BayesCovid: Uncovering Hidden and Complex Relations of Pandemic Dynamics using an AI-driven System

This application allows users to analyse pandemic dynamics using various Bayesian models. The user can select a CSV file and choose from multiple Bayesian models to process the data, visualise the dependency network, and save the Conditional Probability Tables (CPT) along with the performance of the models.

Used Packages:

To ensure the smooth operation of this application, several Python packages are used.

- pgmpy
- tabulate
- pandas
- networkx
- matplotlib

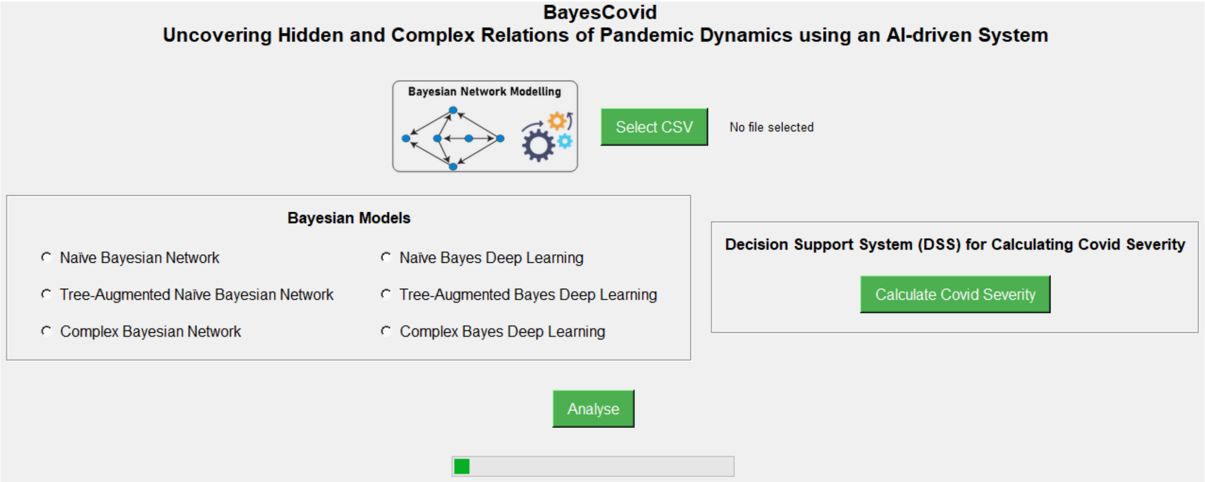
Additional Packages:

- tkinter: Typically included with Python, used for the GUI.
- PyMuPDF (fitz): For handling PDF files.

The **pgmpy** library is utilised for working with probabilistic graphical models, enabling the application to perform complex statistical analyses and inferences. The **tabulate** package aids in creating visually appealing and easy-to-read tabular data outputs, enhancing the presentation of results. For data manipulation and analysis, **pandas** is indispensable, offering robust data structures and functions. The **networkx** library facilitates the creation, manipulation, and study of complex networks and graphs, which is crucial for the application's underlying algorithms. Visualization of data and results is managed using **matplotlib**, providing a wide range of plotting capabilities. Additionally, the **tkinter** package, typically bundled with Python, is employed to create the graphical user interface, ensuring user-friendly interaction with the application. For handling and processing PDF files, **PyMuPDF** (imported as **fitz**) is incorporated, enabling efficient PDF manipulation and integration within the application. These packages collectively contribute to the functionality and usability of the application, making it a comprehensive tool for the intended tasks.

1. Functional Aspects

1.1. User Interface: A user-friendly interface is developed using Python language.



BayesCovid

Uncovering Hidden and Complex Relations of Pandemic Dynamics using an AI-driven System

Bayesian Network Modelling
Select CSV
No file selected
Bayesian Models
○ Naïve Bayesian Network
○ Tree-Augmented Naïve Bayesian Network
○ Complex Bayesian Network
○ Naïve Bayes Deep Learning
○ Tree-Augmented Bayes Deep Learning
○ Complex Bayes Deep Learning
Decision Support System (DSS) for Calculating Covid Severity
Calculate Covid Severity
Analyse

1.1.1. Implementation of Bayesian Network Models:

- **CSV File Selection:** A button labeled "Select CSV" allows users to browse and select a CSV file. The selected file's name is displayed.

- **Model Selection:** Six radio buttons are provided for users to select one of the following Bayesian models:

  - Naïve Bayesian Network
  - Tree-Augmented Naïve Bayesian Network
  - Complex Bayesian Network
  - Naïve Bayes Deep Learning
  - Tree-Augmented Bayes Deep Learning
  - Complex Bayes Deep Learning

- **Analysis Execution:** An "Analyse" button that starts the processing of the selected model with the selected CSV file.

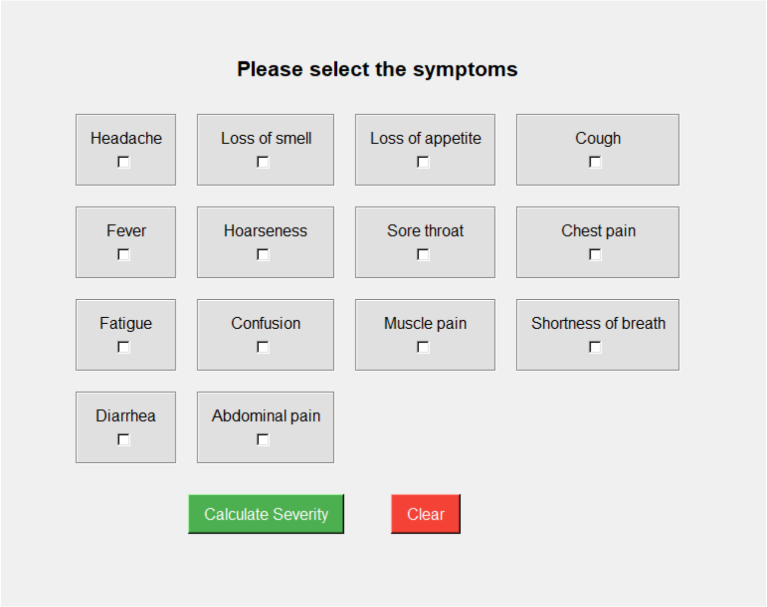
- **Progress Indicator:** A progress bar that shows the processing status.

- **Plot Display:** Displays the dependency network plot generated by the selected model.

- **CPT Output:** Saves the Conditional Probability Table (CPT) output to a file.

1.1.2. Decision Support System for Calculating Covid Severity

This component of the application assists clinical staff in calculating the severity of COVID-19. This feature assists in selecting the detected symptoms that the patient exhibits and subsequently determines the severity of COVID-19.



Please select the symptoms

Headache □	Loss of smell □	Loss of appetite □	Cough □
Fever □	Hoarseness □	Sore throat □	Chest pain □
Fatigue □	Confusion □	Muscle pain □	Shortness of breath □
Diarrhea □	Abdominal pain □		

1.2. File Handling:

- Users can select a CSV file for processing.
- The tool supports reading and processing the selected CSV file with the specified Bayesian model.

1.3. Model Processing:

- Executes applications corresponding to the selected Bayesian model by executing the main file called "**BayesCovid.ipynb**" in jupyter notebook.
- Processes the data using Bayesian network algorithms and deep learning models.
- Generates and displays the dependency network plots.
- Saves the CPT output to a file.
  - **Note: To see the whole information of CPT:**

    Go to the following address in the computer:

    C:\Users\Umit\AppData\Local\anaconda3\Lib\site-packages\pgmpy\factors\discrete

    # Open file "pgmpy/factors/discrete/CPD.py"

    # make this line deactivate by add "#"

    --> # cdf_str = self._truncate_strtable(cdf_str)

2. Computational Aspects

2.1. Bayesian Network Models:

- **Naïve Bayesian Network:** A simple Bayesian network with strong independence assumptions between the predictors.
- **Tree-Augmented Naïve Bayesian Network:** An extension of the Naïve Bayesian Network that allows for dependencies among the predictors by adding a tree structure.
- **Complex Bayesian Network:** A more sophisticated Bayesian network that can capture more complex dependencies among the predictors.

2.2. Deep Learning Models:

- **Naïve Bayes Deep Learning:** Combines deep learning techniques with a Naïve Bayesian Network.
- **Tree-Augmented Bayes Deep Learning:** Combines deep learning techniques with a Tree-Augmented Bayesian Network.
- **Complex Bayes Deep Learning:** Combines deep learning techniques with a Complex Bayesian Network.

2.3. Execution and Processing:

- The tool executes the selected models.
- Selected model processes the CSV file, and generates the dependency network plot.
- The **nbconvert** library captures the output of the model execution, including the CPT.

2.4. Plot Generation:

- Uses the **networkx** and **matplotlib** libraries to generate and visualise the dependency network plots.
- Converts the plots to PDF format for display.

2.5. Output Handling:

- The CPT output is captured from the notebook execution and saved to a text file.

3. Scalability Aspects

3.1. Data Handling:

- The tool is designed to handle large CSV files efficiently.
- Uses pandas for efficient data manipulation and processing.

3.2. Model Execution:

- The selected model are executed using the **nbconvert** library, which allows for scalable and efficient execution of the notebooks.

3.3. Parallel Processing:

- The tool can be extended to support parallel processing of multiple models or datasets simultaneously.

3.4. Cloud Deployment:

- The tool can be deployed on cloud platforms to leverage scalable computing resources.
- Supports integration with cloud storage services for handling large datasets.

3.5. Extensibility:

- The tool is designed to be extensible via the big data framework, such as Apache Hadoop, Apache Spark, allowing for the addition of new models and functionalities.
    - Please refer to ***HADOOP 3.2.0 installation.pdf*** and ***Apache Spark installation.pdf*** files to install big data system to deploy **BayesCovid** over them.
- Modular architecture enables easy updates and maintenance.

3.6. Performance Optimization:

- Efficient algorithms and data structures are used to optimise the performance of the tool.
- The tool can be further optimised for specific use cases and datasets.

4. Demo: For a detailed demonstration of the application and its features, please refer to the demo video available at this link.

- https://youtu.be/7j36HuC9Zto