities industries

dropped to the lowest absolute return rate during the 2008 Financial Crisis

The least educated members of society (less than a bachelor’s degree)

are most at-risk of becoming unemployed during a financial crisis

Industry and civic leaders must prepare

to implement programs that quickly and efficiently support these most vulnerable individuals and industries to mitigate long-term fiscal damage

Lessons Learned

Data quality

It was extremely helpful to have identified and leveraged high-quality and well-documented data sources

Structural decisions

Developing the schema requires tremendous vetting and iterative review to deliver best results

Schema Selection

Most datasets have only rate and date columns (Snowflake over Galaxy)

Ideal for our business case: compare trends for multiple indexes together

Time dimension is extremely important

Data availability

Finding the right data can be challenging, especially when attempting to avoid paywalls

Adding more data could yield additional insights

Thanks!