Final Project Report: Employment Checker

Chaoyu Li, chaoyuli@usc.edu, 6641732094

Wenjie Xiao, wenjiex@usc.edu, 7691362676

Yuxi Fan, yuxifan@usc.edu, 9691041731

DSCI-551: Foundations of Data Management

Instructor: Wensheng Wu

Apr. 24, 2022

Table of Contents

Website: *https://share.streamlit.io/yuxi-fan/employment_checker/main*

1. Topic

Our project mainly focuses on helping job seekers track the current employment situation in the United State. We are now gradually entering a post-pandemic age with the economy and job market recovering. Our website can help users directly see the change in the employment situation from the three different scopes: National, state-wide, and city-wide. Our website also utilizes machine learning techniques, which provide users with a prediction model of future employment level change. For users with the need of tracking employment situations in both China and the United State, we also add a feature that can display the change in the unemployment rate in these two countries.

2. Motivation

As graduate students in the first year of master's programs, most of us have already begun looking for job opportunities and many of us are suffering from worries and anxiety about the current job market. It will be great to have a platform that can help us track the employment situation in the United States. We would like to develop a web application that can aggregate data from different sources and directly showcase the changing employment situation in the United States. Our website can predict future employment trends based on past employment changes. When talking about the current employment situation in the US, COVID-19 is an inescapable topic as it has dramatically impacted the global economy over the past two years. Therefore, we also introduced the data of COVID-19 as a reference for analysis and prediction.

3. Web architecture

Our website is deployed using *Streamlit* platform. *Streamlit* is an open-source app framework that can create a web app with python. With the help of *streamlit*, we were able to save the trouble of setting up our server and aggregating all of our frontend and backend code in python scripts. The detailed architecture of our web application is shown in Figure 1.

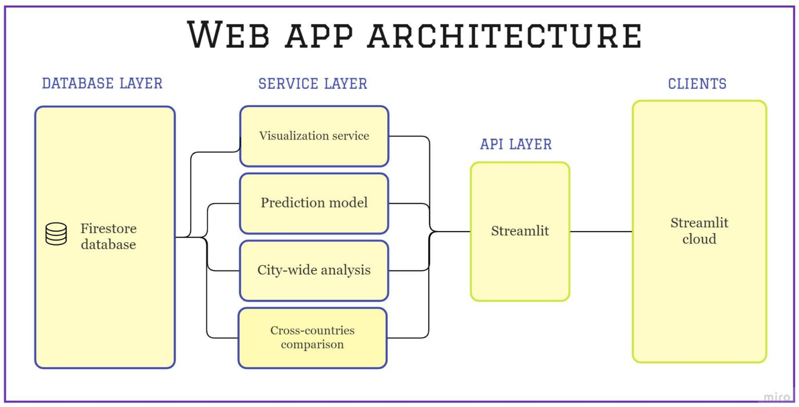

Fig. 1 Web App Architecture Diagram

4. Component of our web

Our web application consists of one main page and five sub-pages which provides five different services(Fig. 2). On the main page, we include the introduction of the web application, source of our datasets, as well as the overview and screenshots of the five pages. Users can navigate through those five pages using the select bar located on the left of the screen. We are going to discuss the details of the five functions on the website later in section 7.

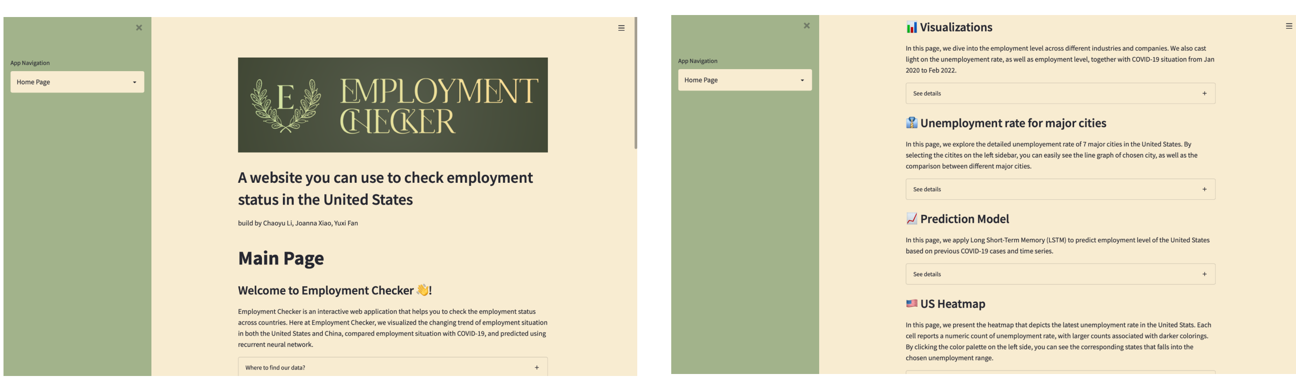

Fig.2 Main page screenshot

5. Data flow

The data flow diagram of the whole project is shown in Figure 3.

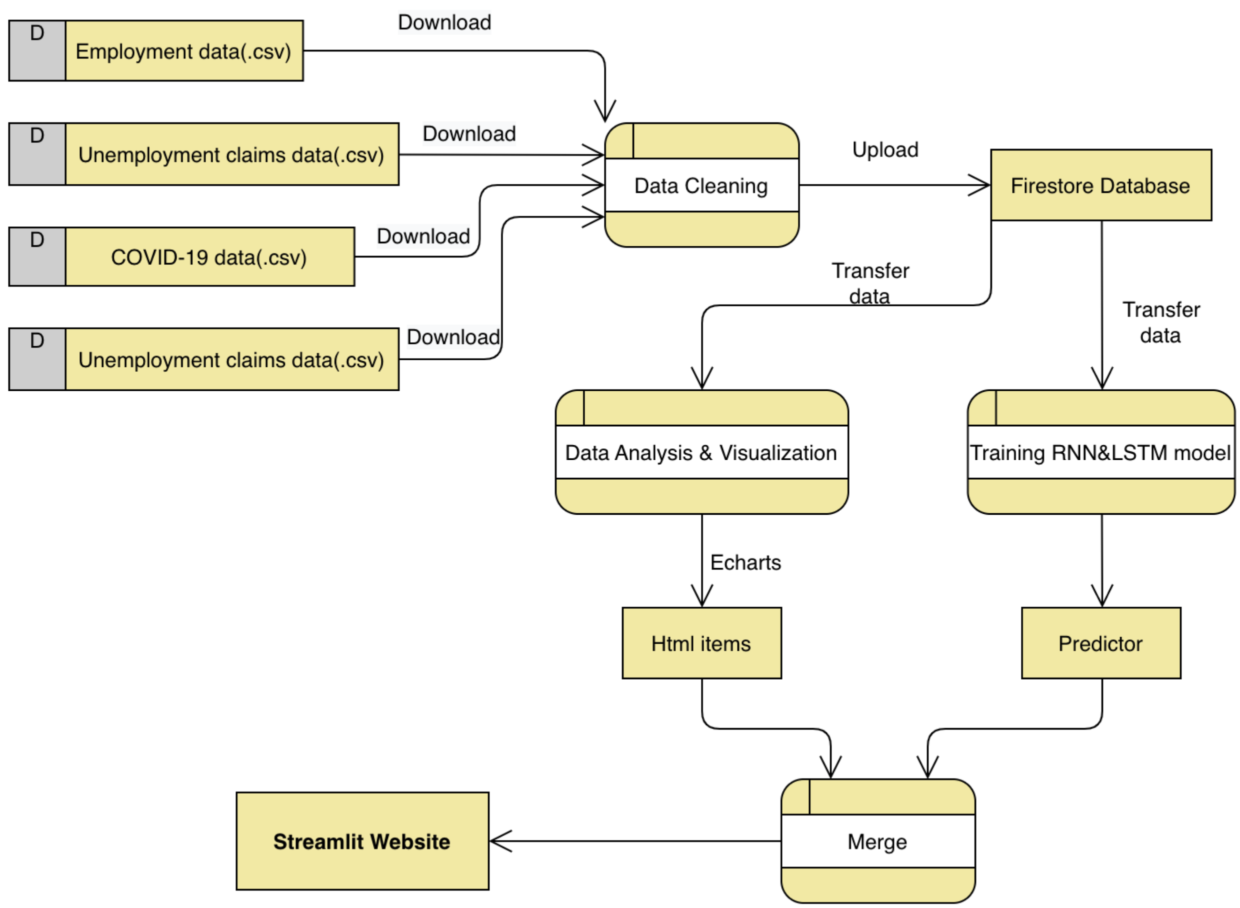

Fig. 3 Data Flow Diagram

The format of the original data is all .csv. We collected national and state employment data in the United States from 2018 to 2021, unemployment data in the United States from 2018 to 2021, and COVID-19 in the United States from 2018 to 2022. Since there is no channel to obtain detailed data related to China, we only collect general data from China for data cleaning. Taking the employment dataset as an example, first, we checked for all missing data. Next, we kept only the attributes of the dataset that we were interested in, such as employment rates and changes in employment across industries. Finally, we package the cleaned data into .json format and upload it to the Firebase Realtime Database. In all subsequent processes, we only read data from the cloud database.

In the data analysis and visualization section, we read data from Firebase Database and visualize the parts we are interested in to show the trends of different data over time series. In this section, we convert some data formats to images or Html to build the website.

In the training predictive model section, we read the data from Firebase Database and convert it to the appropriate format for training RNN and LSTM models. For the data predicted by the model, we package it into the same format as the dataset in the cloud database and upload it to the Firestore Database so that the web page can directly call it.

6. App development process

6.1. Data cleaning

We use the python *pandas* library to help us with data cleaning. First, we dealt with the missing values. Fortunately, there are not too many missing data in the dataset we collected, and most missing data are tolerably missing, which means that we don't have to worry too much about the impact of missing data on our analysis results. Next, we wrapped the data in .json format and uploaded it to Firebase Realtime Database(Fig. 4).

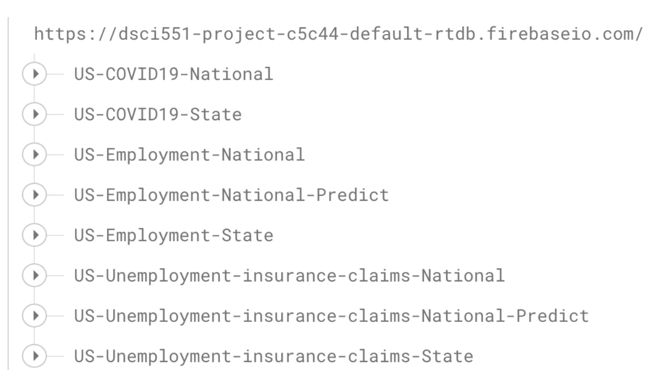

Fig. 4 Overview of datasets in Firebase Database

6.2. Data analysis and visualization

We use *pandas,* and *matplotlib* to help us analyze the data. First, we read the data from the cloud database. For univariate analysis, we plotted it directly over time using *matplotlib*. Figure 5 is an example of our visualization of employment-related data.

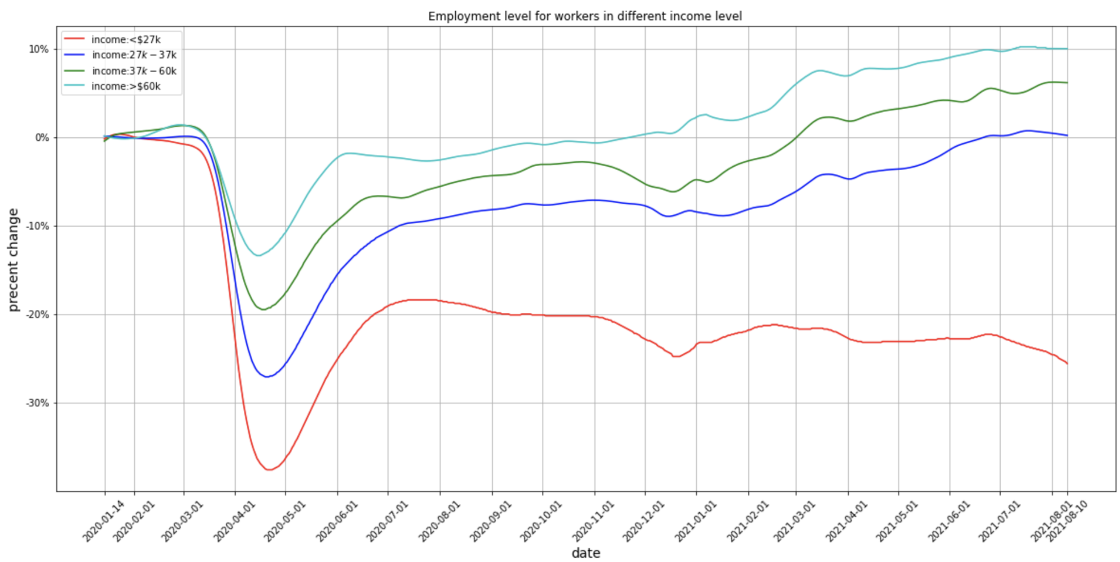

Fig. 5 Changes in employment in different income levels

For multivariate analysis, we first use pandas to extract their common time series and then combine the data we care about into a dataframe. Finally, we use matplotlib to plot the dual y-axes to show their relationship. Figure 6 shows the relationship between unemployment and the number of new COVID-19 cases.

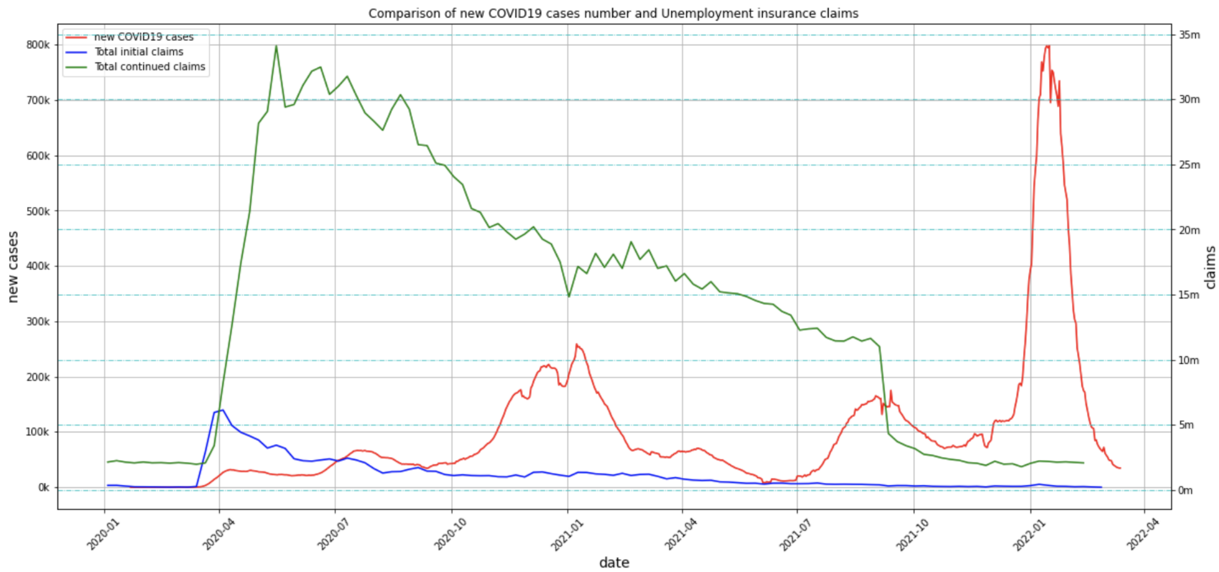

Fig. 6 Comparison of COVID-19 new cases and unemployment claims

We also analyzed and visualized data for all states. After analyzing these data, we found that the employment situation in the United States is similar to the reality we have personally experienced. This shows that the data is real and meaningful. We hope to let more people know about the employment situation they are facing through our application so that they can have a better plan for their life.

6.3. Training predictor

After understanding the changes in the employment situation in the past, we wanted to predict the changing trend of the employment situation for a period of time in the future through the past data, because it will be a very direct help for users. We trained machine learning models for COVID-19 new cases, employment, and unemployment separately to help us predict future data. In this section, we use the *PyTorch* framework to help us build neural network models quickly. We also imported the *optuna* library for automatic parameter tuning.

First, we designed an RNN network to make univariate predictions on COVID-19 data. We only added a hidden layer of 100 neurons. In this model, the time series is the only variable, and the number of new COVID-19 people is the target. According to Figure 7, we found that the predicted curve is very similar to the true curve, which shows that our prediction works well. Moreover, we observe that the model predicts that in the summer of 2022, the daily number of new cases of COVID-19 will stabilize at 0-1000, which is also in line with reality (assuming no new variant disease strains appear).

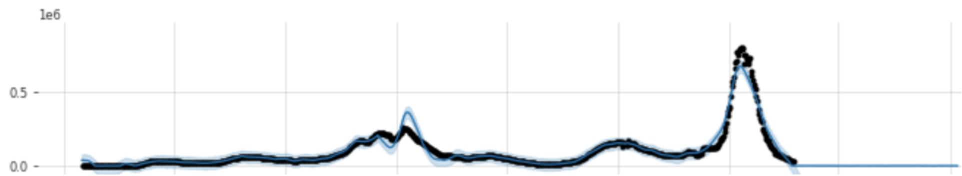

Fig. 7 COVID-19 prediction curve(blue) and actual curve(black)

In addition, we designed an LSTM network to make bivariate predictions on employment data and unemployment data. Here, the time series and COVID-19 new cases are both the variables, and the employment and unemployment data are the targets.

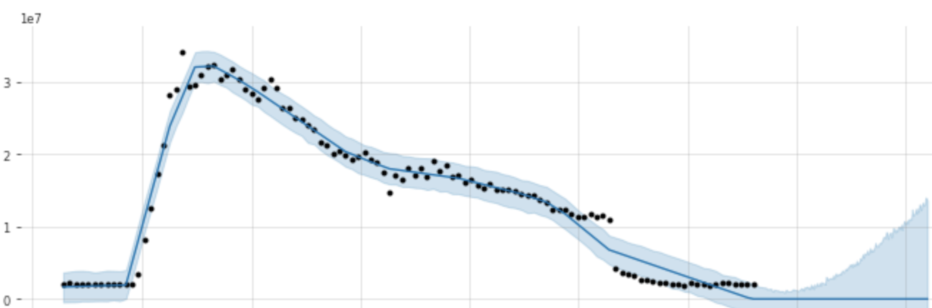

Fig. 8 Unemployment prediction curve(blue) and actual curve(black)

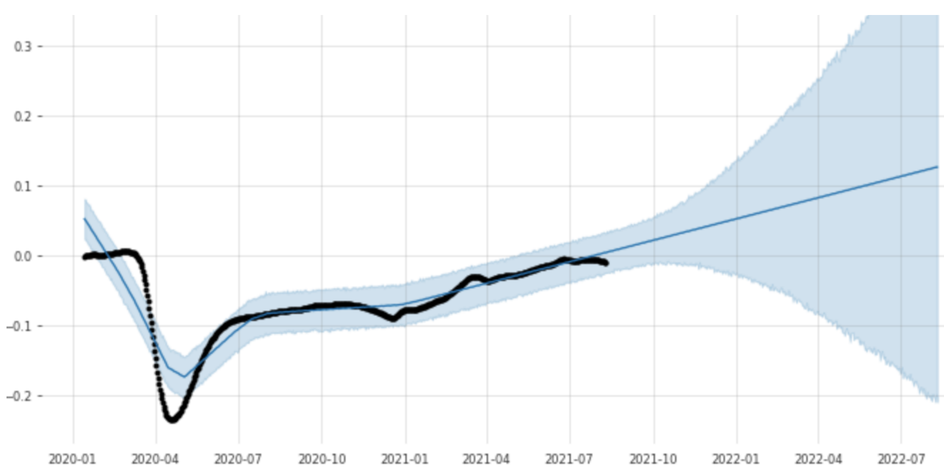

Fig. 9 Employment prediction curve(blue) and actual curve(black)

6.4. Website building

We implemented all the elements needed for the website locally and then assembled them into the website. For complex components, we first used *echarts* to implement because *echarts* can use JavaScript to achieve many complex functions. Then we rendered these components into Html files, then we combined these Html files into our website built with *streamlit*.

7. Main functions of the app

There are in total five functions we provided in our web app (Fig. 10): data visualization, unemployment rate display and comparison for major cities in the United States, employment level change prediction, US unemployment rate heatmap, and the unemployment rate comparison between China and the United States.

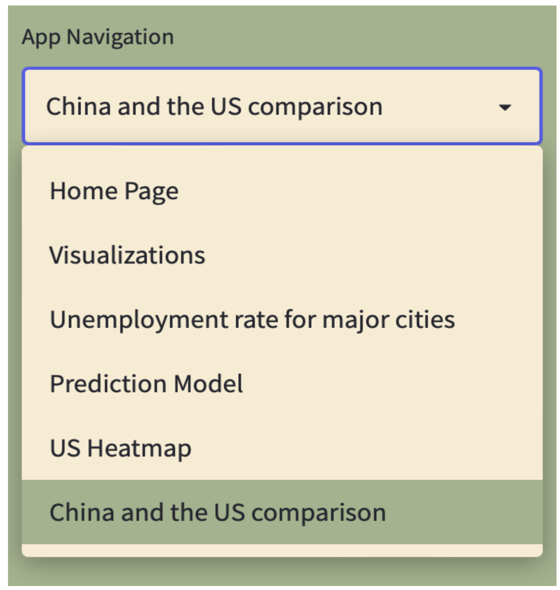

Fig 10. Overview of web app functions

7.1. Data visualization

On this page(Fig. 11), users can select the visualization they would like to see.

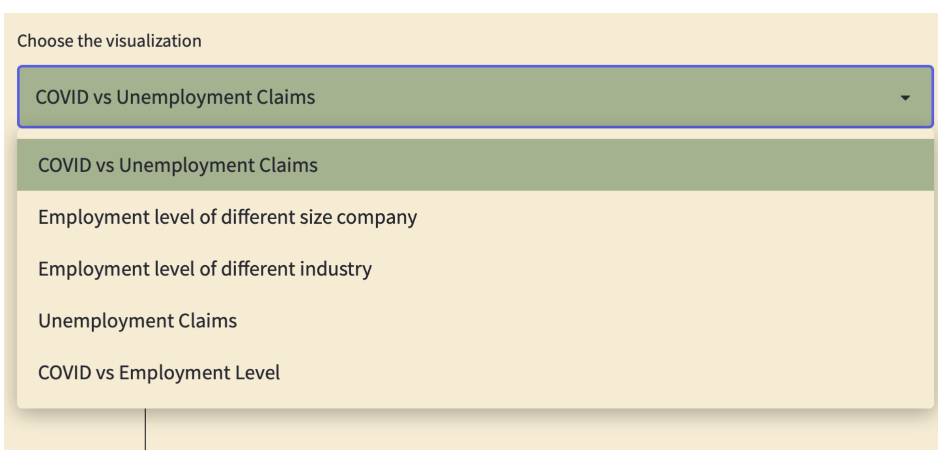

Fig. 11 Overview of the visualization contents

For the COVID vs Unemployment Claims and COVID vs Employment level plot(Fig. 12), users can click the play button to play the change in trend animation or use the slider to look at the trend on any specific date.

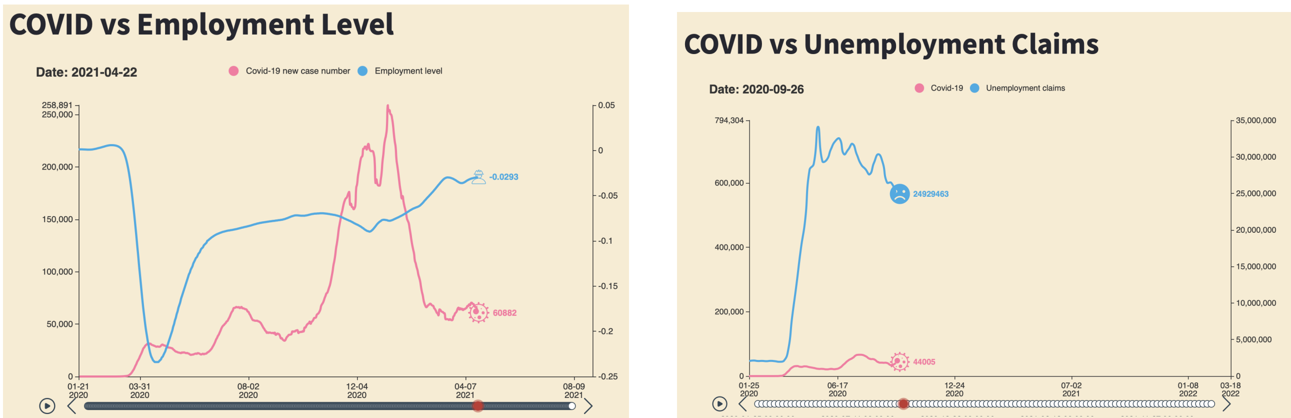

Fig. 12 One frame of each trend animation

For the unemployment insurance claims, employment levels of different size companies, and different industry graphs, users can see the change in unemployment situation in different industries, different size companies, and different types of unemployment insurance claims(Fig. 13). Users can click on the diamond shape icons to only display the curve they want to see. They can also use the slider to specify the date they are interested in.

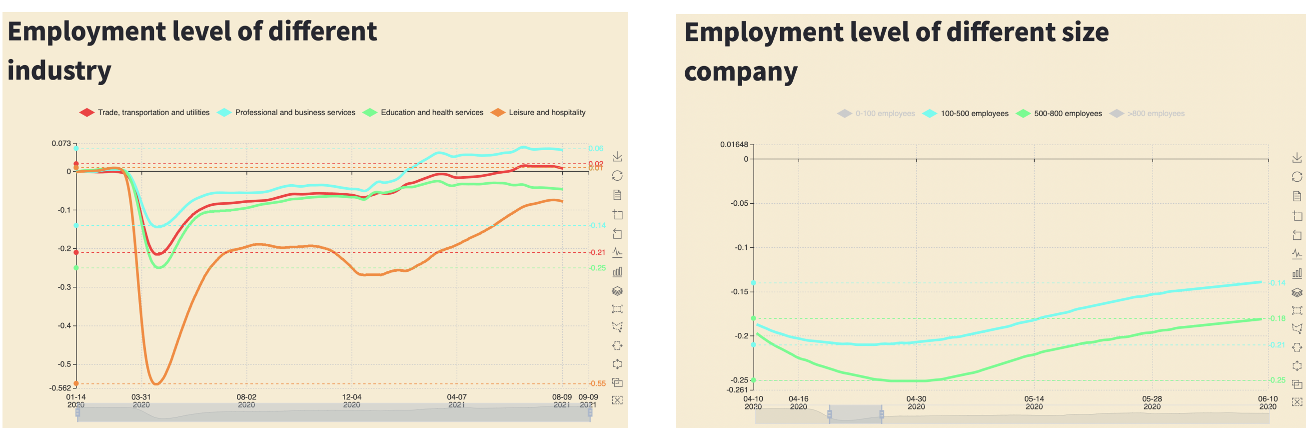

Fig. 13 Example of employment curve

7.2. The unemployment rate for major cities

Users can use the select bar under "city" to choose the unemployment trend of the city they want to see(Fig. 14). This page provides the comparison function for users who would like to compare the latest unemployment rate of different cities. Users can select the target city from the second multi-select bar and click the "Show comparison" button to display the bar chart.

Fig. 14 Example of unemployment charts

7.3. Prediction model

We have also trained a model to predict the change in employment level in the future. In this section, we display the model visualization as well as an interactive graph(Fig. 15). Users can see the value of employment level change when they select a date.

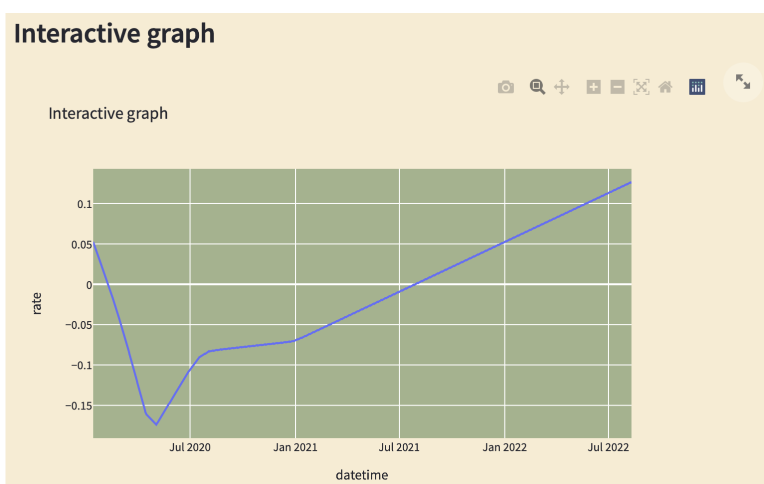

Fig. 15 Interactive graph about predict data

7.4. United States Employment Heatmap

On this page, we display the newest changes of employment rate of different states on the United States map. When users click the state on the map, it will show the newest changes of employment rate.

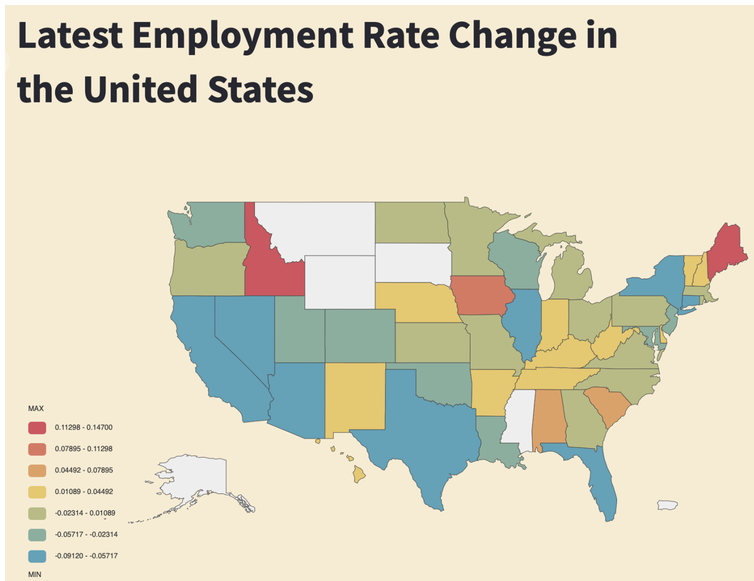

Fig. 16 Interactive graph about predict data

7.5. China and the United States unemployment rate comparison

This page is for users who are interested in employment situations in China and the United States. We use water drop graphs to visualize the unemployment rate and users can use the radio button to select the year.

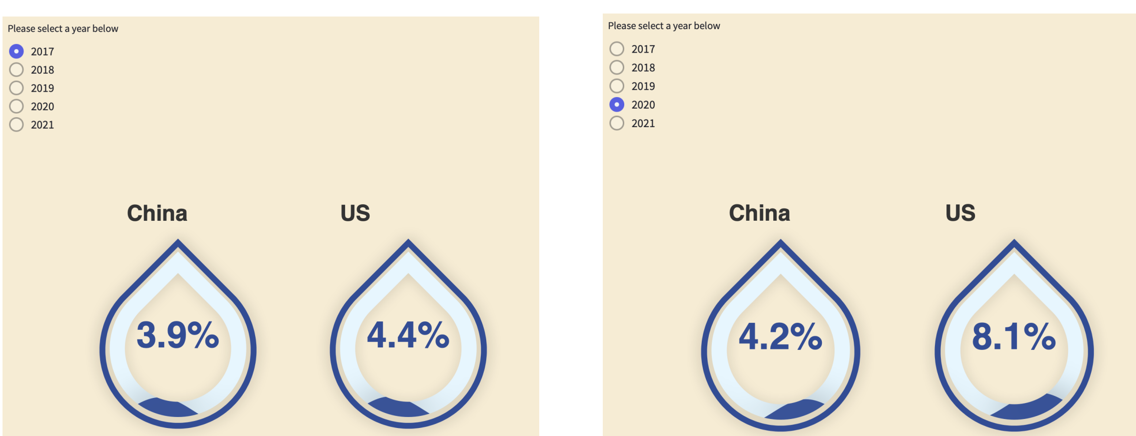

Fig. 17 Interactive graph about predict data

8. Reflection

Through a thorough analysis and deep dive into the current situation of employment across countries and COVID-19 pandemic, we discovered that the employment situation is closely related to COVID-19 cases. To be more specific, as COVID-19 cases increase, the overall employment rate decreases correspondingly. As for employment level across industries, it's quite reasonable to observe the decreasing trend in leisure and hospitality industries, while only a mild drop for education and health services. As regards the employment situations in major cities, we can easily tell that the job market has gradually recovered based on the decreasing line of unemployment rate, although up till now some states may recuperate at a lower speed. Overall, employment has recovered gradually in the wake of COVID-19 pandemic, and through this in-depth study process, we gained a better understanding of the current job market as well as future employment trends.

9. Challenges faced

9.1. Parameter tuning

Since we used RNN networks as our prediction models, parameter tuning becomes an unavoidable challenge. We only considered one or two variables when training the model, but the dataset we used is from reality. A problem here is that changes in the employment rate or unemployment rate, in reality, are often affected by many different factors, so it is difficult for our model to fit a curve that is particularly similar to the actual situation. To alleviate this problem, we tried to use the *optuna* library to implement automatic parameter tuning. It turned out that although the final result is not perfect, our model showed relatively good performance.

9.2. Limitation of streamlit

We used *streamlit* as our website building platform. *Streamlit* is really handy on publishing sites, and it also provides us with a lot of packaged tools. However, when we needed to do some complex component design, such as animation, map display, and other functions, we found that the tools provided by *streamlit* cannot help us. Our solution is that we used *echarts* to implement those complex components first. Because the language used by *echarts* is JavaScript, we can implement almost any component we need. Next, we rendered these components into Html files and then imported these files into the python file of *streamlit*.

10. Team member and responsibility

Name	Responsibilities
Chaoyu Li	Data cleaning, data analysis and visualization, prediction model implementation, website components implementation.
Joanna Xiao	Data collection, data analysis and visualization, UI design, website components implementation
Yuxi Fan	Data collection, data cleaning, database maintenance, website components implementation, website construction

Appendix:

Code Link:
https://drive.google.com/drive/folders/1q9MZaK_4YSZn63KitdIaKfm0L7Xgp7pA?usp=sharing