SociaLens Technical Documentation

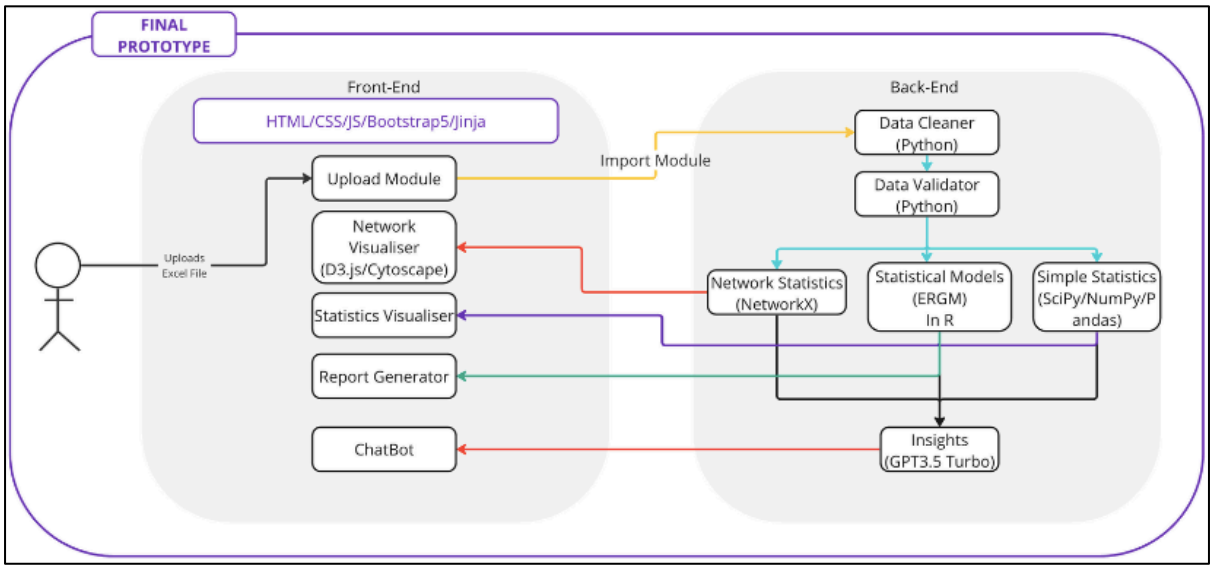
SociaLens Beta (13/5/2024)

Purpose of application, target audience, and user groups

The purpose of SocialLens is to function as a cross-platform tool which can perform comprehensive social network analysis using advanced statistical modelling. Current applications are impaired by an inability to be scalable and in performing advanced statistical modelling. SocialLens is proposed to bridge this gap within the field of social network analysis – by establishing a robust, user-friendly interface that can perform sophisticated statistical and network analyses, whilst also allowing the users to gain a comprehensive insight into this information.

As SocialLens has been designed to be broadly applicable to any kind of social network dataset, the target users can broadly be defined as those who are interested in analysing social networks. This can include professionals from a wide range of sectors such as sociology, health, epidemiology, education, and urban planning.
Although the sectors are from many different areas, all these fields contain a necessity to develop a better understanding of the relationships existing within a network to derive actionable insights. Hence, SocialLens will be a tool to that allows complex statistical and network analyses for all these use-cases.

System architecture



- **Upload Module** → allows the user to upload their dataset of interest
- **Data Cleaner Module** → cleans the uploaded dataset
- **Data Validator Module** → ensures the dataset satisfies pre-specified conditions
- **Analysis Modules** → Simple statistics, network statistics and ERGM analysis is performed on the dataset. The derived information is passed onto the GPT3.5 Turbo component.
- **Export and Visualisation Module** → The derived information from the Analysis module is visualised (Network Statistics, Simple Statistics), presented in a report (Simple Statistics), and can be understood using the ChatBot (Network Statistics, Simple Statistics, Statistical Models).

Deployment

You may download and run the app on your own browser or use our hosted version on AWS.

You can access SociaLens via this elastic IP address hosted on AWS. 107.22.181.150

Downloading SociaLens

Prerequisites

Before proceeding, you need to have the following installed:

- Python 3.6 or higher
- pip (Python package installer)
- Git (for cloning the repository)

Step 1: Clone the Repository

- Open a terminal on your machine.
- Navigate to the directory where you want to clone the repository.
- Run the following command to clone the repository:

```
git clone https://github.com/jayasinp/SociaLens-Alpha.git
```

Step 2: Set Up the Python Environment

- Change to the directory of the cloned repository:

```
cd your-repository-name
```

- Create a virtual environment to manage the dependencies for the project:

```
python -m venv venv
```

- Activate the virtual environment:
- On Windows:

```
.\venv\Scripts\activate
```

- On macOS or Linux:

```
source venv/bin/activate
```

Step 3: Install dependencies

```
pip install -r requirements.txt
```

Step 4: Run the Flask Application

Set the environment variable to tell Flask how to load the application. Ensure you are still in the project directory and your virtual environment is activated:

```
On windows: set FLASK_ENV=development
On macOS or Linus: export FLASK_ENV=development
```

Step 4: Run the app

```
flask run
```

Open a web browser and navigate to http://127.0.0.1:5000/ to view the Flask application.

Troubleshooting Common Issues

ModuleNotFound Error: Make sure all packages are installed correctly via requirements.txt and your virtual environment is active.

Port Already in Use: If you receive an error that the port is already in use, you can specify a different port by using flask run --port=5001 or any available port number.

Using SociaLens

Steps to using application:

1. The first step is to provide the application with the dataset to be analysed. There are two methods to provide the required data:

   If you have dataset in the suitable format [.csv, .xlsx, .xls] → upload the file using the **Upload Dataset** tab. Upon successful upload, the page will contain a populated card for the dataset under the **Uploaded Files** section. Uploaded .xlsx files are split into separate sheets and stored in the “raw_data_objects” directory. The data is also reformatted into .json and .csv format.

   If you require to specify the URL for a website to be analysed for its network of reference → specify the link using the **Scraper** tab. For links specified by this method, the network visualisation is provided immediately on the same page. [Please note: the other tabs and analysis functionality will not be applicable to the scraper method]

2. After successful upload of the dataset, **Explore Data** will require user input to select the dataset to be explored. Following this, it requires user input to select a sheet corresponding to the selected dataset. The selected sheet will be presented in a tabulated format for exploration from the user

3. **Descriptive Statistics** provides simple statistics in tabulated format for a selected dataset and the selected sheet. It also displays the normal distribution plot for the selection.

4. **Network Statistics** provides network statistics in tabulated format for a selected dataset and the selected sheet.

5. **ERGM ->** Open terminal or RStudio. Using the .xlsx or .csv files that are created in the “raw_data_objects” folder, run the following code for .xlsx files:

```
# Load an edge list from an Excel file
file_path <- "path_to_your_file.xlsx"  # Update this to the path
of your .xlsx file
edges <- read_excel(file_path)

# View the first few rows of your edge list
head(edges)
```

If using .csv files:

```
# Load an edge list from a CSV file
file_path <- "path_to_your_file.csv"  # Update this to the path of
your .csv file
edges <- read.csv(file_path)
```

```
# View the first few rows of your edge list
head(edges)
```

Create a network object:

```
# Ensure the network package is loaded
if (!requireNamespace("network", quietly = TRUE)) {
  install.packages("network")
}

library(network)

# Convert the edge list to a network object
net <- network(edges, directed = TRUE)  # Set directed = FALSE if
your network is undirected

# Check the network object
print(net)
```

Fit the ERGM:

```
# Fit an ERGM model
model <- ergm(net ~ edges + mutual)

# Print the model summary
summary(model)
```

6. **Network Visualiser** presents an interactive visualisation for the selected data – presenting each node and its associations visually

7. **Network Statistics Visualiser ->** Data is run through an algorithm for community detection. Use this feature to view all communities from your edge list data. You can also highlight by centrality and degree and inspect which parts of the network are connected or disconnected.

8. **O-Machine** is the generative AI tool which requires user input for dataset, sheet, and type of statistics (network, descriptive, EGRM). After these inputs, the user can type questions relating to the selected dataset to gain insights about the data.