Leveraging Social Media Data to Inform Family Caregiving:

The User Guide

Team SLUG (4):
Shivani Parekh, Lucas Rodriguez, Urmi Lalchandani, Gem Gasca
Client: Dr. Yong Choi

Contents

Preface

Readme

This is the user guide for utilizing and understanding our research project on family caregiving. This guide will give a broad overview of our project, the basic functionalities of the different aspects of the beta version, and information on how to expand the project to suit different research needs.

Intended Audience

This user guide is intended for researchers and developers interested in learning more about our caregiving research project or adding to it. Researchers using our datasets and machine learning results can learn more about using our resources and how the project was conducted. This includes documentation for our natural language processing (NLP) and information on how the data was obtained.

Developers working on this project in the future can use this guide as a starting point for our approach and our code. The code can be found on our Github page and also includes comments to aid in understanding it.

Related Documentation

- NRCLex Emotion Analysis: https://pypi.org/project/NRCLex/
- NLTK: https://www.nltk.org/
- Gensim library: https://radimrehurek.com/gensim/
- pyLDAvis library: https://pyldavis.readthedocs.io/en/latest/readme.html

---

Product Overview

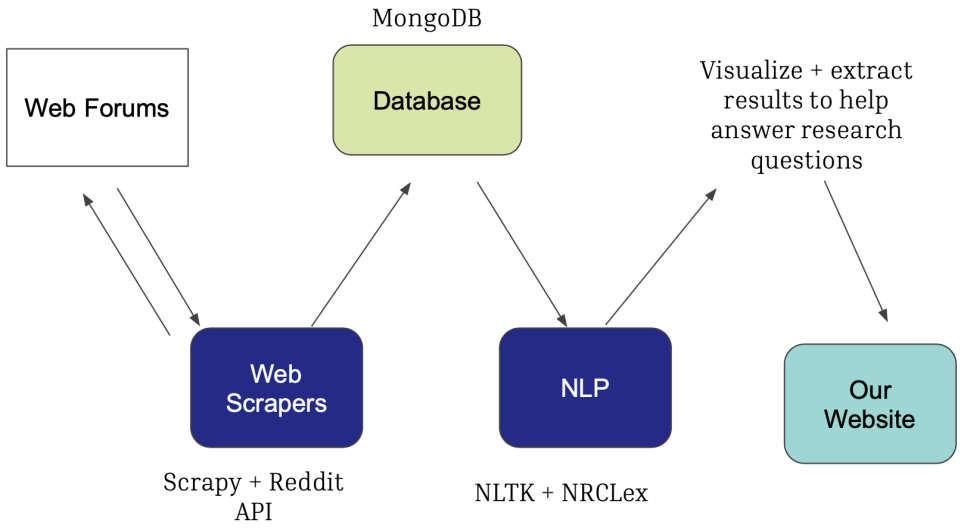
Background

Social media has grown exponentially in its use for health-related purposes and, in turn, has become an untapped data source for health research. Although various research studies have been conducted around offering better support to patients, caregivers are often neglected when it comes to understanding their needs and priorities. Our project aims to harness the untapped data from social media to solve this problem.

We created a public data repository from online caregiver discussion forums to enhance the ability to inform family caregiving research. The data was collected by web-scraping posts

from several online forums (AlzConnected, ALSForums, AgingCare, and Reddit) and stored in a database. We also scraped publicly available user data for any future research exploring user-specific questions.

After data collection, we conducted an analysis using machine learning in the form of natural language processing. This includes topic modeling, sentiment analysis, and emotion analysis for each forum that was scraped. Researchers will use the data to answer various questions regarding the unmet needs of caregivers, the health of caregivers, and the effects of the pandemic. The data collected is adaptable and will be made public to create opportunities for collaboration in answering future research questions.



Description

The main components of our project include web scraping, database formation, natural language processing (NLP), computing descriptive statistics about the data, and website creation. The final product is our public data repository and the website illustrating the data analysis performed. Our website consists of the descriptive statistics for each forum, exploratory machine learning and natural language processing, documentation to help others understand the NLP processes followed, and a questions form for outside researchers to contact the team. Using NLP, we were able to perform topic modeling, sentiment analysis, and emotion analysis on each of the following online forums:

- AlsForums:
  - Current caregivers
  - Past caregivers
- AgingCare
  - Caregivers Forum
- Reddit

    - Alzheimers
    - Support for caregivers
    - Caregivers: Providing Help for Loved Ones
    - Caregiving
- AlzConnected
    - Caregivers Forum

The results of the NLP are being used by researchers to analyze the effectiveness of social media data in caregiving research, as well as trying to discover unknown and unmet needs of caregivers.

Approach

The project is broken down into two parts: the web scraping and the NLP. First, we had to gather the data by traversing the forums and scraping all the relevant data into a database. To web scrape, we used Scrapy, a web scraping framework that crawls websites and follows links. Scrapy, which is a Python library, gets the data from all the pages of interest recursively. We can then send the data to the database one post at a time, as the data came in. We used MongoDB for our database and the PyMongo library to interact with our database in Python.

Once the data was in the database, we could move on to the second part of the project. Using Google Colab, we read in the data from the database (using PyMongo) and put it into a pandas DataFrame. From there, we gathered descriptive statistics for each dataset and performed various NLP techniques. These techniques include topic modeling, sentiment analysis, and emotion analysis. We also used different techniques for visualizing this NLP.

Our NLP results, documentation, and some other functionalities were then put onto our website. Our website uses Flask to serve the pages and is written in HTML and CSS. There are different pages for each forum with the visualizations for our exploratory machine learning. There is also a question form for communicating with our team. The form sends an email to Dr. Choi with the contents of the question. There is also a labeling webpage hosted on the site to facilitate data labeling. The page is hidden, meaning there is no link to get to it from our website. This will allow Dr. Choi and his team to label data and write that information back to the database. This could be used in the future to train machine learning models for better results.

Technical Specifications

As mentioned above, we used Scrapy to web scrape the various forums. Since this framework recursively scrapes the data we want, we had to specify the data we wanted and

which links to follow. We did this using CSS/HTML tags and Xpaths. This resulted in the data below:

AlzConnected

- post_id: the post_id is a field that can be used to link posts to each other. All posts within the same thread have the same post_id.
- date: the date and time of the post (between 11/29/2011 - 02/18/2022)
- title: the title of the post
- body: the body of the post
- reply: True if the post is a reply, False if the post is the original post in the thread
- user_name: the username of the post author
- user_date_joined: the date the user joined the site (between 08/17/2011 - 02/18/2022)
- user_num_posts: the number of posts the user has made on the site
- url: the URL to the post
- support_type: Some posts may have a support_type field. These were added through manual labeling and designate the support type that the post is requesting

AlsForums

- post_id: the post_id is a field that can be used to link posts to each other. All posts within the same thread have the same post_id
- date: the date and time of the post (between 05/17/2003 - 02/16/2022)
- title: the title of the post
- body: the body of the post
- reply: True if the post is a reply, False if the post is the original post in the thread
- user_name: the username of the post author
- user_date_joined: the date the user joined the site (between 04/24/2003 - 02/16/2022)
- user_num_posts: the number of posts the user has made on the site
- user_reason_joined: the reason the user joined - often abbreviated | CAN BE EMPTY
- user_diagnosis: the date the user or user's person was diagnosed | CAN BE EMPTY
- user_country: the user's home country | CAN BE EMPTY
- user_state: the user's home state | CAN BE EMPTY
- user_city: the user's home city | CAN BE EMPTY
- url: the URL to the post

AgingCare

- post_id: the post_id is a field that can be used to link posts to each other. All posts within the same thread have the same post_id.
- title: the title of the post
- body: the body of the post
- date: the date and time of the post (between 10/18/2007 - 04/06/2022)
- user_name: the username of the post author
- reply: True if the post is a reply, False if the post is the original post in the thread
- keywords: the keywords selected for main post by user
- url: the URL to the post

Reddit

- post_id: The ID of the post
- date: The date and time of the post
- title: The title of the post
- body: The body of the post
- num_upvotes: The number of upvotes the post received
- num_downvotes: The number of downvotes the post received
- reply: True if the post is a reply, False if the post is the original post in the thread
- user_name: the username of the post author
- url: the URL to the post

To connect to our database during reading and writing, we used a connection string. It is sensitive information because anyone with access to our connection string can access the database. So, we tried to keep it private.

The data we scraped had to be preprocessed with tokenization, stopword removal, and lemmatization using the spaCy library. Then we were able to move on to Topic Modeling and NLP.

For the topic modeling in our Google Colab notebooks, we used a process called LDA (Latent Dirichlet Allocation), provided in the gensim and pyLDAvis libraries. In order to implement LDA, we followed these steps: loading data, cleaning data, tokenizing data, training a model with the data, and analyzing the results. LDA outputs a specified number of topics with words that are most likely to belong to those topics. It uses conditional probability to group words from documents. The visualization package used, pyLDAvis, allows us to interpret the individual topics and better understand their relationships.

Sentiment Analysis was done using NLTK's SentimentIntensityAnalyzer and NaiveBayesClassifier. The first step was to mark each post as positive or negative. We did this by using the SentimentIntensityAnalyzer to get the polarity of text, which is a value from [-1, 1] with -1 being the most negative and 1 being the most positive. Neutral posts (value of 0) are marked as Negative during the sentiment analysis. We also got the subjectivity of each post, which is a value from [0, 1] with 0 being the most objective and 1 being the most subjective. Labeling posts as positive or negative is done using VADER (Valence Aware Dictionary for Sentiment Reasoning). This model takes into account the general sentiment of a post (polarity) and the intensity of emotion. The model can also understand context at a basic level when analyzing words. Once that was done, we used NLTK's Naive Bayes Classifier to observe which words are linked to negative and positive sentiments. This output shows us how much more likely a word is to be associated with either a negative or positive sentiment.

Emotion Analysis is another form of natural language processing that aims to classify text as a specific emotion, like joy or anger, as well as the severity of the emotion. The Python library we used is called NRCLex. This package measures the following emotional effects:

- fear
- anger
- anticipation
- trust
- surprise
- positive
- negative
- sadness
- disgust
- joy

In order to visualize these results, we found the mean scores for each emotion across a set time period. We also targeted the most frequent emotions of various bodies of text to then find those most dominant.

The questions form on our website works by connecting to an SMTP server and sending an email with the contents of the question. The email that is used is an Outlook email: socialmediafamilycaregivingresearch@outlook.com.

The labeling webpage works by fetching a post that is not yet labeled (determined by the contents of the support_type field, in our case) and displays it on the page. Once the label is submitted, the label is written back to the database for the corresponding post.

Installation

Our public Github repository contains a step-by-step guide on executing our web-scrapers for each individual forum in addition to running our website.

Install and Run Web Scrapers

Requirements:

Install these packages in order to run the web scrapers

- Scrapy
    - pip install scrapy
- pymongo
    - pip install pymongo

- dnspython
    - pip install dnspython

Running Scrapers (AlzConnected, AlsForums, and AgingCare)

The scraped data is written to a MongoDB database as it is scraped. The credentials for this database need to be provided to the program before running; these credentials will be in the form of a MongoDB Connection String. Once the connection string is obtained, paste it into a file called '''credentials.txt''' and insert it into the ./web_scrapers/web_scrapers directory.

Once the credentials are inserted, change into the correct directory and run scrapy:
cd web_scrapers/web_scrapers

Scrapy can be run with either scrapy crawl <spider_name> or scrapy crawl <spider_name> -O <output_file_name> to write the output to a file.

Possible values of <spider_name>:

- alz for this forum
- als for this forum
- als_past_caregivers for this forum:
- ac for this forum
- ac-discussion for this forum
- ac-questions for this forum

NOTE: The scrapers will not run without a valid credentials string in credentials.txt

Options

There are some variables at the top of each spider that can be configured to change the behavior of the web scrapers. To configure the settings, open up the file containing the spider of interest. These spiders can be found in the ./web_scraping/web_scraping/spiders directory. Once the spider is open, you will see these variable at the top of the class definition:

- **name**: This is the name of the spider, do not change it
- **start_page**: The first page of the forum to scrape
- **end_page**: The last page of the forum to scrape
- **write_to_database**: If True, the data is written to the database. If False, the data is not written to the database.
- **collection_name**: The name of the MongoDB collection to send the data to. This needs to be configured to match your database information

There is also a line that should be changed in ./web_scraping/web_scraping/mongoDB.py that should be changed if you are using your own database. The last line should be changed to include your MongoDB database's name.

Running Scrapers (Reddit)

To scrape reddit, PRAW (Python Reddit API Wrapper) requires credentials. Follow these steps to create the credentials file.

1. Create an account with reddit.
2. Go to this link: https://www.reddit.com/prefs/apps and create an app. Save the "personal use script" (14 characters long) and the "secret" (27 characters long).
3. Create a file called '''praw.ini''' and paste the following code in it, inputting your own information. This file should be placed into your working directory.
4. Modify the below options in reddit_scraper.py. Then run this file to scrape and push into the database.

```
[DEFAULT]
client_id=[the personal use script you saved earlier]
client_secret=[the secret you saved earlier]
user_agent=[name of your application]
username=[your account username]
password=[your account password]
```

Options

Some variables can be modified to fit your purposes. These can be found at the bottom of reddit_scraper.py:

- **collection**: The name of the MongoDB collection to send the data to. This needs to be configured to match your database information
- **after_date**: The starting date to scrape posts
- **before_date**: The ending date to scrape posts
- **subreddit**: The name of the subreddit to scrape
- **post_limit**: The maximum number of posts to scrape (default is 1)

Natural Language Processing

The natural language processing code can be found on the Google Colab notebooks. The notebooks will pull the data in from the database and perform the natural language processing. You will need to run all of the cells and, in the Text Preprocessing, upload the "StopWords_Comprehensive.txt."

Website

The website's flask server can be run locally using the instructions below or you can view the hosted website here. You must have a mac or linux machine and have Python installed to run our flask server.

1. Clone the repository onto your computer:

In Terminal type (in the directory you want to clone into):
`git clone https://github.com/ayelrod/Social-Media-to-Inform-Family-Caregiving-Research.git`
Or, download the code from github at:
https://github.com/ayelrod/Social-Media-to-Inform-Family-Caregiving-Research
by hitting the green button that says "Code". Once downloaded you have to unzip the folder.

2. Install flask

This can be done by typing `pip install flask` into the terminal

3. Change into the directory that has the web app

You'll first need to cd into the directory where you downloaded the code.
From there: `cd Social-Media-to-Inform-Family-Caregiving-Research/web_app`

4. Set and run the app

```
export FLASK_APP=SocialMediaNLP
flask run
```

5. In your browser, open up the site

The link will most likely be http://127.0.0.1:5000/

Functionalities

Scrapers

The web scrapers are one of the functionalities available through the Github code. They can be run manually or using a script to scrape posts from the forums of interest.

Database

The database is located on MongoDB and includes all of our datasets. The database is only accessible to Dr. Choi as of right now. Each forum has its own collection within the database. The database currently has 865MB of data but will expand as more posts are scraped in. On our current plan, the database can hold 2GB of data.

Home page

The home page of the website includes background information and the motivation for our project. It also has pictures of each team member. It can be found on the website.

Documentation

The documentation page on the website has information about how the natural language processing was done. This includes information about topic modeling, sentiment analysis, and emotion analysis. Refer to this page when wanting to know how the results of the NLP were obtained.

Ongoing Research

The ongoing research pages have all the results of the natural language processing, for each site. A dropdown menu allows you to select which forum you want to see the results of. These results include the descriptive statistics for the dataset, topic modeling results, sentiment analysis results, and emotion analysis results. Some of the pages link to other pages with a different analysis for that forum (Ex: Pre vs Post Covid analysis).

Questions

The questions tab on the website allows other researchers and site visitors to communicate with our client. Anyone can submit a question to collaborate with our client, ask about

accessing the dataset, and anything else they may want to know. The form will be emailed to socialmediafamilycaregivingresearch@outlook.com.

Labeling

The labeling page allows users to manually label our datasets. On this page, the user will be shown the title and body of a post and they must answer a question about that post. By selecting from a list of answers, they can label the post with what they think is the best answer. An example of this would be labeling a post with the type of emotion present in the post. The label is then written to the database and can be used later to train a machine learning model. The page is not visible on the main website, so users must know the link to access this page.

Troubleshooting

Google Colab Limitations

One of the biggest problems we faced was running machine learning in Google Colab. There is a limited amount of resources allotted to us in the free tier, and this caused problems for us. For example, sentiment analysis on the larger data sets resulted in the runtime running out of RAM and restarting. This was less than ideal, and we solved it by lowering the number of features used in the model. The notebooks generally take a long time to run (sometimes ~3 hours) so be patient.

MongoDB Storage

Our current MongoDB package allows for up to 2 GB of storage (at $9 per month). We have used slightly less than 1 GB (861 MB) to store the data from all of the forums scraped thus far. That being said, if in the future more forums are web-scraped or more data is put into the database, be conscious of the 2 GB storage maximum.

Frequently Asked Questions

Are the Forums continuously web-scraped for new posts?

The forums are not continuously scraped for new posts. The web-scraping code must be run manually. However, there is a script in the Github repository that will run all the web-scrapers in

order to update the database with the most recent posts. This scraper should be run frequently to keep the database up to date.

Are the datasets publicly available?

The datasets are not yet publicly available. The goal is to get them published on our website when the client is ready so that other researchers can use the datasets. The website will also display the proper way to cite our datasets.

Contact Information

Shivani Parekh
shparekh@ucdavis.edu

Lucas Rodriguez
lucrod@ucdavis.edu

Urmi Lalchandani
ulalchan@ucdavis.edu

Gem Gasca
ggasca@ucdavis.edu

Glossary

- **ALS (amyotrophic lateral sclerosis)** - a progressive neurodegenerative disease that affects nerve cells in the brain and spinal cord.
- **Alzheimer's** - a type of dementia that affects memory, thinking and behavior.
- **CSS** - (cascading style sheets) a style sheet language used for describing the presentation of a document written in a markup language such as HTML.
- **Data Labeling** - the process of identifying raw data and adding one or more meaningful and informative labels to provide context so that a machine learning model can learn from it.
- **Family Caregivers** - (also known as "**carers**") are "relatives, friends, or neighbors who provide assistance related to an underlying physical or mental disability for at-home care delivery and assist in the activities of daily living (ADLs) who are unpaid and have no formal training to provide those services."
- **Flask** - a micro web framework written in Python.
- **Gensim** - an open-source library for unsupervised topic modeling, document indexing, retrieval by similarity, and other natural language processing functionalities, using modern statistical machine learning.
- **Google Colab** - a free Jupyter notebook environment that runs entirely in the cloud.
- **HTML** - (HyperText Markup Language) the standard markup language for documents designed to be displayed in a web browser.

- **LDA** - (Latent Dirichlet Allocation) a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar.
- **MongoDB** - a source-available cross-platform document-oriented database program.
- **Naive Bayes classifier** - a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features.
- **Natural Language Processing (NLP)** - a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data.
- **NLTK** - (Natural Language Toolkit) a Python library for building programs to work with human language data
- **NRCLex** - a Python library to measure emotional effect from a body of text.
- **Pandas** - a software library written for the Python programming language for data manipulation and analysis.
- **PRAW** - (Python Reddit API Wrapper) a Python module that provides a simple access to Reddit's API.
- **PyLDAvis** - a Python package used for visualizing topic modeling.
- **Scrapy** - a free and open-source web-crawling framework written in Python.
- **Spacy** - an open-source software library for advanced natural language processing, written in the programming languages Python and Cython.
- **VADER** - (Valence Aware Dictionary and sEntiment Reasoner) a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media.
- **Web-Scraping** - data scraping used for extracting data from websites
- **Xpath** - an expression language designed to support the query or transformation of XML documents.

Appendix

Introduction

Although there have been various research studies conducted around offering better support to patients, caregivers for patients with chronic illness are often

neglected when it comes to understanding their needs and priorities. Our project aims to harness the untapped data from social media to solve this problem.

Social media is a popular tool that is getting more and more accessible, especially with cell phones and personal computers. As a result, oftentimes caregivers and patients turn to online forums for community support and as a way to get answers. These online forums are a largely untapped data source that could give valuable insight into caregiver needs as well as ongoing issues in the caregiving community. The data gathered from these online forums will give researchers more relevant and personal information than they might be able to get from other community outreach.

We plan on scraping the data into an open-source database and performing sentiment analysis and topic modeling that will ultimately allow researchers to identify and address major issues that caregivers are facing on a day-to-day basis. We are gathering data from a few online forums - a caregivers forum for amyotrophic lateral sclerosis (alsforums.com), a caregivers forum for Alzheimer's (alzconnected.org), a caregivers forum for aging family members (agingcare.com), and some subreddits related to caregiving. All of the data and visuals we gather pertaining to this research will be available on a website that we create. The website will be publicly accessible for users to gain insights about our research as well as open for users to post and comment with their own questions or research.

Technology survey

Web Scraping

We are using Scrapy to perform web scraping due to the fact that we need to follow many links on the websites.

Option 1: Beautiful Soup

Pros:

- Simple

Cons:

- Mainly for parsing web pages, not web scraping

Option 2: Scrapy

Pros:

- Supports "crawling" (following links)
- Best for larger scale web scraping

Cons:

- More complicated

Option 3: Selenium

Pros:

- Browser automation
- Robust

Cons:

- Slow
- Isn't necessarily meant for web scraping

Database Technologies

We decided to employ MongoDB for storing our web-scraped data because it is cost-efficient and integrates well with Python that we'll be using for our NLP.

Option 1: Google Sheets

Pros:

- Easy to use, don't have to learn query language

Cons:

- Messy and hard to read for lots of data
- Can't query data or store a lot of data

Option 2: Microsoft Access

Pros:

- Can query data
- Easy to install and integrate (basically anything based in windows)
- Large amount of storage capacity for free

Cons:

- File size limit (limited past 2GB)
- Lacks security

Option 3: MongoDB

Pros:

- Good for storing unstructured non-relational data
- Good support with Python
    - MongoDB stores data in JSONs
    - JSONs equivalent to dictionaries in Python
- Easy to get help on Google since it is widely used

Cons:

- Not free
- No flexible querying

Option 4: Amazon RDS

Pros:

- Cost-efficient
- Scalable - which will be useful for when we scrape more websites

Cons:

- Possible data loss
- Performance not guaranteed

Option 5: Firebase

Pros:

- Good for real-time data - not applicable to our project
- Good for web apps

Cons:

- Limited querying capabilities
- Not free past basic use

Coding Language and Environment

Python has useful libraries that will prove helpful for web scraping as well as data modeling. Jupyter notebook will help us keep our code organized in different cells. Google Colab allows us to collaborate and share our NLP code.

- Python
  - Client asked us to use Python
  - Can use different python libraries to perform analysis on data
- Jupyter Notebook
- Google Colab
  - Allows collaboration
  - Easy way to share our results with clients

Project Management Technologies

We are using JIRA as our project management tool because it allows us to create sprints and it allows us to split up issues nicely. It has a clean visual interface that is easy to use. Github is how we will keep our code organized.

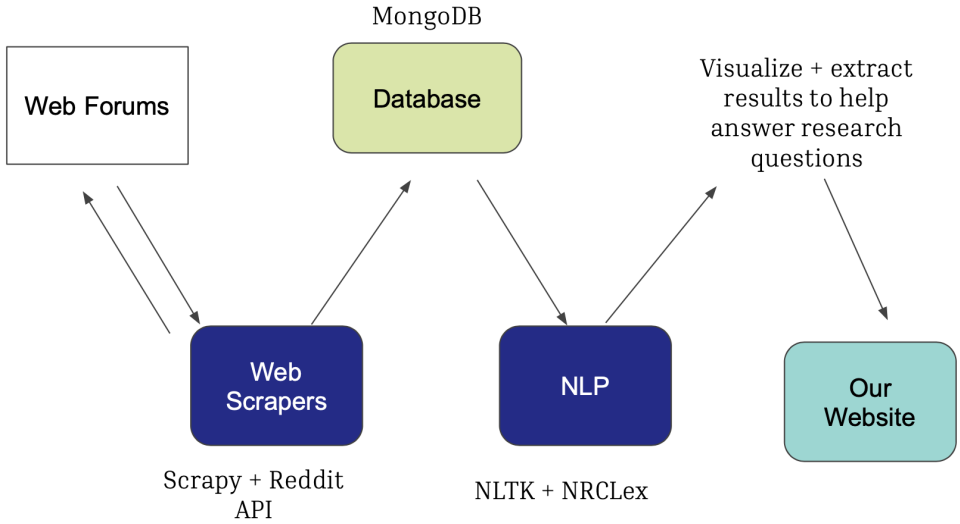
- Github
  - Organize code
  - Version history

Website Development

We chose to use Flask because of the easy Python integration, and because it is better suited for us as beginners in web development. Although Flask has fewer built in features, it doesn't matter much for us since our website will be fairly simple, and we won't have to deal with heavy traffic on the site because our users will mostly be researchers. The website will also need to be able to handle dynamic changes since new research will be posted on it every so often, which Flask does well.

- Flask
  - Pros:
    - Previous familiarity
    - Good for beginners in web development
    - Python so it integrates well with NLP in Colab notebooks
  - Cons:
    - Fewer built-in features
    - Not as good with heavy workloads
- Django
  - Pros:
    - More features
    - Good for complex applications
  - Cons:
    - Not good for dynamic project

System architecture overview



The first step of the project is to scrape the data from the two websites of interest. This will be done using Scrapy, a Python framework for web scraping. Scrapy fetches a web page, given a URL to that page, and then extracts the information we want from it. This information includes the body of these blog posts, the date they were posted, and the user of the post. Scrapy then finds the link for the next page to scrape and fetches it. The process repeats. This process is referred to as "crawling" by the Scrapy framework. The links are checked against the database as they are found to make sure posts aren't scraped twice.

As the data is extracted from the page, it is sent to the database. The database is a MongoDB database hosted in the cloud. This allows multiple users to run the web scraping script at the same time and write to the database concurrently. Once the desired information is scraped from the website, it is packaged into a JSON and added to the database. The goal is to scrape all the posts we can from the websites and have them stored neatly in the database for our second half of this project.

After the database is complete and filled with all the scraped data, we need to do some natural language processing. This will be done in Python using the NLTK library and any other libraries that may be useful for this purpose. The goal is to help answer some research questions related to the needs of caregivers using sentiment analysis, topic modeling, and other NLP methods. The data will be pulled from the database, preprocessed, and then NLP will be performed.

The results of this processing will need to be understood and possibly visualized. With the help of domain experts, these results will help answer the questions we were given.

The data and visualizations we get from our NLP results, as well as our data sets, will be put on a website to share the data with the world wide web. Our website may also include a section where users can ask questions about the data sets and site admins can prepare answers for those questions to post them on the site. In this case, we will use a MongoDB database to store these questions and answers. The questions and answers can be posted on a form through the website and they will be sent to the database. They can then be fetched from the database to display on the website.

Requirements

User stories (functional and non-functional):
* Here, a researcher includes professors, graduate/ PhD students, and other such researchers.

1. As a researcher, I want to help support patients and caregivers based on the results of sentiment analysis.
2. As a researcher, I want to identify the unmet needs of caregivers through data analysis performed on the data in the database so that action can be taken to meet these needs.
3. As a researcher, I want to know how the caregiver's own health, including physical and mental health is by looking at the web-scraped data.
4. As a researcher, I want to know how the Covid-19 Pandemic has affected the sentiments of caregivers and changed their needs through the database.
5. As a researcher, I want to be able to publicly access the web-scraped data online so that this research can be a community effort.
6. As a researcher, I want to be able to query the database so that I can quickly filter the data that I am looking for to answer my questions.
7. As a researcher, I want to be able to see the sentiment analysis of the text posts in the database so that I can understand the experiences that lead to positive and negative sentiments.
8. As a researcher, I want to be able to see the text of posts as well as the replies on that post so that I can study the dynamics between members of the forum.
9. As a researcher, I want to be able to take the web scraper code and modify it for a different forum to answer my research questions.
10. As a researcher, I want to access the website containing research about social media and caregiving through the browser.
11. As a researcher, I want to be able to submit research questions to the website through a text box that will be reviewed by the owner of the website.
12. As a website admin, I want to have access to the questions that have been asked on the website and be able to choose which questions to display on the site.
13. As a researcher, I want to be able to post my own research results about social media and caregiving to the website as a post.
14. As a researcher I want to be able to find answers to some research questions about social media and caregiving on the website.

15. As a researcher, I want to be able to read the documentation of how the natural language processing displayed on the website was performed.
16. As a researcher, I want to see descriptive statistics, topic models, and sentiment analysis of the different forums that were researched.
17. As a researcher, I want to be able to navigate different pages of the website to see the analysis of data from the different caregivers' forums.

Prototyping code

Github Link:
https://github.com/ayelrod/Social-Media-to-Inform-Family-Caregiving-Research/

Technologies employed

Web Scraping

- Scrapy
    - "An open source and collaborative framework for extracting the data you need from websites. In a fast, simple, yet extensible way."

Database

- MongoDB
    - "An application data platform built on the leading modern database."

Code Sharing

- Github
- Google Colab

Web Development

- Flask

Cost analysis

Hosted MongoDB Database (cost taken on by client):

- Currently on free tier
- $9 per month if we exceed 500 MB of data
    - This will most likely be the case

Web Hosting

- Dr. Choi will handle web hosting through UC Davis

Social / Legal Aspect of the Product

A potential concern of scraping data from the online forums includes hindering the functionality of the websites (if one were to put too many requests through and shut servers down). We have scraped data from two websites so far and have not run into any issues with performance. Our client is currently undergoing talks with the boards of some websites to ensure there are no problems for users, in addition to making sure the privacy of users is maintained.

Furthermore, our project will be open source; we plan on having it available on github and through a website so that researchers may obtain and use our code. This means that the project does not have any legal issues in terms of ownership. We wish to provide this data so that researchers can gain answers to their research questions. A given researcher can also take our code and modify it for a different forum, if so desired, and get his or her own data related to other chronic illnesses.