Predictive Analysis and Classification of Under-5 Child Mortality Causes Using Machine Learning Models

The Aria International

Abstract

The project aims to develop predictive and classification algorithms to forecast or classify diseases and mortality rates based on historical data. This data contains detailed information on mortality rates caused by various diseases over several years, categorized by countries, organizations, and regions. Additionally, the study will analyze trends and patterns in death rate changes over time for specific areas.

The primary objective is to identify critical diseases that require immediate attention and understand which diseases are contributing most significantly to child mortality. This analysis will provide actionable insights for organizations like the WHO and other health agencies, empowering them with valuable decision-making tools to look over those decease which are causing high death rate and decrease it.

The study employs a variety of machine learning algorithms, including:

For Regression Analysis: Linear Regression, K-Nearest Neighbors (KNN) Regressor, and Decision Tree Regressor.

For Classification Analysis: Decision Tree Classifier, K-Nearest Neighbors (KNN) Classifier, and Gradient Boosting Classifier.

By leveraging these models, the project aims to enhance predictive capabilities and support strategic interventions to mitigate child mortality on a global scale.

Table Of Content

Table Of Tables

Table Of Figures

Keywords

1. Machine Learning
2. Random Forest
3. Regression
4. Classification
5. KNN
6. Prediction
7. Data Visualization
8. Diseases
9. Findings
10. Gradient Boosting Classifier.

1. Introduction

1.1. Background

Child mortality, especially among children under the age of 5, remains one of the most pressing global health challenges. Despite notable progress over the past decades, approximately 5 million children under 5 years of age died in 2020 alone. Many of these deaths were caused by conditions such as pneumonia, diarrhea, malaria, complications during childbirth, and neonatal disorders—conditions that could largely have been prevented or effectively treated with timely and accessible healthcare interventions**.**

1.1.1 The Importance of Health on a Global Scale.

The health and survival of children under the age of 5 are critical indicators of a nation's overall development and well-being. Despite significant advancements, under-5 mortality remains a pressing global issue, with millions of preventable deaths occurring each year. Ensuring good health for populations worldwide is not just a moral obligation but also a strategic priority that underpins the sustainability and growth of nations.

1.1.2 Human Rights and Equity:

Every child has the right to survive and thrive. Reducing under-5 mortality ensures equitable access to healthcare and opportunities for a better future.

1.1.3 Impact on Families and Communities:

The death of a child has devastating emotional and economic impacts on families and communities.

1.1.4 Global Development Goals:

Addressing under-5 mortality is essential for achieving Sustainable Development Goal 3 (Good Health and Well-being), which aims to end preventable deaths of newborns and children under 5 by 2030.

1.1.5 Challenges and Opportunities in Addressing Under-5 Mortality

Reducing under-5 mortality faces significant challenges, including inadequate healthcare access, socioeconomic inequalities, persistent infectious diseases, malnutrition, and geographic disparities, particularly in low- and middle-income countries. Climate change and limited data also exacerbate the issue. However, opportunities exist to address these challenges through technological advancements like predictive analytics, global initiatives such as UNICEF's immunization campaigns, and community-based healthcare models. Focused interventions in maternal care, nutrition, and sanitation, alongside policy reforms and increased funding, can

significantly reduce child mortality. By leveraging innovation, education, and collaboration, we can save millions of lives and improve global child health outcomes.

1.2 The Relationship between AI and Health

Artificial Intelligence (AI) is transforming healthcare by improving efficiency, accuracy, and accessibility in diagnosis, treatment, and prevention. AI-driven technologies, such as machine learning algorithms and natural language processing, enable healthcare professionals to analyze vast amounts of data, identify patterns, and make predictions that were previously unattainable. For example, AI models can predict death rate by diseases over years, disease outbreaks, personalize treatment plans based on patient data, and assist in early detection of illnesses like cancer..

Despite its benefits, challenges such as data privacy, algorithm bias, and the need for robust regulatory frameworks remain. Nonetheless, the synergy between AI and healthcare offers immense potential to address global health challenges, reduce disparities, and improve outcomes for patients worldwide.

Figure 1.2-1 AI Transforming On Health Care

1.2.1 How Machine Learning Is Used in Healthcare

Machine Learning (ML) is a powerful tool for addressing child mortality by identifying risk factors, improving early diagnosis, and enabling timely interventions. Here's how ML is applied to reduce under-5 deaths caused by diseases:

- **Reducing Neonatal Mortality**

Birth Risk Prediction: ML models identify pregnancies at risk of complications, such as preterm births or birth asphyxia, enabling early interventions.

Neonatal Monitoring: ML-powered devices monitor vital signs in newborns, detecting conditions like sepsis or respiratory distress early for timely treatment.

- **Early Detection of Diseases**

Predictive Analytics: ML models analyze health records, environmental factors, and genetic data to predict diseases such as pneumonia, malaria, and diarrhea, which are leading causes of under-5 deaths. Early detection allows for quicker treatment and better survival outcomes.

- **Nutritional Support**

ML algorithms identify patterns in malnutrition data and predict areas with food insecurity, enabling targeted nutritional interventions for at-risk children.

- **Public Health Monitoring and Disease Forecasting**

By analyzing environmental and epidemiological data, ML models predict disease outbreaks like malaria or cholera, allowing health systems to prepare and respond effectively.

1.3 What is the ML Process?

ML is a data driven approach to discovering patterns in the data that can be exploited for making predictions. Data collection, cleansing and preparation for analysis are the initial steps in this process. In addition, the data is treated with various machine learning algorithms in order to determine patterns. In order to make predictions, the results of that analysis shall be used.

1.4 Problem-Statement:

Child mortality under age 5 remains a critical global health issue, with millions of preventable deaths annually due to diseases like pneumonia, malaria, and diarrhea etc. Despite abundant historical data, there is a lack of advanced tools to forecast mortality trends and prioritize high-risk diseases effectively.

Organizations like WHO struggle to derive actionable insights from complex datasets to guide interventions. This project aims to address these challenges by developing machine learning models for predicting and classifying mortality rates.

1.5 Aims:

- Develop predictive and classification models for forecasting mortality rates in children under age 5.
- Identify and prioritize high-risk diseases contributing to child mortality, such as pneumonia, malaria, and diarrhea.
- Provide actionable insights to organizations like the WHO for more informed decision-making.
- Optimize resource allocation by identifying high-risk regions and critical diseases for targeted interventions.
- Analyze trends and patterns in mortality rates over time, categorized by countries, regions, and organizations.

1.6 Objectives:

- Collect and preprocess historical data on mortality rates, diseases, and socio-economic factors.
- Implement regression analysis models (e.g., Linear Regression, KNN Regressor, Decision Tree Regressor) to predict mortality rates.
- Implement classification models (e.g., Decision Tree Classifier, KNN Classifier, Gradient Boosting Classifier) to classify regions or countries based on mortality risk levels.
- Evaluate the models using appropriate metrics (e.g., MSE for regression, Accuracy for classification).
- Conduct trend analysis to identify significant changes in mortality rates over time.
- Visualize results through heatmaps and other interactive tools to provide clear insights into high-risk areas.

1.7 Scope and Structure of the Project and Report:

This report covers the implementation of machine learning algorithms in child mortality analysis, focusing on datasets that combine numerical and categorical variables. It includes a detailed description of the ML process, from data collection to model deployment. The structure of the report is as follows:

- **Introduction:** Overview of child mortality and the role of machine learning in healthcare.
- **Machine Learning Process**: Explanation of data collection, preprocessing, feature engineering, model selection, and evaluation.
- **Case Study Application:** Implementation of machine learning techniques to child mortality detection and prediction.
- **Results and Evaluation:** Evaluation of model performance and insights drawn from the data.
- **Conclusion:** Despite the challenges, the project aims to provide a reliable tool for healthcare professionals to predict and personalize child mortality different factors like cause of death, most important factor influencing death rate, yearly or country wise death rate which can improve early detection and treatment.

2. Literature Review

2.1. Related Work:

[1]. Author: Susan Idicula-Thomas & Ulka Gawde Year: 2021. Machine learning (ML) algorithms have been successfully employed for prediction of outcomes in clinical research.

[2] .Author: Lily Tapak, MSc1, Hossein Mahjub, PhD2, Omid Hamidi, MSc3, Jalal Poorolajal, PhD2 Year: 2013. This study compared two traditional classification methods (logistic regression and Fisher linear discriminant analysis) and four machine-learning classifiers (neural networks, support vector machines, fuzzy c-mean, and random forests) to classify persons with and without diabetes.

[3]. Author: Harimoorthy, K (Harimoorthy, Karthikeyan) ; Thangavelu, M (Thangavelu, Menakadevi)Year: 2021. Data in the healthcare industry consists of patient information and disease- related information. This medical data and machine learning techniques will help us to analyse a large amount of data to find out the hidden patterns in the disease.

Research Paper	Health Analysis	Variables/ Features	Methodology	Accuracy
Comparison of machine learning algorithms applied to symptoms to determine infectious causes of death in children	SVM, ANN,CART, gradient boosting modeling, and KNN, c5	Cause of death, disease code, ICD-10 codes, Number of cases, %cases	Machine Learning	Based on disease SVM 80%-98% . ANN, KNN, CART, GBM 70%-90% C5 80%-90%
Real-Data Comparison of Data Mining Methods in Prediction of Diabetes in Iran	Fuzzy-C-mean, SVM, Random Forest, Logistic Regression, Linear Discriminant Analysis,	Sex,Smoking, Job, Hypertension, Residential, Physical Activity	Data Mining And Machine Learning	Lr = 0.93 Lda= 0.92 Fcm = 0.85 SVM = 0.98 NN = 0.93% RF = 0.93

	Neural Network.			
Multi-disease prediction model using improved SVM-radial bias	SVM-Radial bias kernel, SVM-Linear, SVM-Polynomial, Random forest and Decision tree	Datasets: Chronic kidney disease, Diabetes	Big Data Analytics	98.3%, 98.7%, and 89.9% in Chronic Kidney Disease, Diabetes

Table 2-1: Literature Review

2.2. Gaps in Existing Research Addressed by the Case Study

- **Limited Adoption of Advanced Machine Learning Techniques**

  The studies reviewed, including [1], [2], and [3], highlight the use of traditional statistical and predictive models in survival prediction. However, they do not extensively explore the integration of modern machine learning techniques such as Random Forest or XGBoost for improving predictive accuracy. The case study fills this gap by leveraging advanced machine learning models and comparing their performance to identify the most effective approach for survival status prediction.

- **Challenges in Handling Class Imbalance**

  A recurring challenge in the studies [1] and [3] is the imbalance in survival datasets, particularly with fewer instances of advanced-stage cases. The case study addresses this gap by applying preprocessing techniques that mitigate the impact of imbalanced classes, ensuring that the predictive models are robust and unbiased.

- **Absence of Real-World Applicability**

  Existing research, such as in [2] , focuses on theoretical or experimental validations without clear pathways for real-world implementation. This case study emphasizes actionable insights, demonstrating how the model's outputs can inform healthcare organization, such as early identification of high-risk disease which can cause high risk death and which specific area/country to be first prioritize for health aids.

- **Limited Use of Comprehensive Evaluation Metrics**

  While [1] and [2] primarily assess model performance using accuracy or basic statistical measures, they do not evaluate using comprehensive metrics like R^2 and Accuracy score. The case study addresses this gap by adopting a multifaceted evaluation strategy, enabling a more thorough understanding of model performance.

3. Dataset Description

The dataset appears to track child mortality rates for various diseases under the age of 5, including causes such as malaria, HIV/AIDS, meningitis, pneumonia, and more. Each record includes the following columns:

· **Entity**: The country or region or organizations (e.g., Afghanistan).

· **Code**: A unique identifier for the country or region (e.g., AFG for Afghanistan).

· **Year**: The year the data was recorded (e.g., 1990, 2019).

· These 17 columns representing the number of deaths due to various diseases

- **Malaria**
- **HIV/AIDS**.
- **Meningitis**.
- **Nutritional Deficiencies**.
- **Other Neonatal Disorders**
- **Other diseases**
- **Whooping Cough**
- **Lower Respiratory Infections**
- **Congenital birth defects**
- **Measles**
- **Neonatal sepsis and other neonatal infections**
- **Neonatal encephalopathy due to birth asphyxia and trauma**
- **Drowning**
- **Tuberculosis**
- **Neonatal preterm birth**
- **Diarrheal Diseases**.
- **Neoplasms**.
- **Syphilis**.

3.1 Key Features:

1. **Deaths per Disease**: The number of deaths for each disease under 5 years of age is recorded separately.
2. **Time-Series Data:** The data is available over multiple years, allowing for trend analysis and the identification of changes over time.
3. **Geographical Coverage**: The dataset includes information for different countries, which will enable regional comparisons

4. Methodology

Following is a step-by-step process for building a Predicting and Classifying Child Mortality Rates model.

4.1 Data Collection:

Data Sources: The dataset “cause-of-death-in-children is downloaded from kaggle.

Data Description: Entity (Countires or Organizations), Year (1990-2019), Code (Country code),

Disease Names 17 columns : Malaria, HIV/AIDS, Meningitis, Nutritional deficiencies, Other neonatal disorders, Whooping cough, Lower respiratory infections, Congenital birth defects, Measles,Neonatal sepsis and other neonatal infections, Neonatal encephalopathy due to birth asphyxia and trauma, Drowning, Tuberculosis, Neonatal preterm birth, Diarrheal diseases, Neoplasms, Syphilis

Data Availability: The "Code" column contains missing values because some regions or organizations may not have a country abbreviation

4.1.1. Exploratory Data Analysis (EDA):

Regular Expression: Applied regex to remove specific part from column.

Descriptive Statistics: Calculate mean, median, standard deviation, and correlation among the features.

4.1.2. Data Preprocessing:

Data Cleaning: Handling missing data by imputation (fillna= ‘Unknown’)

Data Transformation: Encoding categorical columns Entity and Code by applying LabelEncoder.

4.1.3. Data Visualization:

- Plotting trends of mortality rates by year, disease, or region.
- Visualizing correlations between diseases,years, country/area and mortality rates.
- Using plotly, bar charts, and scatter plots to understand relationships.

4.1.4. Machine Learning Models:

Model Selection:

Regression Models (for predicting mortality rates):

1. Linear Regression
2. K-Nearest Neighbors (KNN) Regressor.
3. Decision Tree Regressor.

Classification Models (for classifying regions or countries into high-risk categories based on mortality rates, Year, Diseases, Country/area):

1. Decision Tree Classifier.
2. K-Nearest Neighbors (KNN) Classifier.
3. Gradient Boosting Classifier.

4.2 Model Implementation:

- Scaling numerical data for machine learning models that require scaling (e.g., normalization or standardization.
- Training the models using the training dataset**.**

4.3 Model Evaluation:

- **Regression Model Evaluation**: Using performance metrics such as RMSE.
- **Classification Model Evaluation**: Using metrics such as Accuracy.
- Cross-validation to ensure model robustness.
- **Comparing Models:** Comparing the performance of different models to select the best one based on the evaluation metrics.

4.4 Tools and Libraries Used:

- Programming Languages: Python.
- Data Analysis Libraries: Pandas, NumPy.
- Data Visualization: Matplotlib, Seaborn, Plotly.
- Table: Tabulate
- Machine Learning Libraries: Scikit-learn

5. Result and Analysis

5.1 Four Case Studies:

5.1.1 Regression Analysis:

Case Study 1: Predicting Death Count: In this predictive analysis, the model takes the region, disease, and year as inputs and provides the predicted death count for specific disease in specific region/country as output.

Model Evaluation: By using RMSE evaluating model performance of Linear Regression, KNN Regressor and Decision Tree Regressor.

```
Enter the region (e.g., Africa, Asia, Europe): Afghanistan
Enter the disease (e.g., Malaria, HIV/AIDS, Tuberculosis): Malaria
Enter the year you want to predict: 2030
Model RMSE Scores:
KNN: 94.61, Linear Regression: 46.25, Decision Tree: 53.86
The best model is Linear Regression.
Predicted deaths for Malaria in Afghanistan in 2030: 111
```

Figure 5.1.1-1: Taking User Input, Printing RMSE OF Each Model And Predicting Death Count

5.1.2 Classification Analysis:

Case Study 2: Analysis By Entity And Year: In this classification analysis, the model takes country and year as input and provide top disease causing high death rate in specific region/country and in specific year.

Model Evaluation: By using Accuracy Score evaluating model performance of Random Forest Classifier, Gradient Boosting Classifier and Decision Tree Classifier.

```
Enter the country name: India
Enter the year: 2019
====================================
Random Forest Accuracy: 0.26
Gradient Boosting Accuracy: 0.86
Decision Tree Accuracy: 0.95
Best Model: Decision Tree with Accuracy: 0.95
======================================================================
The top diseases causing high death rates in India in 2019 are:
Neonatal preterm birth
```

Figure 5.1.2-1: Taking User Input, Printing Accuracy Score And Classifying Top Disease

Case Study 3: Analysis By Year: In this classification analysis, the model takes year as input and provide top cause of disease in specific year for every region/country.

Model Evaluation: By using Accuracy Score evaluating model performance of Random Forest Classifier, Gradient Boosting Classifier and Decision Tree Classifier.

```
Enter the year: 2018
Random Forest Accuracy: 0.70
Gradient Boosting Accuracy: 0.87
Decision Tree Accuracy: 0.94
Best Model: Decision Tree with Accuracy: 0.94
Top cause of death by disease in 2018:
```

Country	Disease	Death Count
Afghanistan	Congenital birth defects	16521
African Region (WHO)	Diarrheal diseases	368837
Albania	Congenital birth defects	102
Algeria	Neonatal preterm birth	5914
American Samoa	Neonatal preterm birth	2

Figure 5.1.2-2: Taking User Input, Printing User Input And Classifying Top Cause Of Death By Disease

Angola	Diarrheal diseases	7700
Antigua and Barbuda	Neonatal preterm birth	3
Argentina	Congenital birth defects	2101
Armenia	Congenital birth defects	135
Australia	Congenital birth defects	290
Austria	Congenital birth defects	91
Azerbaijan	Lower respiratory infections	1426
Bahamas	Congenital birth defects	7
Bahrain	Congenital birth defects	31
Bangladesh	Neonatal encephalopathy due to birth asphyxia and trauma	15700

Figure 5.1.2-3 Continue Output

Barbados	Congenital birth defects	7
Belarus	Congenital birth defects	179
Belgium	Congenital birth defects	113
Belize	Neonatal preterm birth	26
Benin	Malaria	8224
Bermuda	Malaria	0
Bhutan	Neonatal preterm birth	90
Bolivia	Neonatal preterm birth	1701
Bosnia and Herzegovina	Neonatal preterm birth	51
Botswana	Neonatal preterm birth	296

Figure 5.1.2-4: Continue Output

Western Pacific Region (WHO)	Congenital birth defects	49549
World	Lower respiratory infections	709340
World Bank High Income	Congenital birth defects	15683
World Bank Low Income	Lower respiratory infections	219630
World Bank Lower Middle Income	Lower respiratory infections	444305
World Bank Upper Middle Income	Congenital birth defects	95665
Yemen	Neonatal preterm birth	10526
Zambia	Diarrheal diseases	4102
Zimbabwe	Lower respiratory infections	4675

Figure 5.1.2-5: Continue Output

Case Study 4:Analysis By Disease And Year : In this classification analysis, the model will take year as input and provide top 5 countries with highest occurrence of specific disease and specific year.

Model Evaluation: By using Accuracy Score evaluating model performance of Random Forest Classifier, Gradient Boosting Classifier and Decision Tree Classifier.

```
Enter the year: 2019
Enter the disease name (must match exact column name): Malaria
================================================================================
Random Forest Accuracy: 0.71
Gradient Boosting Accuracy: 0.86
Decision Tree Accuracy: 0.94
Best Model: Decision Tree with Accuracy: 0.94

The top 5 countries with the highest occurrence of Malaria in 2019:
```

Country	Death Count
World	356363
Sub-Saharan Africa (WB)	345485
African Region (WHO)	342099
World Bank Low Income	200065
World Bank Lower Middle Income	155443

Figure 5.1.2-6: Taking User Input, Printing Accuracy And Classifying Top 5 Countries

6. Discussion

6.1. Significance of the Findings

The case study on child mortality rates due to various diseases provides significant insights into the global health challenges faced by under-5 children. The key findings of the study can be summarized as follows:

- **Identifying Key Diseases:** The study highlights the leading causes of child mortality, such as malaria, HIV/AIDS, pneumonia, and neonatal disorders, providing valuable information for public health efforts targeting high-mortality diseases.

- **Trends Over Time:** By analyzing mortality trends over multiple years, the study helps track changes in child mortality rates, showcasing the effectiveness of health interventions or identifying emerging challenges that need more focus.

- **Geographical Insights:** The dataset enables regional comparisons, showing how mortality rates vary across countries or regions. This information can guide policy makers and health organizations in targeting interventions more effectively in regions with higher mortality rates.

- **Predictive Modeling**: Through machine learning models, the study provides predictive insights, enabling the forecasting of mortality rates, which can be instrumental for resource allocation, health infrastructure planning, and early intervention.

- **Classifying High-Risk Areas**: The classification models help categorize regions or countries into high-risk categories based on mortality rates, helping prioritize regions that need urgent health interventions.

6.2. Limitations of the Study

While the study provides valuable insights, there are several limitations that must be considered:

- **Data Completeness**: The presence of missing values, especially in the "Code" column, can impact the reliability of the data, limiting the analysis to regions with available country codes. Some regions or organizations may not be fully represented, leading to biased conclusions.

- **External Factors** Not Included: The study focuses on mortality due to specific diseases but does not account for external factors like healthcare quality, socio-economic conditions, or vaccination rates, which could influence mortality rates.

- **Predictive Model Accuracy**: The accuracy of the predictive models, particularly for regression tasks such as predicting death counts, may not be perfect. Regression models like Linear Regression, Decision Tree Regressor, or KNN may be limited by their assumptions and the complexity of the data.

- **Limited to Available Data**: The study only covers data from 1990 to 2019, limiting its ability to capture very recent trends or the impacts of new health interventions.

6.3. Potential Improvements and Future Research Directions

- **Data Augmentation and Cleaning:** Further data collection and refinement could reduce missing values and improve data completeness. Imputation techniques like KNN imputation or using external datasets for filling missing data could be explored.

- **Incorporating External Variables**: Including socio-economic factors, healthcare infrastructure, vaccination rates, and other demographic details could lead to a more holistic analysis of child mortality rates.

- **Advanced Machine Learning Models**: While the current study uses basic models such as Decision Trees and KNN, exploring more advanced models like XGBoost, Random Forests, or deep learning techniques could improve the predictive power and accuracy of mortality predictions.

- **Cross-National Comparisons**: Future studies could compare mortality rates in countries with similar economic conditions, healthcare systems, and educational levels to identify effective interventions that could be adopted in other regions.

- **Longitudinal Analysis:** The study could extend beyond 2019 to understand the evolving nature of child mortality rates, particularly in light of ongoing global health crises like the COVID-19 pandemic or changes in healthcare practices.

- **Classifying High-Risk Areas:** Further refinement of the classification models using more granular data could improve the identification of high-risk regions, allowing for better-targeted interventions.

- **Real-Time Data Integration:** Incorporating real-time or near real-time data from health organizations could allow for more timely interventions and improvements in child mortality reduction strategies.

By addressing these limitations and exploring the outlined future directions, the study could provide more comprehensive and actionable insights for reducing child mortality rates globally.

7. Conclusion:

In conclusion, our project represents a significant step forward in leveraging technology to address the critical global issue of under-5 child mortality. By developing machine learning models for classification and prediction, we aim to provide actionable insights into the leading causes of death among children under the age of five. These insights will help identify high-risk regions, prioritize diseases requiring urgent attention, and guide effective resource allocation. We remain committed to refining our methodologies, improving data quality, and enhancing analytical techniques to ensure a meaningful impact in reducing child mortality rates and strengthening global health initiatives.

7.1. The broader impact of the study in medical applications and research

The broader impact of this study in medical applications and research is significant across various areas:

- **Improved Health Policy:** The study aids in identifying high-risk regions and diseases, helping policymakers allocate resources effectively and design targeted interventions.
- **Enhanced Public Health Decision-Making:** It supports data-driven decisions on disease prevention and management, optimizing public health strategies.
- **Advancement in Epidemiology**: The study provides insights into disease interactions, helping researchers understand factors contributing to child mortality and guiding policy development.
- **Global Health Interventions:** By identifying high mortality causes, it informs global strategies and supports efforts toward achieving SDG 3 (reducing child mortality).
- **Disease Prediction and Prevention:** The predictive models could lead to early warning systems for outbreaks and improved preventive healthcare strategies.
- **Addressing Health Inequities:** The study highlights disparities in child mortality, enabling further research into reducing healthcare access gaps and social determinants of health.
- **Pediatric Research:** It provides data for focused medical research on diseases like malaria, pneumonia, and neonatal infections, leading to better treatments and interventions.
- **Machine Learning in Healthcare:** The study demonstrates the potential of machine learning in predicting disease trends, which can be applied across various healthcare domains.

8. References:

[1]. https://rdcu.be/d6DFS

[2]. https://doi.org/10.4258/hir.2013.19.3.177

[3]. https://www.webofscience.com/wos/woscc/full-record/WOS:000505384300008

9. Appendix

9.1. Codes

LOADING DATASET

```
file_path = r'C:\Users\admin\Desktop\causes-of-death-in-children.csv'
df = pd.read_csv(file_path)
df.head()
```

	Entity	Code	Year	Deaths - Malaria - Sex: Both - Age: Under 5 (Number)	Deaths - HIV/AIDS - Sex: Both - Age: Under 5 (Number)	Deaths - Meningitis - Sex: Both - Age: Under 5 (Number)	Deaths - Nutritional deficiencies - Sex: Both - Age: Under 5 (Number)	Deaths - Other neonatal disorders - Sex: Both - Age: Under 5 (Number)	Deaths - Whooping cough - Sex: Both - Age: Under 5 (Number)	Deaths - Lower respiratory infections - Sex: Both - Age: Under 5 (Number)	Deaths - Congenital birth defects - Sex: Both - Age: Under 5 (Number)	Deaths - Measles - Sex: Both - Age: Under 5 (Number)	Deaths - Neonatal sepsis and other neonatal infections - Sex: Both - Age: Under 5 (Number)	Deaths - Neonatal encephalopathy due to birth asphyxia and trauma - Sex: Both - Age: Under 5 (Number)	I
0	Afghanistan	AFG	1990	21	10	1709	1779	7112	2455	20224	12850	8649	420	1599	
1	Afghanistan	AFG	1991	41	12	1743	1822	7574	2385	20879	13701	8669	520	1804	
2	Afghanistan	AFG	1992	51	13	1954	2069	8614	2370	23585	15812	8539	662	2160	
3	Afghanistan	AFG	1993	24	16	2252	2427	9458	2659	27116	17855	8949	723	2414	
4	Afghanistan	AFG	1994	52	19	2446	2649	9823	3187	29271	18835	10642	736	2519	

Figure 9-1: Loading Dataset

EDA

```
# Remove specific parts from column names
df.columns = df.columns.str.replace(r' - Sex: Both - Age: Under 5 \(Number\)', '', regex=True)
```

```
df.columns = df.columns.str.replace(r'Deaths - ', '', regex=True)
```

```
df.head()
```

	Entity	Code	Year	Malaria	HIV/AIDS	Meningitis	Nutritional deficiencies	Other neonatal disorders	Whooping cough	Lower respiratory infections	Congenital birth defects	Measles	Neonatal sepsis and other neonatal infections	Neonatal encephalopathy due to birth asphyxia and trauma	Drov
0	Afghanistan	AFG	1990	21	10	1709	1779	7112	2455	20224	12850	8649	420	1599	
1	Afghanistan	AFG	1991	41	12	1743	1822	7574	2385	20879	13701	8669	520	1804	
2	Afghanistan	AFG	1992	51	13	1954	2069	8614	2370	23585	15812	8539	662	2160	
3	Afghanistan	AFG	1993	24	16	2252	2427	9458	2659	27116	17855	8949	723	2414	
4	Afghanistan	AFG	1994	52	19	2446	2649	9823	3187	29271	18835	10642	736	2519	

Figure 9-2: Pre-processing

DATA MANIPULATION

```
print("\nHandling Missing Values...")
missing_values = df.isnull().sum()
print(missing_values[missing_values > 0])
```

```
Handling Missing Values...
Code      690
dtype: int64
```

```
df['Code'] = df['Code'].fillna('Unknown')
```

```
print(df.isnull().sum().sum())
```

```
0
```

Figure 9-3: Handling Missing Values

LABEL ENCODER

```
label_encoder_code = LabelEncoder()
label_encoder_entity = LabelEncoder()
df['Code_encoded'] = label_encoder_code.fit_transform(df['Code'])
df['Entity_encoded'] = label_encoder_entity.fit_transform(df['Entity'])
df.head()
```

	Entity	Code	Year	Malaria	HIV/AIDS	Meningitis	Nutritional deficiencies	Other neonatal disorders	Whooping cough	Lower respiratory infections	...	Neonatal sepsis and other neonatal infections	Neonatal encephalopathy due to birth asphyxia and trauma	Drowning	Tuberculosi
0	Afghanistan	AFG	1990	21	10	1709	1779	7112	2455	20224	...	420	1599	776	80
1	Afghanistan	AFG	1991	41	12	1743	1822	7574	2385	20879	...	520	1804	748	80
2	Afghanistan	AFG	1992	51	13	1954	2069	8614	2370	23585	...	662	2160	777	86
3	Afghanistan	AFG	1993	24	16	2252	2427	9458	2659	27116	...	723	2414	872	97
4	Afghanistan	AFG	1994	52	19	2446	2649	9823	3187	29271	...	736	2519	961	106

5 rows × 22 columns

Figure 9-4: Handling Categorical Data By Applying Label Encoder

Case Study 1: Predicting Death Count:

APPLYING REGRESSION ML MODELS

```
def preprocess_data(df, region, disease):
    df_region = df[df['Entity'] == region]
    df_region = df_region[['Year', disease]].dropna()  # Remove rows with missing values

    X = df_region[['Year']]  # Independent variable: Year
    y = df_region[disease]   # Dependent variable: Disease

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # Scaling (Standardization)
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    return X_train_scaled, X_test_scaled, y_train, y_test, scaler
```

Figure 9-5: Splitting and Scaling Features For Prediction Task

```python
def evaluate_models(X_train_scaled, X_test_scaled, y_train, y_test):
    # Train KNN model
    knn_reg_model = KNeighborsRegressor(n_neighbors=5)
    knn_reg_model.fit(X_train_scaled, y_train)
    y_pred_knn = knn_reg_model.predict(X_test_scaled)
    rmse_knn = np.sqrt(mean_squared_error(y_test, y_pred_knn))

    # Train Linear Regression model
    lr_model = LinearRegression()
    lr_model.fit(X_train_scaled, y_train)
    y_pred_lr = lr_model.predict(X_test_scaled)
    rmse_lr = np.sqrt(mean_squared_error(y_test, y_pred_lr))

    # Train Decision Tree model
    dt_reg_model = DecisionTreeRegressor(random_state=42)
    dt_reg_model.fit(X_train_scaled, y_train)
    y_pred_dt = dt_reg_model.predict(X_test_scaled)
    rmse_dt = np.sqrt(mean_squared_error(y_test, y_pred_dt))

    # Compare RMSE and select the best model
    print(f"Model RMSE Scores:\nKNN: {rmse_knn:.2f}, Linear Regression: {rmse_lr:.2f}, Decision Tree: {rmse_dt:.2f}")
    if rmse_knn <= rmse_lr and rmse_knn <= rmse_dt:
        return knn_reg_model, "KNN Regressor"
    elif rmse_lr <= rmse_knn and rmse_lr <= rmse_dt:
        return lr_model, "Linear Regression"
    else:
        return dt_reg_model, "Decision Tree Regressor"
```

Figure 9-6: Evaluation Function- Training Models, Calculating RMSE Comparing RMSE And Selecting Best Model

```python
def predict_deaths(region, disease, year_to_predict, best_model, scaler):
    # Scale the input year
    year_scaled = scaler.transform([[year_to_predict]])  # Scale the input year
    predicted_deaths = best_model.predict(year_scaled)
    print(f"Predicted deaths for {disease} in {region} in {year_to_predict}: {predicted_deaths[0]:.0f}")
```

```python
try:
    # Step 1: Get user input
    region = input("Enter the region (e.g., Africa, Asia, Europe): ")
    disease = input("Enter the disease (e.g., Malaria, HIV/AIDS, Tuberculosis): ")
    year_to_predict = int(input("Enter the year you want to predict: "))

    # Step 2: Preprocess the data
    X_train_scaled, X_test_scaled, y_train, y_test, scaler = preprocess_data(df, region, disease)

    # Step 3: Train models and evaluate RMSE
    best_model, best_model_name = evaluate_models(X_train_scaled, X_test_scaled, y_train, y_test)

    print(f"The best model is {best_model_name}.")

    # Step 4: Predict deaths using the best model
    predict_deaths(region, disease, year_to_predict, best_model, scaler)

except ValueError as e:
    print(f"Error: {e}")
```

Figure 9-7: Predicting Death Count By Taking User Input

Case Study 2: Classification Analysis By Country And Year:

```python
def preprocess_top_diseases_data(df, country, year):
    # Filter data for the selected country and year
    df_filtered = df[(df['Entity'] == country) & (df['Year'] == year)]

    if df_filtered.empty:
        raise ValueError(f"No data available for the selected country ({country}) and year ({year}).")

    disease_columns = [col for col in df.columns if col not in ['Entity', 'Code', 'Year']]


    if df_filtered[disease_columns].isnull().all().all():
        raise ValueError(f"No death count data for any diseases in {country} in {year}.")

    disease_counts = df_filtered[disease_columns].iloc[0]

    top_diseases = disease_counts.sort_values(ascending=False).head(5).index.tolist()
    df['High_Death_Disease'] = df[disease_columns].idxmax(axis=1)  # Add top disease as label

    # Encode the target variable and categorical features
    label_encoder_disease = LabelEncoder()
    df['High_Death_Disease'] = label_encoder_disease.fit_transform(df['High_Death_Disease'])

    label_encoder_country = LabelEncoder()
    df['Entity_encoded'] = label_encoder_country.fit_transform(df['Entity'])

    # Select features and target
    X = df[['Year', 'Entity_encoded']]
    y = df['High_Death_Disease']

    # Split into train and test sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # Scale features
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    return X_train_scaled, X_test_scaled, y_train, y_test, top_diseases, scaler, label_encoder_disease, label_encoder_country
```

Figure 9-8: Preprocess Function- Filtering Data According To Country And Year

```python
def evaluate_classifier_model(X_train_scaled, X_test_scaled, y_train, y_test):
    models = {
        "Random Forest": RandomForestClassifier(random_state=42),
        "Gradient Boosting": GradientBoostingClassifier(random_state=42),
        "Decision Tree": DecisionTreeClassifier(random_state=42)
    }

    accuracies = {}
    for model_name, model in models.items():
        model.fit(X_train_scaled, y_train)
        y_pred = model.predict(X_test_scaled)
        accuracy = accuracy_score(y_test, y_pred)
        accuracies[model_name] = accuracy
        print(f"{model_name} Accuracy: {accuracy:.2f}")

    # Select the best model
    best_model_name = max(accuracies, key=accuracies.get)
    best_model = models[best_model_name]
    print(f"Best Model: {best_model_name} with Accuracy: {accuracies[best_model_name]:.2f}")

    return best_model, best_model_name
```

Figure 9-9: Evaluate Classifier Model Function By Accuracy Score

```python
def predict_top_diseases(country, year, classifier, scaler, label_encoder_disease, label_encoder_country, df):
    # Prepare the data for the specific country and year
    df_filtered = df[(df['Entity'] == country) & (df['Year'] == year)]
    if df_filtered.empty:
        print(f"No data available for {country} in {year}.")
        return

    # Extract the features
    X_predict = df_filtered[['Year', 'Entity_encoded']]

    # Ensure 'Entity_encoded' is available
    if 'Entity_encoded' not in X_predict.columns:
        X_predict['Entity_encoded'] = label_encoder_country.transform(X_predict['Entity'])  # Transform if missin

    # Scale the features
    X_predict_scaled = scaler.transform(X_predict[['Year', 'Entity_encoded']])  # Only use relevant columns for s

    # Make predictions
    predicted_diseases = classifier.predict(X_predict_scaled)
    predicted_disease_names = label_encoder_disease.inverse_transform(predicted_diseases)

    print("=====================================================================")
    print(f"The top diseases causing high death rates in {country} in {year} are:")
    for disease in predicted_disease_names:
        print(disease)
```

Figure 9-10: Predicting Disease Name By Country And Year

```python
try:
    # User Input
    country = input("Enter the country name: ")
    year = int(input("Enter the year: "))
    print("==================================")

    # Preprocess data
    X_train_scaled, X_test_scaled, y_train, y_test, top_diseases, scaler, label_encoder_disease, label_encoder_country = preproce

    # Train models and select the best one
    best_model, best_model_name = evaluate_classifier_model(X_train_scaled, X_test_scaled, y_train, y_test)

    # Predict top diseases
    predict_top_diseases(country, year, best_model, scaler, label_encoder_disease, label_encoder_country, df)

except ValueError as e:
    print(f"Error: {e}")
```

Figure 9-11: Taking User Input For Prediction

Case Study 3: Classification Analysis By Year:

Analysis By Year ¶

```
]: def preprocess_top_diseases_data_by_year(df, year):
    # Filter data for the selected year
    df_filtered = df[df['Year'] == year]

    if df_filtered.empty:
        raise ValueError(f"No data available for the year {year}.")

    non_disease_columns = ['Entity', 'Code', 'Year', 'Entity_encoded', 'Code_encoded']
    disease_columns = [col for col in df.columns if col not in non_disease_columns]

    # Check if the dataset contains any death counts for diseases
    if df_filtered[disease_columns].isnull().all().all():
        raise ValueError(f"No death count data for any diseases in {year}.")

    return df_filtered, disease_columns
```

Figure 9-12: Filtering Data By Year

```
def predict_top_diseases(year, df):
    # Preprocess the data for the selected year
    df_filtered, disease_columns = preprocess_top_diseases_data_by_year(df, year)

    # Prepare features and target for training
    X = df[['Year', 'Entity_encoded', 'Code_encoded']]
    y = df[disease_columns].idxmax(axis=1)

    # Encode categorical features
    label_encoder_disease = LabelEncoder()
    df['High_Death_Disease'] = label_encoder_disease.fit_transform(y)

    label_encoder_country = LabelEncoder()
    df['Entity_encoded'] = label_encoder_country.fit_transform(df['Entity'])

    label_encoder_code = LabelEncoder()
    df['Code_encoded'] = label_encoder_code.fit_transform(df['Code'])

    # Split data into train and test sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # Scale features
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    # Train models and evaluate
    best_model, best_model_name = evaluate_classifier_model(X_train_scaled, X_test_scaled, y_train, y_test)

    # Predict the top diseases for the selected year
    results = []

    for _, row in df_filtered.iterrows():

        disease_counts = pd.to_numeric(row[disease_columns], errors='coerce')

        # Rank diseases based on death counts and select the top one
        high_cause_of_death = disease_counts.idxmax()
        death_count = disease_counts.max()


        if high_cause_of_death != 'High_Death_Disease':
            results.append([row['Entity'], high_cause_of_death, death_count])

    # Output the results as a table, ensuring that High_Death_Disease doesn't appear
    print(f"Top cause of death by disease in {year}:")
    print(tabulate(results, headers=["Country", "Disease", "Death Count"], tablefmt="grid"))
```

Figure 9-13: Predicting Top Cause Of Death By Disease In Year

```python
# Example Usage
try:
    # User Input
    year = int(input("Enter the year: "))

    # Predict top diseases for the selected year
    predict_top_diseases(year, df)

except ValueError as e:
    print(f"Error: {e}")
```

Figure 9-14: TAking Year As User Input

Case Study 4: Classification Analysis By Disease:

Analysis By Disease

```python
def preprocess_top_diseases_data1(df, year):
    # Filter data for the selected year
    df_filtered = df[df['Year'] == year]
    if df_filtered.empty:
        raise ValueError(f"No data available for the year {year}.")

    non_disease_columns = ['Entity', 'Code', 'Year', 'Entity_encoded', 'Code_encoded']
    disease_columns = [col for col in df.columns if col not in non_disease_columns]

    if df_filtered[disease_columns].isnull().all().all():
        raise ValueError(f"No death count data for any diseases in {year}.")

    return df_filtered, disease_columns
```

```python
def get_top_disease_countries(year, df, disease_name):
    # Preprocess the data for the selected year
    df_filtered, disease_columns = preprocess_top_diseases_data1(df, year)

    # Check if the selected disease exists in the columns
    if disease_name not in disease_columns:
        raise ValueError(f"{disease_name} data is missing from the dataset.")

    # Sort countries by the occurrence of the selected disease
    df_sorted = df_filtered[['Entity', disease_name]].sort_values(by=disease_name, ascending=False)

    # Get the top 5 countries
    top_5_countries = df_sorted.head(5)

    # Output the results as a table using tabulate
    print(f"\nThe top 5 countries with the highest occurrence of {disease_name} in {year}:")
    print(tabulate(top_5_countries[['Entity', disease_name]].values,
                   headers=["Country", "Death Count"], tablefmt="grid"))
```

Figure 9-15: Filtering Data By Year And Disease And Providing Top 5 Countries

```
try:
    # User Input
    year = int(input("Enter the year: "))
    disease_name = input("Enter the disease name (must match exact column name): ").strip()
    print("================================================================================")

    # Prepare features and target for training
    df_filtered, disease_columns = preprocess_top_diseases_data1(df, year)
    X = df[['Year', 'Entity_encoded', 'Code_encoded']]  # Features: Year, Entity_encoded, Code_encoded
    y = df[disease_columns].idxmax(axis=1)  # Target: Disease with highest death count

    # Encode categorical features
    label_encoder_disease = LabelEncoder()
    df['High_Death_Disease'] = label_encoder_disease.fit_transform(y)

    label_encoder_country = LabelEncoder()
    df['Entity_encoded'] = label_encoder_country.fit_transform(df['Entity'])

    label_encoder_code = LabelEncoder()
    df['Code_encoded'] = label_encoder_code.fit_transform(df['Code'])

    # Split data into train and test sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # Scale features
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    # Evaluate classifier models and determine the best one
    best_model, best_model_name = evaluate_classifier_model(X_train_scaled, X_test_scaled, y_train, y_test)

    # Get the top 5 countries for the disease
    get_top_disease_countries(year, df, disease_name)

except ValueError as e:
    print(f"Error: {e}")
```

Figure 9-16: Printing The Table

Models	Case Study 1 (Regression Task) (RMSE)	Case Study 2 (Classification Task) (Accuracy Score)	Case Study 3 (Classification Task) (Accuracy Score)	Case Study 4 (Classification Task) (Accuracy Score)
Random Forest Classifier	-	0.27	0,72	0.68
Gradient Boosting Classifier	-	0.84	0.86	0.87
Decision Tree Classifier	-	0.95	0.94	0.95
KNN Regressor	94.61	-	-	-
Linear Regresson	46.25	-	-	-
Decision Tree Regression	53.86	-	-	-

Table 9-1: Models Accuracy Or RMSE For Case Studies

9.2. Graphs

HeatMap

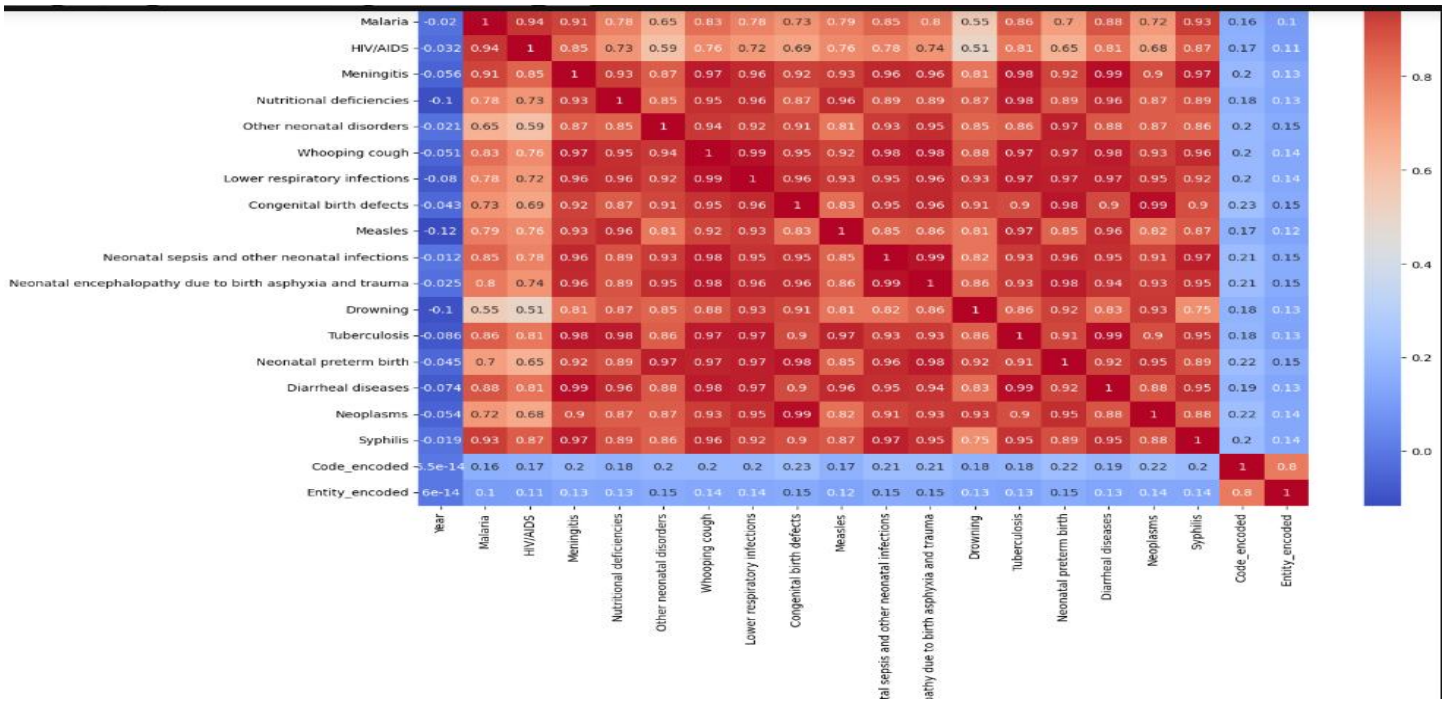

Figure 9.2-1: Co-relation Matrix

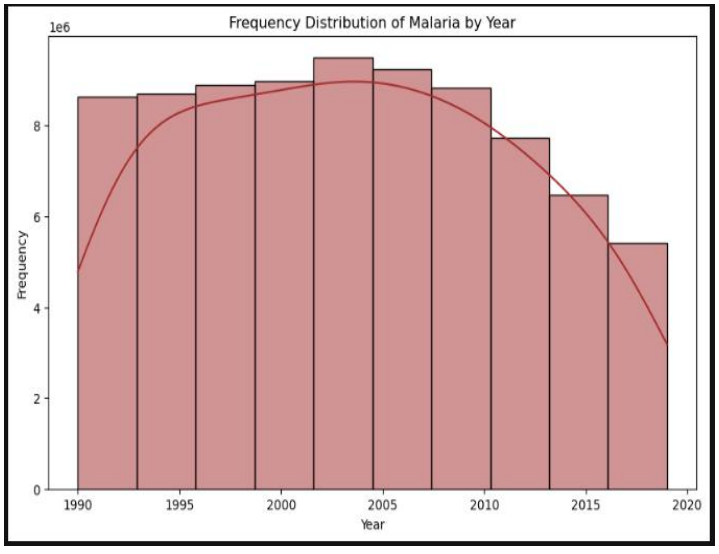

Figure 9.2-2: Malaria By Year

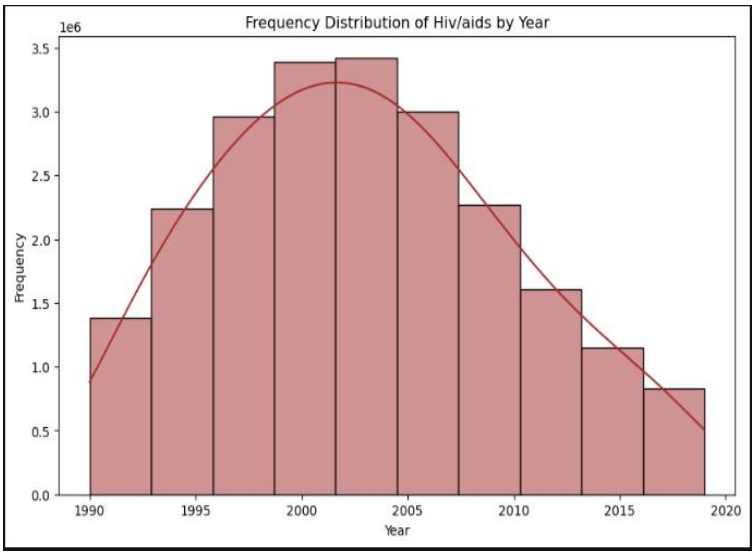

Figure 9.2-3: Hiv/aids By Year

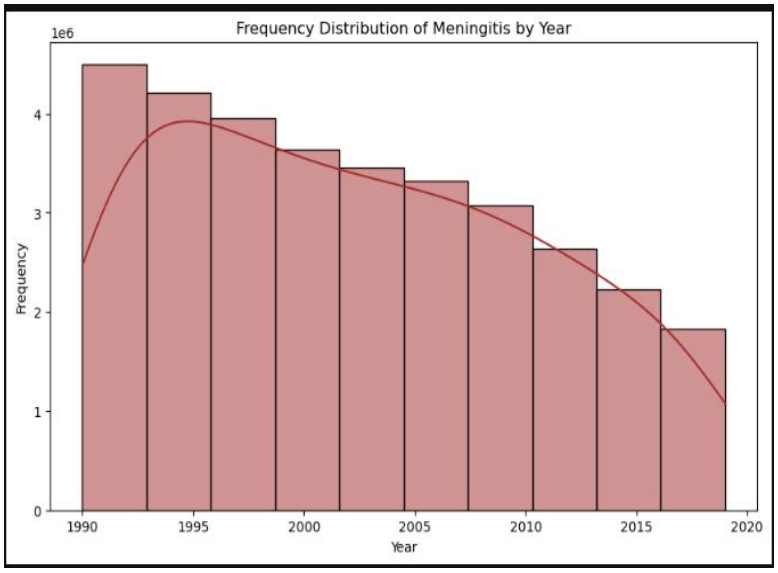

Figure 9.2-4: Meningitis By Year

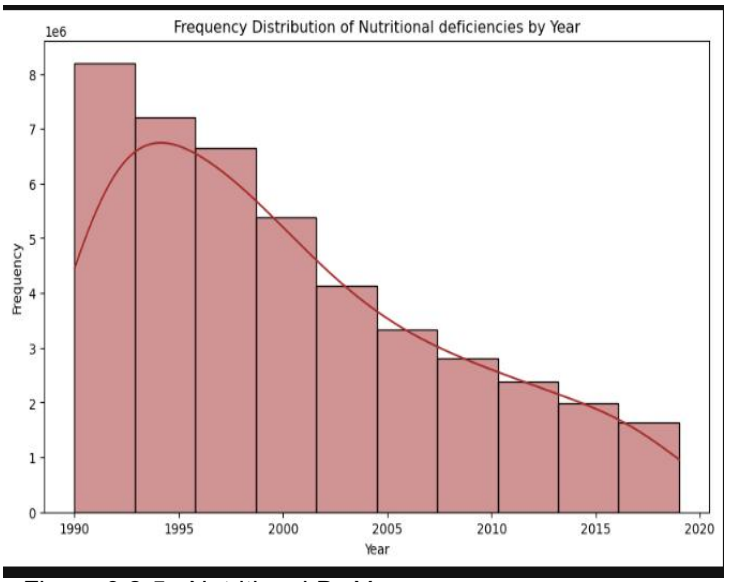

Figure 9.2-5: Nutritional By Year

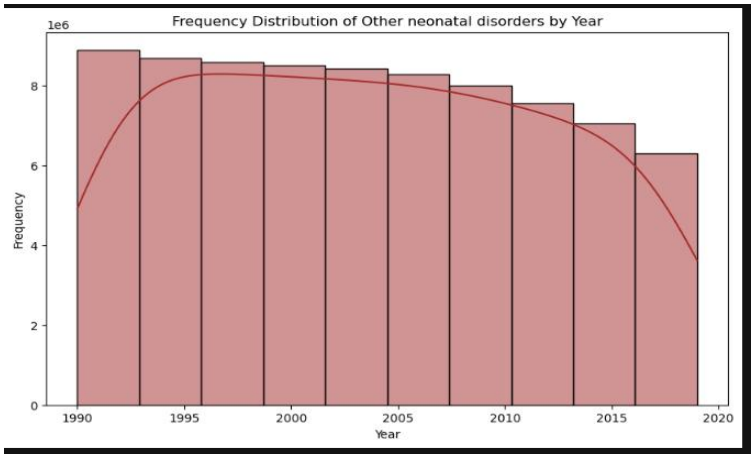

Figure 9.2-6: Other Neontal Disorders ByYear

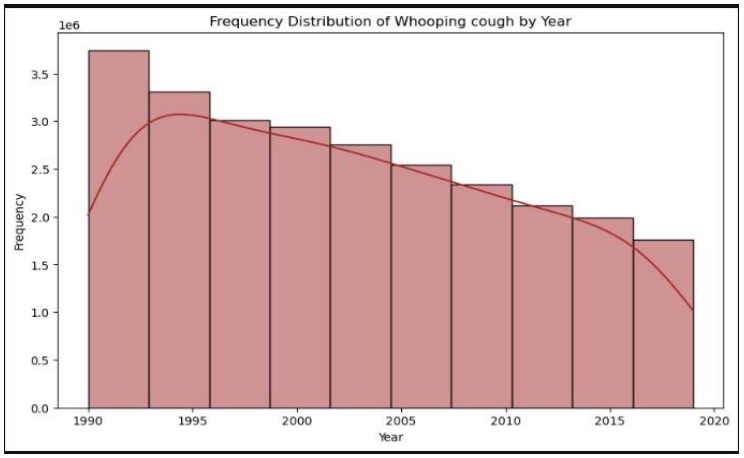

Figure 9.2-7: Whooping Cough By Year

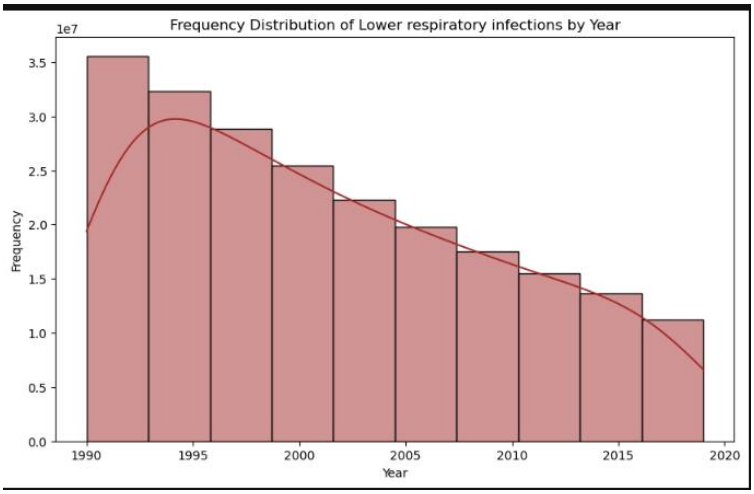

Figure 9.2-8: LRI By Year

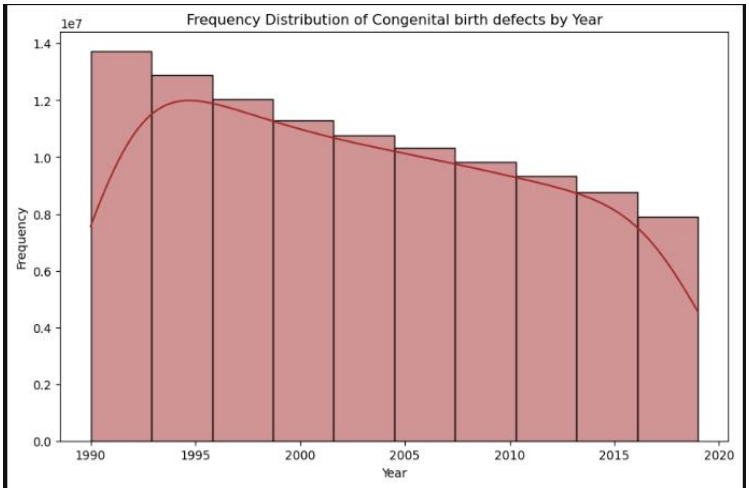

Figure 9.2-9: Congenital Birth Defects By Year

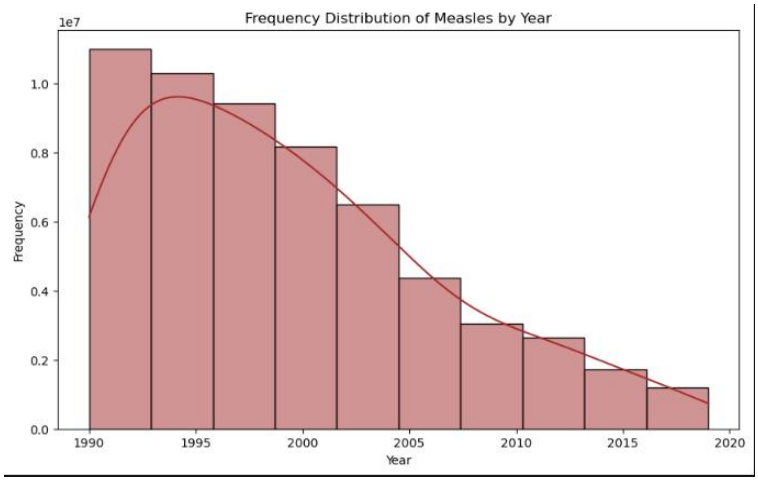

Figure 9.2-10: Measies By Year

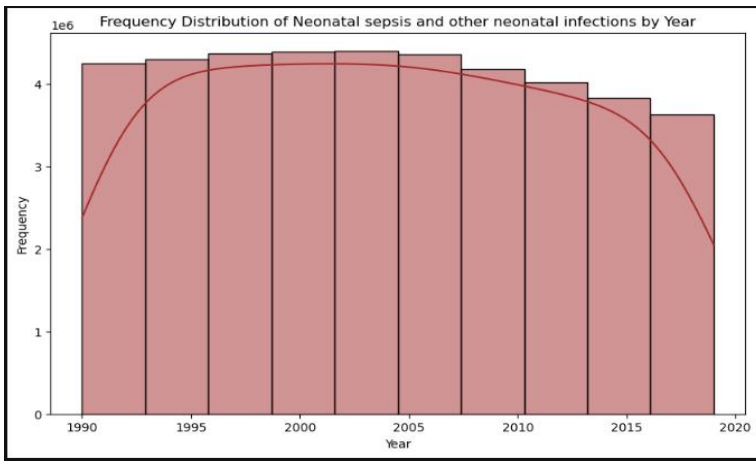

Figure 9.2-11: Neonatal Sepsis And Infection By Year

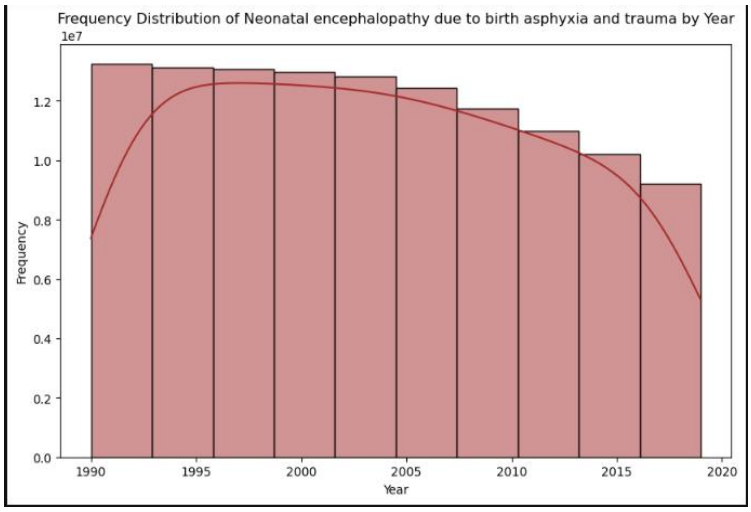

Figure 9.2-12: Neonatal Due To Birth And Trauma By Year

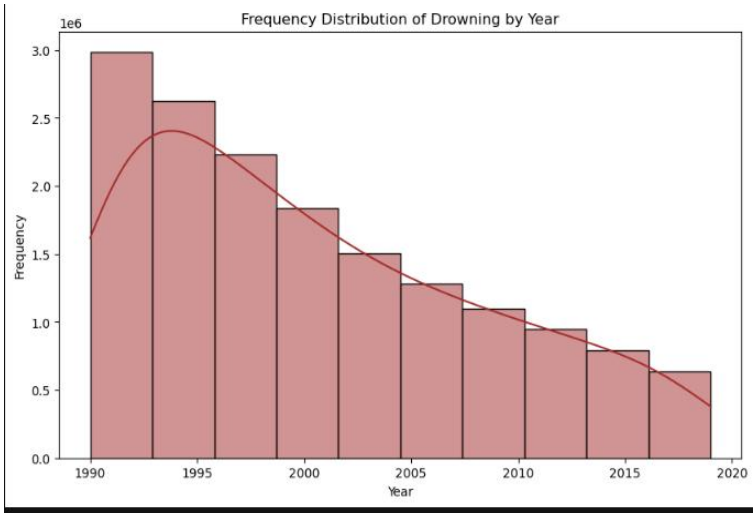

Figure 9.2-13: Drowning By Year

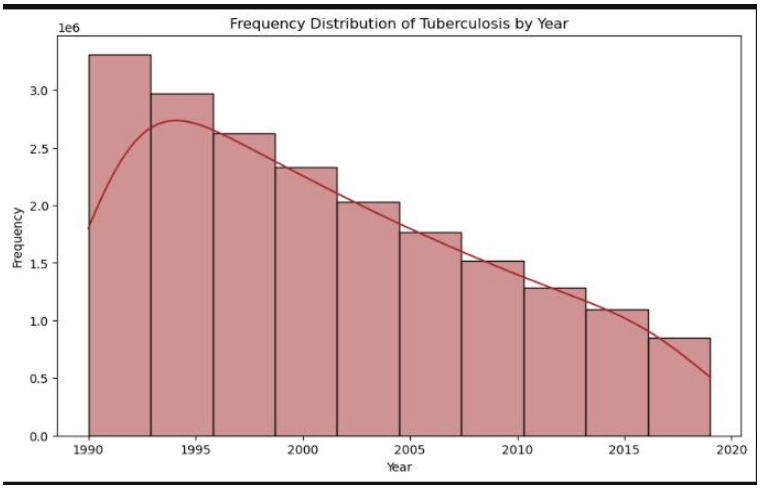

Figure 9.2-14: Tuberculosis By Year

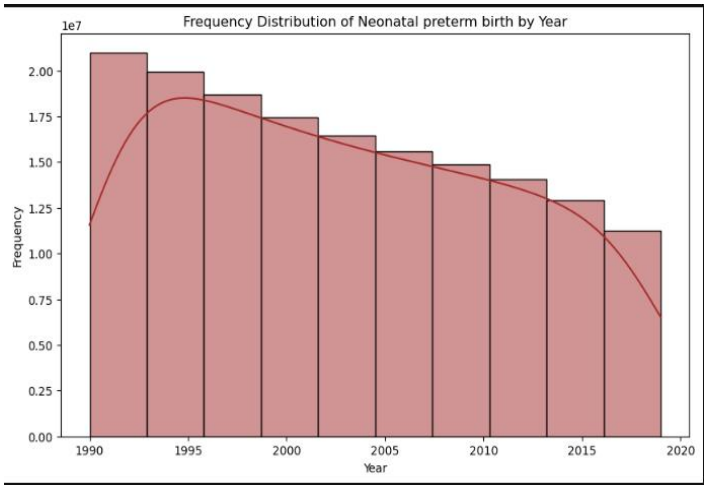

Figure 9.2-15: Neonatal Pertem Birth By Year

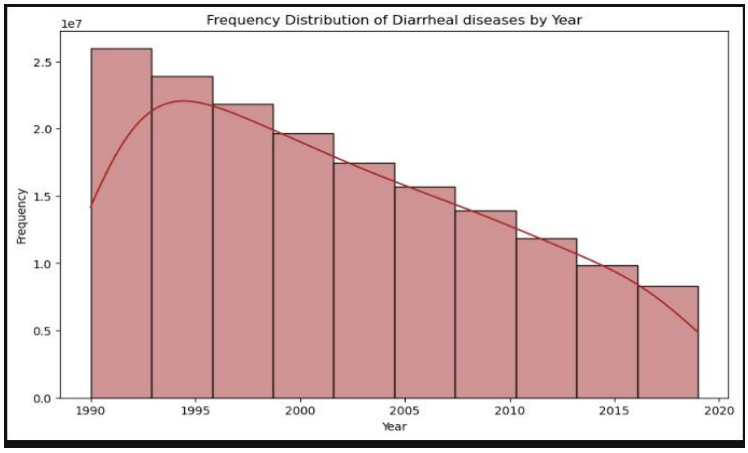

Figure 9.2-16: Diarrheal By Year

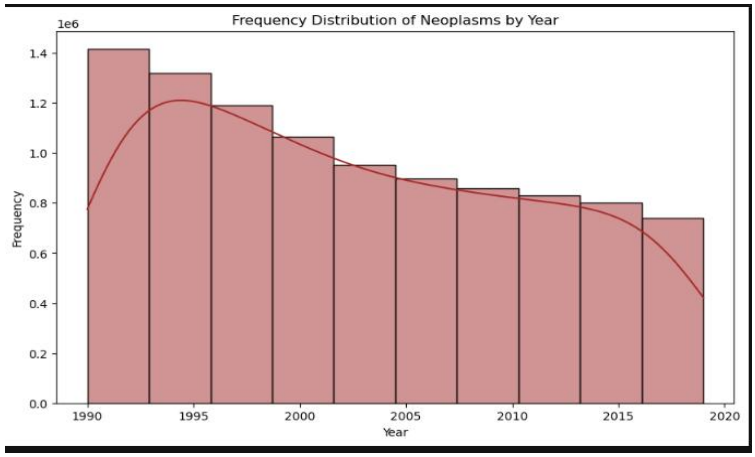

Figure 9.2-17: Neoplasms By Year

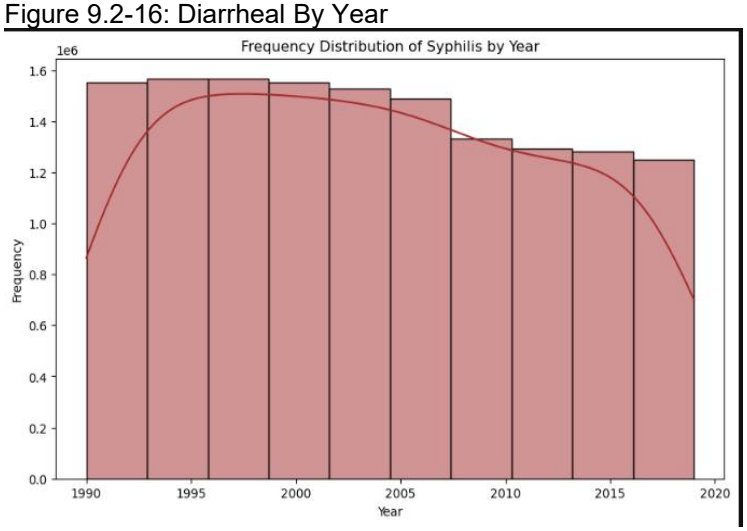

Figure 9.2-18: Syphilis By Year

```
disease_name = input("Enter the disease name (e.g., Malaria, HIV/AIDS, Tuberculosis): ")

if disease_name in df.columns:

    disease_counts = df[['Entity', disease_name]].groupby('Entity')[disease_name].sum().reset_index()
    top_10_disease = disease_counts.sort_values(by=disease_name, ascending=False).head(10)
    plt.figure(figsize=(12, 8))
    plt.bar(top_10_disease['Entity'], top_10_disease[disease_name], color='green')
    plt.title(f'Top 10 Entities with Highest Occurrence of {disease_name}')
    plt.xlabel('Entity (Country/Area/Organization)')
    plt.ylabel(f'{disease_name} Death Count')
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()
else:
    print(f"Error: The disease '{disease_name}' does not exist in the dataset. Please check your input.")
```

```
Enter the disease name (e.g., Malaria, HIV/AIDS, Tuberculosis): Malaria
```

Figure 9.2-19: Graphical Representation-Taking User Input To Visualize Top 10 Entities

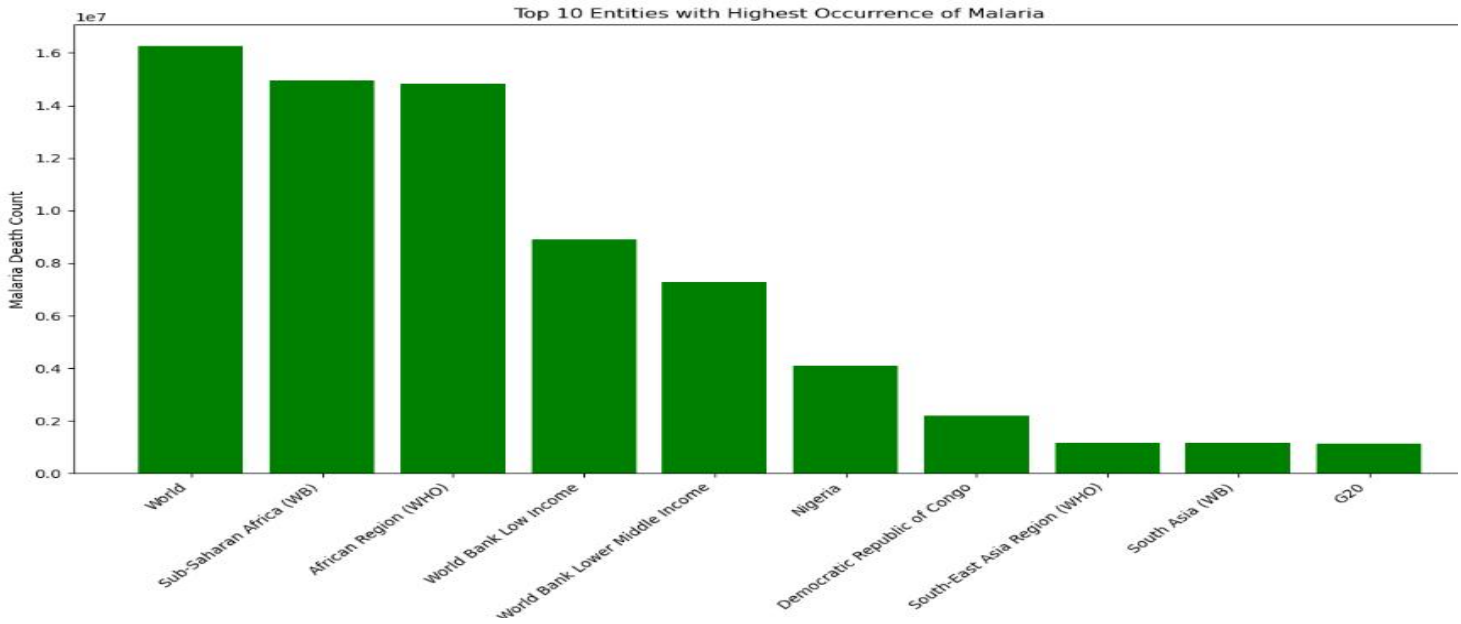

Figure 9.2-20: Output Of Figure 9.2-19

```
entity_name = input("Enter the country or region (e.g., India, Pakistan, Afghanistan): ")

entity_data = df[df['Entity'] == entity_name]

if not entity_data.empty:
    disease_columns = [col for col in df.columns if col not in ['Entity', 'Code', 'Year','Code_encoded','Entity_encoded']]
    disease_death_counts_entity = entity_data[disease_columns].sum()
    top_5_diseases_entity = disease_death_counts_entity.nlargest(5)
    plt.figure(figsize=(8, 8))
    plt.pie(top_5_diseases_entity, labels=top_5_diseases_entity.index, autopct='%1.1f%%', startangle=140)
    plt.title(f'Proportion of Top 5 Diseases with Highest Death Count in {entity_name}')
    plt.axis('equal')
    plt.show()
else:
    print(f"Error: No data found for the entity '{entity_name}'. Please check your input.")
```

```
Enter the country or region (e.g., India, Pakistan, Afghanistan): Pakistan
```

Figure 9.2-21: Graphical Representation-Taking User Input To Visualize Top 5 Diseases

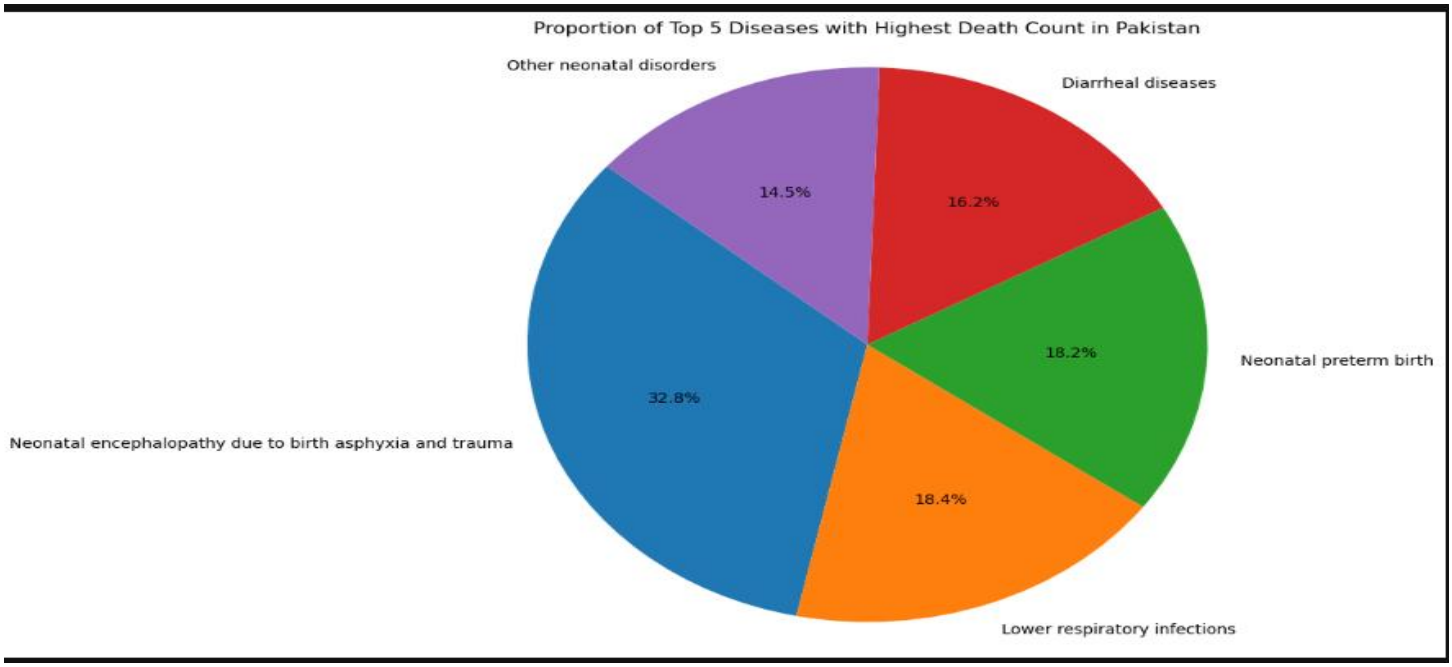

Figure 9.2-22: Pie Chart To Visualize The Output Of 9.2-21

```
entity_name = input("Enter the country or region (e.g., Algeria, India, Pakistan): ")

entity_data = df[df['Entity'] == entity_name]

if not entity_data.empty:

    disease_columns = [col for col in df.columns if col not in ['Entity', 'Code', 'Year','Code_encoded','Entity_encoded']]

    disease_death_counts_entity = entity_data[disease_columns].sum()
    top_10_max_diseases = disease_death_counts_entity.nlargest(10)
    top_10_min_diseases = disease_death_counts_entity.nsmallest(10)
    plt.figure(figsize=(14, 8))

    plt.subplot(1, 2, 1)
    sns.lineplot(x=top_10_max_diseases.index, y=top_10_max_diseases.values, marker='o', color='green')
    plt.title(f'Top 10 Most Occurring Diseases in {entity_name}')
    plt.xlabel('Disease')
    plt.ylabel('Death Count')
    plt.xticks(rotation=90)
    plt.grid(True)

    # Plot for the top 10 least occurring diseases
    plt.subplot(1, 2, 2)
    sns.lineplot(x=top_10_min_diseases.index, y=top_10_min_diseases.values, marker='o', color='red')
    plt.title(f'Top 10 Least Occurring Diseases in {entity_name}')
    plt.xlabel('Disease')
    plt.ylabel('Death Count')
    plt.xticks(rotation=90)
    plt.grid(True)

    # Adjust the layout and show the plot
    plt.tight_layout()
    plt.show()
else:
    print(f"Error: No data found for the entity '{entity_name}'. Please check your input.")
```

```
Enter the country or region (e.g., Algeria, India, Pakistan): Afghanistan
```

Figure 9.2-23: Visualizing Top 10 Most And Least Occurring Diseases In Specific Entity

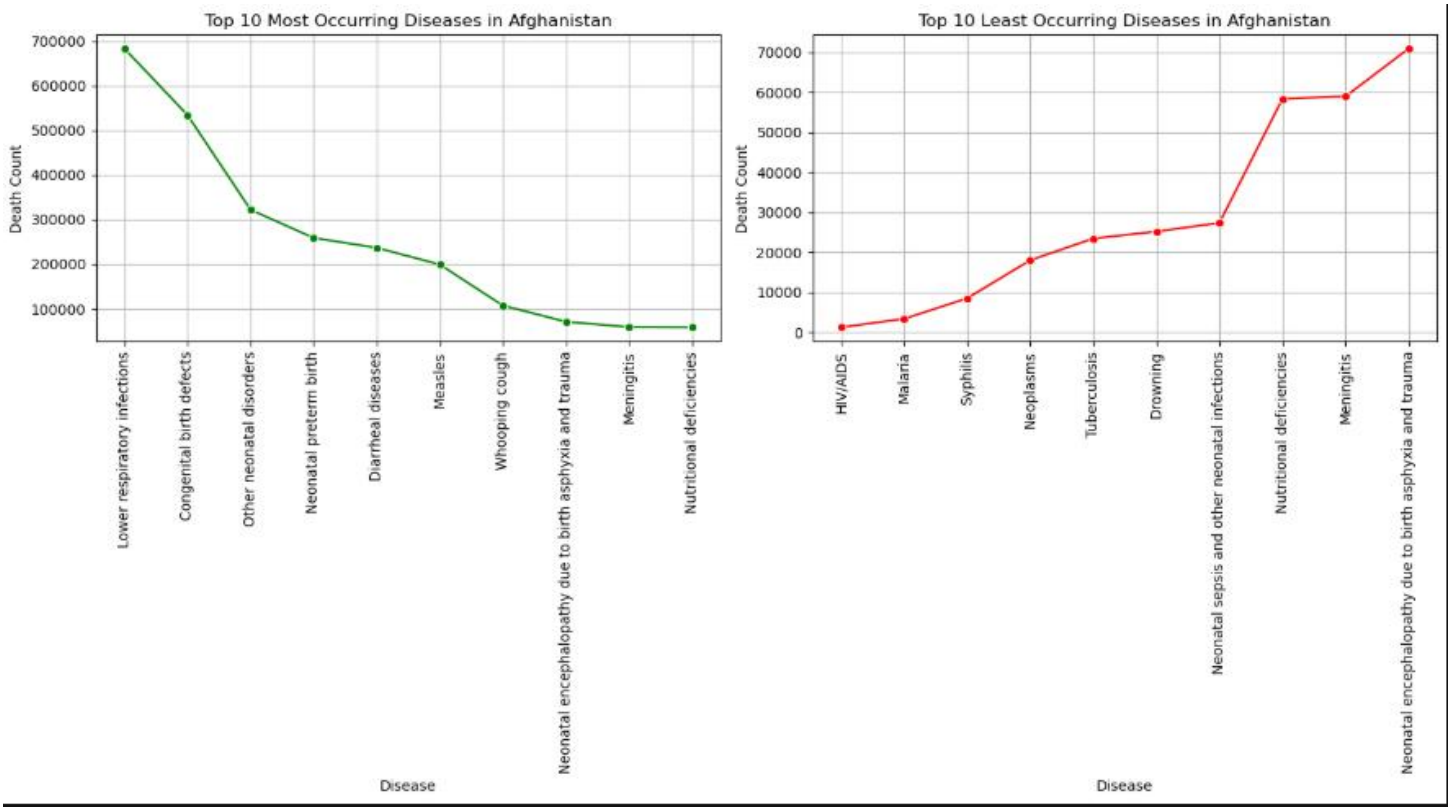

Figure 9.2-24 Output Of Figure: 9.2-23

```
# Ask user for the country/entity name
entity_name = input("Enter the Entity (e.g., Afghanistan, India, etc.): ")

# Filter data for the selected entity
entity_data = df[df['Entity'] == entity_name]

# Check if the entity data is not empty
if not entity_data.empty:

    disease_columns = [col for col in df.columns if col not in ['Entity', 'Code', 'Year']]

    disease_death_counts_entity = entity_data[disease_columns].sum()

    # Convert the result to a DataFrame for easier visualization
    disease_death_df = pd.DataFrame(disease_death_counts_entity).reset_index()
    disease_death_df.columns = ['Disease', 'Total Death Count']

    # Sort the diseases by total death count in descending order
    disease_death_df = disease_death_df.sort_values(by='Total Death Count', ascending=False)

    fig = px.bar(disease_death_df, x='Disease', y='Total Death Count',
                 title=f'Top Diseases Causing Death in {entity_name}',
                 labels={'Total Death Count': 'Number of Deaths', 'Disease': 'Disease'},
                 color='Total Death Count',
                 color_continuous_scale='Viridis')

    # Customize the chart
    fig.update_layout(
        xaxis_tickangle=-45,
        width=1200,
        height=800,
        font=dict(color='black'),
        hoverlabel=dict(font=dict(color='black'))
    )

    # Show the plot
    fig.show()

else:
    print(f"No data found for the entity: {entity_name}")
```

```
Enter the Entity (e.g., Afghanistan, India, etc.): Afghanistan
```

Figure 9.2-25: Visualization By Using Plotly-Sort The Diseases By Total Death Count For Specific Entity

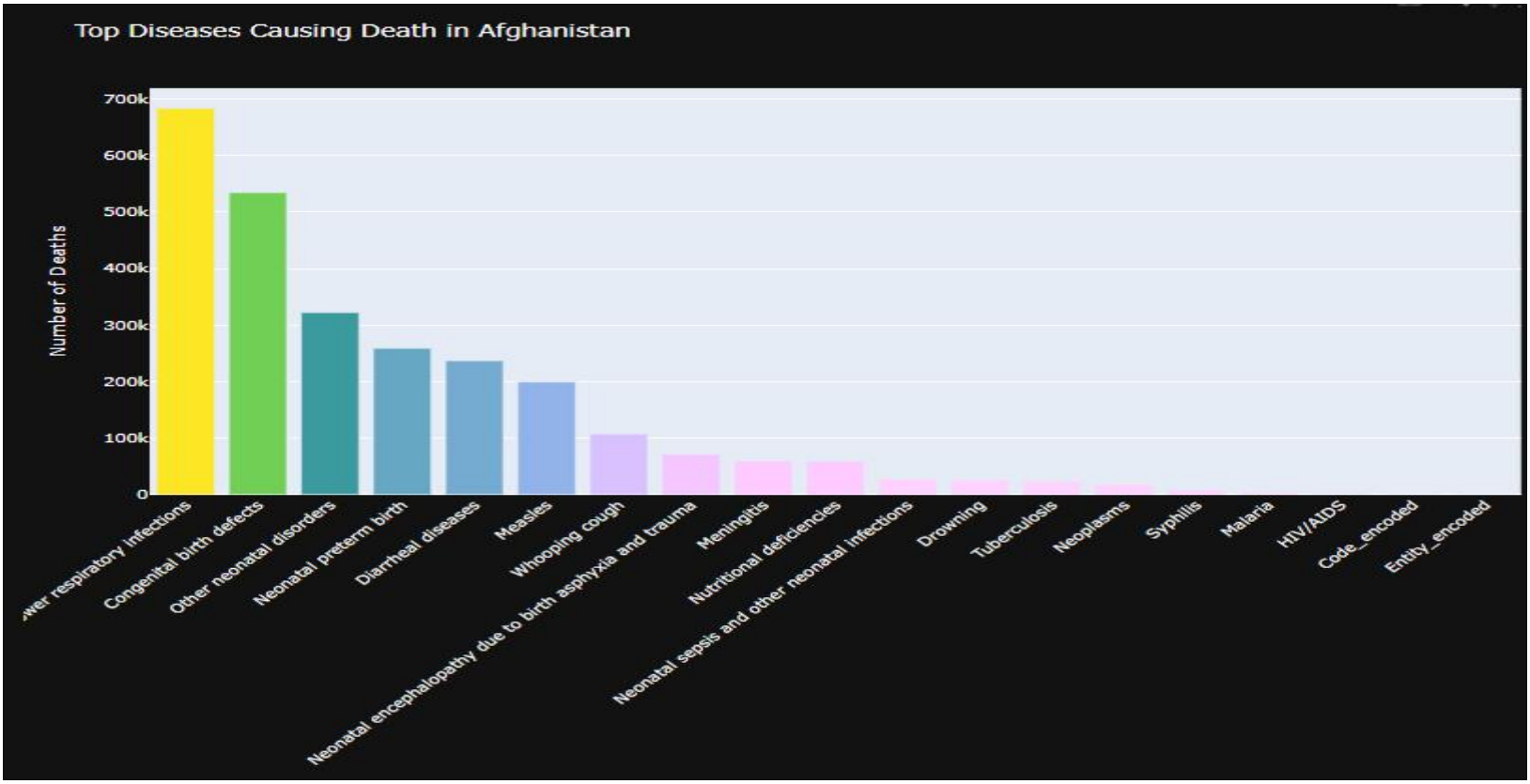

Figure 9.2-26: Output Of 9.2-25

```
year_input = int(input("Enter the year (e.g., 2019): "))
disease_input = input("Enter the disease (e.g., Tuberculosis): ")

data_year = df[df['Year'] == year_input]

if disease_input in data_year.columns:

    disease_deaths_year = data_year.groupby('Entity')[disease_input].sum().reset_index()

    top_10_disease_deaths = disease_deaths_year.sort_values(by=disease_input, ascending=False).head(10)

    fig = px.bar(top_10_disease_deaths,
                 x='Entity',
                 y=disease_input,
                 title=f'Top 10 Entities with Highest Deaths Due to {disease_input} in {year_input}',
                 labels={'Entity': 'Country/Entity', disease_input: f'Total Deaths due to {disease_input}'},
                 color=disease_input,
                 color_continuous_scale='Viridis')


    fig.update_layout(
        xaxis_tickangle=45,
        showlegend=False,
        width=1200,
        height=800
    )

    # Show the chart
    fig.show()

else:
    print(f"Error: {disease_input} data not available in the year {year_input}.")
```

```
Enter the year (e.g., 2019): 2018
Enter the disease (e.g., Tuberculosis): Malaria
```

Figure 9.2-27: By Using Plotly Graph - Taking Year And Diseases As User Input For Visualization

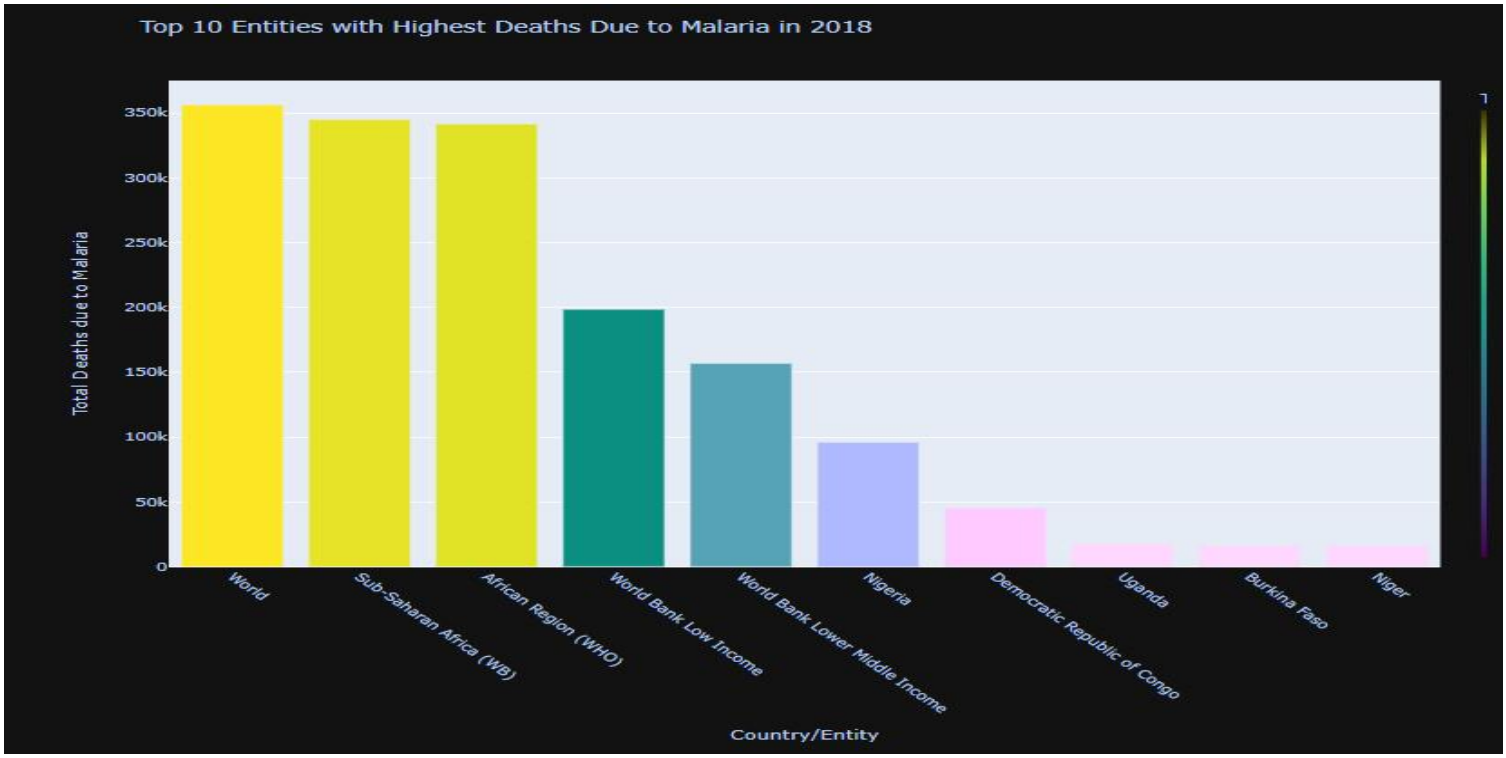

Figure 9.2-28: Output Of 9.2-27 Visualizing Top 10 Entities