Software Requirements Specification
Version 1.0

ViroShield

Theme: FluNet

Category: Big Data Processing Using Data Science Technologies

Contents

1.1 Background and Necessity for the Application

Many times, deaths occur due to unknown reasons and people predict many opinions on these. However, research says that the major rise in deaths is due to the spread of viruses which no one is aware of and thus, becomes a serious problem. The World Health Organization (WHO) has concluded different surveys which requires to be performed as and when there is a rise in the death case and save lives of many.

'Influenza' is a viral disease and a person infected by this can show the symptoms of fever, headache, cough, cold, fatigue, and so on. Some symptoms last for a week while some result in a terrifying cause that leads to death.

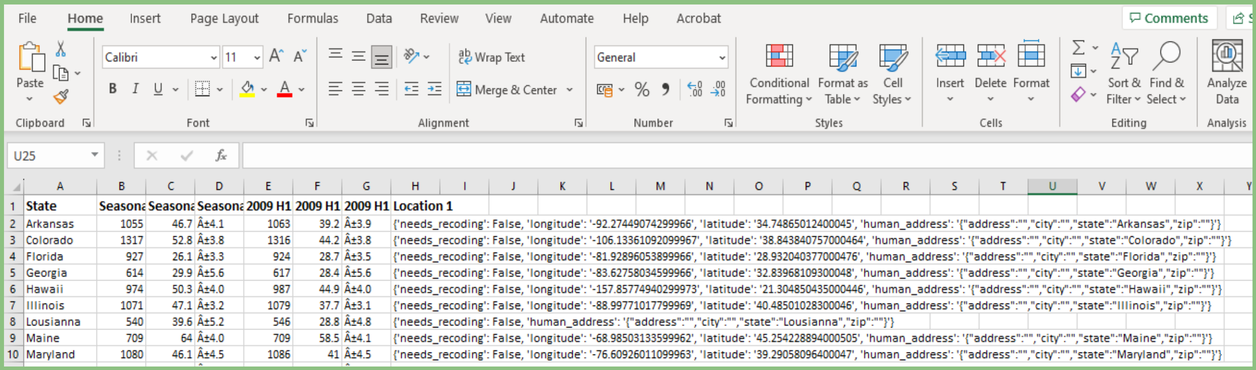
Therefore, to overcome these challenges, a good and reliable application will help predict the desired results, where the patient is aware of the problem he/she is going through and accordingly start the medication.

1.2 Proposed Solution

'ViroShield' aims to predict results that will help individuals understand the cause of health issue and death. By leveraging the power of Big Data techniques, the application can generate outputs based on the data and display the status of the patient's health.

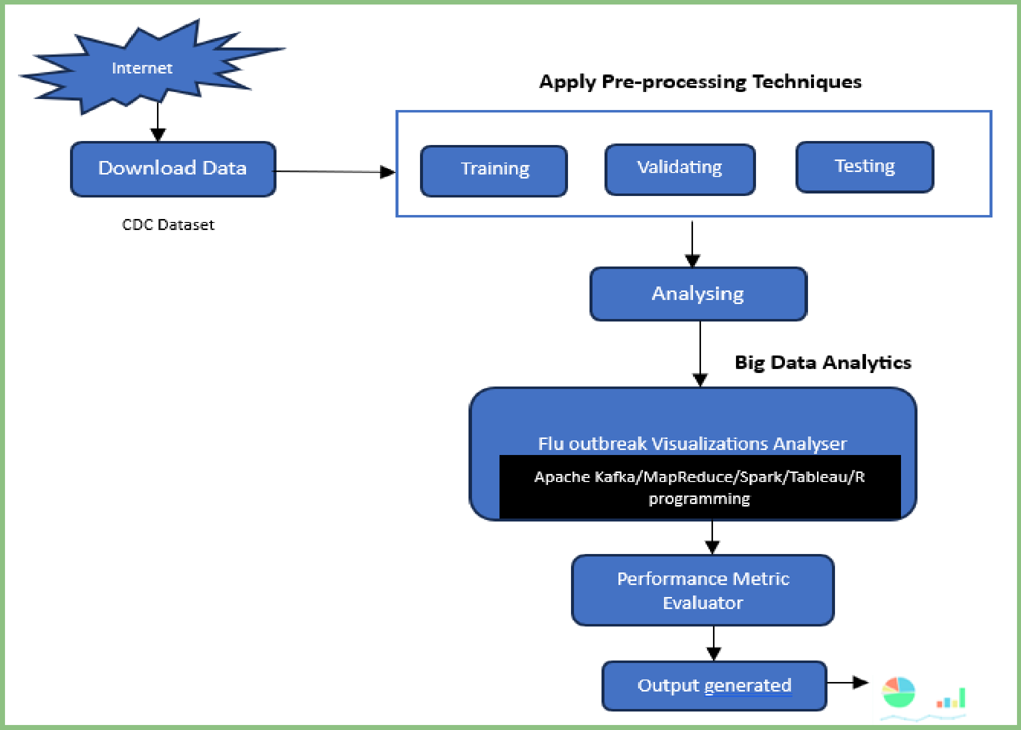
Big Data technologies offer significant advantages for predicting influenza by enabling the efficient processing, integration, and analysis of large and diverse datasets. It will generate a dashboard result where the predicted values will be displayed based on the trained dataset and bar-graph charts. Use Data Visualizations technique for generating the graphical representation of the desired outputs using Tableau and R Programming. This way, it will generate results and users will be acknowledged with the rise in death cases/health issues on the predicted outputs.

Hint: The sample of the downloaded 'CDC' dataset from Kaggle for implementation purposes is as follows:



State	Seasona	Seasona	Seasona	2009 H1	2009 H1	2009 H1	Location 1
Arkansas	1055	46.7	Â±4.1	1063	39.2	Â±3.9	{'needs_recoding': False, 'longitude': '-92.27449074299966', 'latitude': '34.74865012400045', 'human_address': '{"address":"","city":"","state":"Arkansas","zip":""}'}
Colorado	1317	52.8	Â±3.8	1316	44.2	Â±3.8	{'needs_recoding': False, 'longitude': '-106.13361092099967', 'latitude': '38.843840757000464', 'human_address': '{"address":"","city":"","state":"Colorado","zip":""}'}
Florida	927	26.1	Â±3.3	924	28.7	Â±3.5	{'needs_recoding': False, 'longitude': '-81.92896053899966', 'latitude': '28.932040377000476', 'human_address': '{"address":"","city":"","state":"Florida","zip":""}'}
Georgia	614	29.9	Â±5.6	617	28.4	Â±5.6	{'needs_recoding': False, 'longitude': '-83.62758034599966', 'latitude': '32.83968109300048', 'human_address': '{"address":"","city":"","state":"Georgia","zip":""}'}
Hawaii	974	50.3	Â±4.0	987	44.9	Â±4.0	{'needs_recoding': False, 'longitude': '-157.85774940299973', 'latitude': '21.304850435000446', 'human_address': '{"address":"","city":"","state":"Hawaii","zip":""}'}
Illinois	1071	47.1	Â±3.2	1079	37.7	Â±3.1	{'needs_recoding': False, 'longitude': '-88.99771017799969', 'latitude': '40.48501028300046', 'human_address': '{"address":"","city":"","state":"Illinois","zip":""}'}
Lousianna	540	39.6	Â±5.2	546	28.8	Â±4.8	{'needs_recoding': False, 'human_address': '{"address":"","city":"","state":"Lousianna","zip":""}'}
Maine	709	64	Â±4.0	709	58.5	Â±4.1	{'needs_recoding': False, 'longitude': '-68.98503133599962', 'latitude': '45.254228894000505', 'human_address': '{"address":"","city":"","state":"Maine","zip":""}'}
Maryland	1080	46.1	Â±4.5	1086	41	Â±4.5	{'needs_recoding': False, 'longitude': '-76.60926011099963', 'latitude': '39.29058096400047', 'human_address': '{"address":"","city":"","state":"Maryland","zip":""}'}

The sample architecture can be as follows:



Sample Architecture of the Application

The development phase of the application includes following steps:

1. Downloading data from the dataset
2. Training the dataset
3. Analyzing and Testing
4. Visualizing the results
5. Updating health status to the user

After the CDC data is downloaded from Kaggle which is an open-source repository, it will be trained and tested. A dashboard-based output will be generated. Using Data Visualization techniques different charts/bar-graphs/plots will be generated as a desired output.

1.3 Purpose of the Document

The purpose of this document is to present a trained and tested interactive model titled **'ViroShield'**.

This document explains the purpose and features of ViroShield and the constraints under which it must operate. This document is intended for both stakeholders and developers of the application.

1.4 Scope of Project

'ViroShield' will be used to understand the overall health status of the users' who may be infected by the virus, thereby, helping them understand the cause and start with the medication process as soon as possible. This way many lives could be saved by providing effective and accurate prescriptions.

Using the CDC dataset, data will be downloaded and trained to generate results.

Data Visualization process will be used for generating bar-graphs thereby helping the user to understand health status.

1.5 Constraints

The application may not work if data privacy is misused as appropriate security measures are required to protect personal data from unauthorized access, use, or disclosure. Also, when sharing data with third parties, data protection should be assured. Data retention and transparency is a must.

1.6 Functional Requirements

Following are the functional requirements:

i. **Load data into Hadoop cluster –** The data should be loaded into the Hadoop cluster making it ready for analytics.

ii. **Modeling Strategies –** Data mining techniques are used for predicting the values based on the dataset used.

iii. **Training. Validating, and Testing the Dataset –**

Data downloaded requires to be trained where the data should be extracted, explored, and analysis should be performed on the relevant features in order to generate accurate results.

iv. **Big Data Analytics -** With the advent of Big Data, data analysis techniques have evolved to handle large and complex datasets. Technologies like distributed computing, parallel processing, and specialized frameworks (for instance, Apache Hadoop, Spark) enable efficient analysis of massive datasets. Data will be analyzed after the training is done to predict the results on the dashboard for comparison purposes.

v. **Data Visualization –**

R/Python offers numerous packages and libraries for creating visualizations, allowing users to generate a wide range of plots, charts, graphs, and interactive visualizations to explore and present data effectively. Dashboards provide a visual representation of key data and metrics, allowing users to monitor and understand the influenza situation effectively.

1.7 Non-Functional Requirements

There are several non-functional requirements that should be fulfilled by the application as follows:

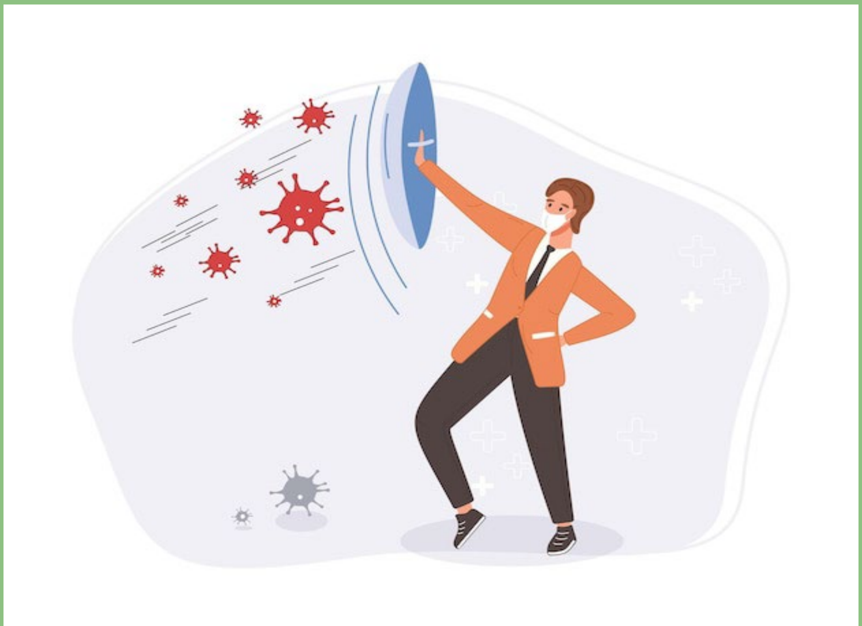
1. **Accessible:** The application should be easy to use and that allows easy interaction with data.

2. **Integration:** The application should be able to integrate with various data sources, both internal and external, to gather comprehensive data for Influenza. It should support data integration from diverse applications, databases, and data formats.

3. **Performance Efficiency:** The application should perform in a proper time constraint that would reflect accurate output.

4. **Secure:** Data should be secured where it should be encrypted. Non-authorized users should not be allowed to download or use.

5. **Reliable:** The application should be reliable, consistent, and provide accurate output (predicted values and visualized data based on bar-graphs) to the end user.



These are the bare minimum expectations from the project. It is a must to implement the functional and non-functional requirements given in this SRS. Once they are complete, you can use your own creativity and imagination to add more features if required.

1.8 Interface Requirements

1.8.1 Hardware

Intel Core i5 Processor or higher
8 GB RAM or above
Color SVGA
500 GB Hard Disk space
Mouse
Keyboard

1.8.2 Software

Technologies to be used:

1. **Frontend:** HTML5, JavaScript, CSS3, or any other frontend programming languages
2. **Data Store:** HDFS 3.x, Apache Spark 3.x, Apache Hive 3.x or higher, Apache HBase 2.x, MongoDB 5.x
3. **Programming:** Python 3.x/R 4.x (or higher)
4. **Visualization:** Tableau Desktop 2023.2

1.9 Project Deliverables

You need to design and build the project and submit it along with a complete project report that includes:

- Problem Definition
- Design specifications
- Diagrams such as user journey map
- Source Code
- Test Data Used in the Project
- Project Installation Instructions for configuring Tableau, and Big Data technologies
- Link of published blog
- Link of GitHub for accessing the uploaded project code (Link should have public access.)
- Tableau Dashboard with appropriate access rights or permission to access for testing

The consolidated project must be submitted on GitHub with a ReadMe.doc file listing assumptions (if any) made at your end. Ensure that you provide the GitHub URL where the project has been uploaded for sharing. The repository on GitHub should have public access. Documentation is a very important part of the project hence, all crucial aspects of the project must be documented properly. Ensure that documentation is complete and comprehensive.

You should publish a blog of minimum 2,000 words on any free blogging Website such as Blogger, Tumblr, Ghost or any other blogging Website. The link of the published blog should be submitted along with the project documentation.

In addition, you must submit a video clip showing the actual working of the application. Over and above the given specifications, you can apply your creativity and logic to improve the application.

~~~ End of Document ~~~