

BDAS

Jinquan Wen id: 943155325

1 Business/Situation understanding

1.1 Business/Situation Objectives

In the past few years, the COVID-19 pandemic has been haunting our lives, and many people have lost their lives because of it. Bartsch and Ferguson(2021) concurred, analyzing global death counts and vaccine administration helps authorities understand the severity of the pandemic, aiding in the formulation of effective public health policies and interventions. Viboud et al.(2020) elaborated:" By comparing death rates and vaccination rates, disparities can be identified, enabling equitable distribution of resources to different regions." What's more, global analysis of COVID-19 outcomes and vaccination efforts promotes international cooperation and resource sharing. The objectives of the business

The goal of this business is to analyze the relationship between global deaths and vaccinations in the context of the Covid-19 pandemic, as well as the relationship between variables such as the number of vaccinations and the number of new cases and the total number of new cases (Tohid khan Bagani, 2023). Business success is measured by analytics that accurately reveal the relationship between COVID-19 mortality and vaccination rates, as well as the relationship between the number of vaccinations, the number of new cases, and the total number of new cases, using statistical and machine learning models The accuracy and predictive power of the analysis should be assessed, and the analysis results should be able

to support decision-making on public health policies and interventions to slow the spread of the epidemic and reduce mortality, by identifying the relationship between vaccination rates and COVID-19 infection and mortality, It can optimize resource allocation and ensure that resources are used fairly and effectively to mitigate the impact of the epidemic, as well as the fairness of resource allocation, coverage, and changes in infection and mortality rates after implementation. Promote cooperation and resource sharing among countries to jointly respond to the COVID-19 pandemic, ensure the quality, reliability and integrity of data, make the analysis results convincing, and create a data analysis framework that can be continuously updated and expanded to respond to the COVID-19 pandemic. Similar public health crises that may arise in the future.

1.2 Situation Assessment

Ensure the data itself is accurate and complete. Resources include global death toll data and vaccination data under covid-19. Extract Covid-19 death data from the first file. This may include total deaths and new deaths categorized by location and time periods. Using this data, we can create time trend charts to understand the overall pattern of Covid-19 deaths and identify potential peaks. Extract Covid-19 vaccination data from the second file. This may include total vaccinations, people vaccinated, people fully vaccinated, and total boosters categorized by location and time periods. The requirement is to assess the overall progress of global vaccination efforts and observe vaccination coverage and trends in different regions through data. Compare Covid-19 vaccination data with

Covid-19 death data. This can help us understand whether vaccination efforts have positively impacted the reduction of death rates. The assumption is that the number of deaths is affected differently by the number of vaccinations in different regions. Constraints are technical and time constraints for collecting data. The risk lies in discrepancies between the collected numbers of deaths and vaccinations and the true numbers. Some unexpected situations that may arise from the identified risks include: 1) Data quality and integrity issues. Data may be incomplete, inaccurate or out of date, which may affect the accuracy and reliability of the analysis.

2) Technology and tool issues. The data mining and analysis tools used may have flaws or limitations that affect the effectiveness and efficiency of the analysis.

3) Privacy and security issues. Data mining may involve sensitive personal and organizational information, raising the risk of data leakage and privacy invasion.

4) Political and regulatory issues. The political and regulatory environment in different countries and regions may affect the availability of data and the feasibility of analysis.

5) Resource and manpower issues. Large-scale data mining and analysis work may not be completed due to insufficient resources and manpower.

6) Communication and cooperation issues. International cooperation and resource sharing can be difficult because of communication and coordination issues.

7) Unknown and unpredictable situations. We may encounter unknown and

unpredictable situations, such as new epidemic outbreaks, natural disasters, etc.

1.3 Data Mining Objectives

The goal of data mining is to analyze and predict the relationship and correlation between the global number of deaths and the number of vaccinations in the context of the Covid-19 pandemic, as well as the correlation between variables such as the number of vaccinations and the number of new cases and the total number of new cases. Provide people with data support to help make better decisions and optimize resource allocation and utilization.

1.4 Project Plan

Project	Star time	Finish time
1.Business/Situation understanding	9.25	9.26
1.1 Identify the objectives of the business	9.25	9.26
1.2 Assess the situation	9. 25	9.26
1.3 Determine data mining objectives	9. 25	9.26
1.4 Produce a project plan	9. 25	9.26
2. Data understanding	9.26	9.28
2.1 Collect initial data	9.26	9.27
2.2 Describe the data	9.26	9.27
2.3 Explore the data	9.27	9.28
2.4 Verify the data quality	9.28	9.28
3. Data preparation	9.28	10.1
3.1 Select the data	9.28	9.29
3.2 Clean the data	9.28	9.29
3.3 Construct the data	9.29	9.30
3.4 Integrate various data sources	9.30	10.1
3.5 Format the data as required	9.30	10.1
4. Data transformation	10.1	10.3
4.1 Reduce the data	10.1	10.2

4.2 Project the data	10.2	10.3
5. Data-mining method(s) selection	10.3	10.4
5.1 Match and discuss the objectives of data mining (1.1) to data mining methods	10.3	10.4
5.2 Select the appropriate data-mining method(s) based on discussion	10.4	10.4
6. Data-mining algorithm(s) selection	10.4	10.6
6.1 Conduct exploratory analysis and discuss	10.4	10.5
6.2 Select data-mining algorithms based on discussion	10.4	10.5
6.3 Build/Select appropriate model(s) and choose relevant parameter(s)	10.5	10.6
7. Data Mining	10.6	10.8
7.1 Create and justify test designs	10.6	10.7
7.2 Conduct data mining	10.7	10.8
7.3 Search for patterns	10.7	10.8
8. Interpretation	10.8	10.12
8.1 Study and discuss the mined patterns	10.8	10.9
8.2 Visualize the data, results, models, and patterns	10.8	10.9
8.3 Interpret the results, models, and patterns	10.9	10.10
8.4 Assess and evaluate results, models, and patterns	10.10	10.11
8.5 Iterate prior steps (1 – 7) as required	10.11	10.12

2. Data understanding

2.1 Collect initial data

The source of the data set is a data website of kaggle. The website address is :

<https://www.kaggle.com/datasets/tohidkhanbagani/covid-19-deaths->

[and-vaccinations-dataset?select=COVID+DEATHS.csv.](#)

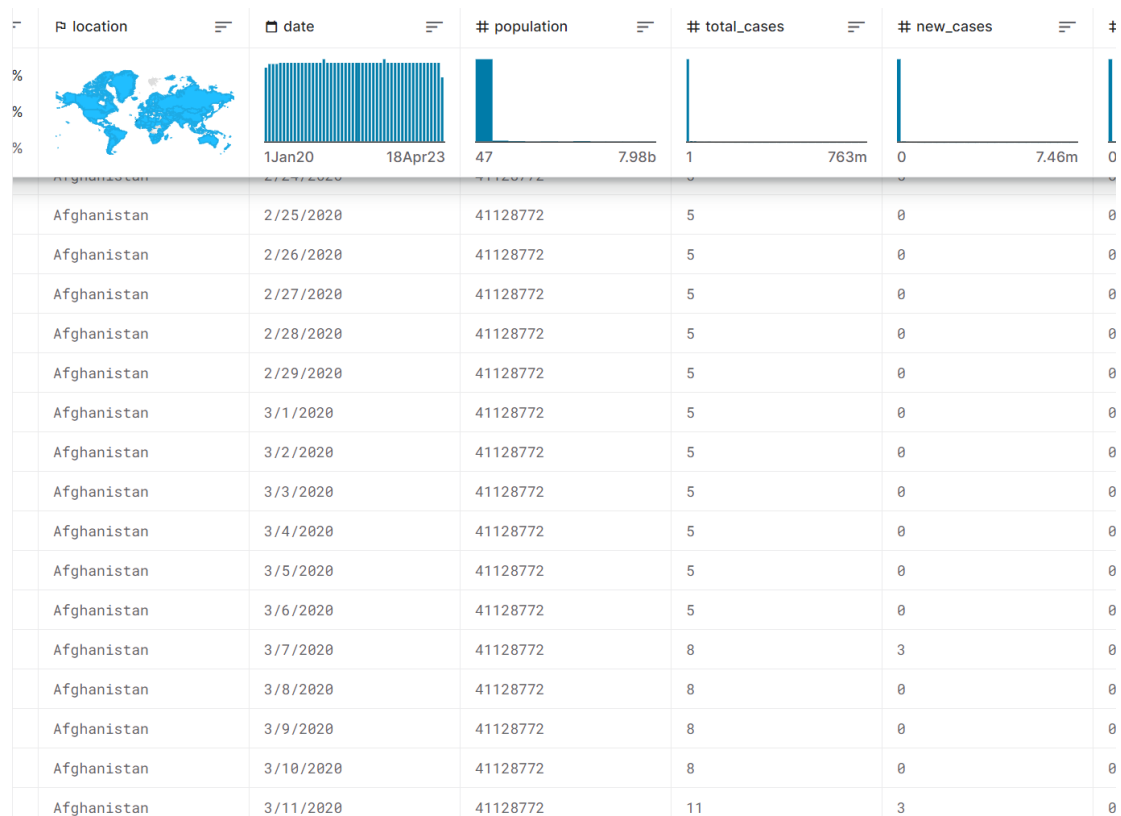


Figure 1 Covid death toll

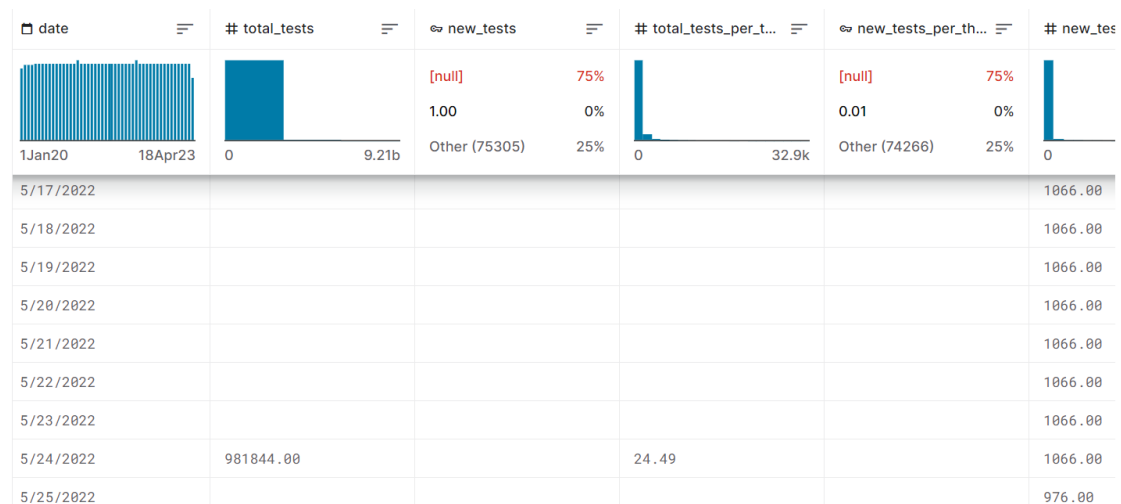


Figure 2 Covid vaccina

Some other data on covid-19 and vaccination status can also be collected on the Johns Hopkins University website, <https://coronavirus.jhu.edu/map.html>. The difficulty of collection is that the data set is huge, which is not conducive to checking the correctness, and the data may be missing. The method of data

collection is csv file format, which is counted and collected by the World Health Organization and Johns Hopkins University website based on the number of deaths and vaccinations provided by each country.

2.2 Data Description

The data includes two data sets, where includes the death data such as iso_code: the ISO 3166-1 alpha-3 code of the country or territory. Continent: the continent of the location. Location: the name of the country or territory. Date: the date of the observation. Population: the population of the country or territory. Total_cases: the total number of confirmed cases of Covid-19. New_cases: the number of new confirmed cases of Covid-19. Total_deaths: the total number of deaths due to Covid-19. New_deaths: the number of new deaths due to Covid-19. COVID VACCINATIONS includes data such as Total_tests: the total number of tests for Covid-19. New_tests: the number of new tests for Covid-19. Total_tests_per_thousand: the total number of tests for Covid-19 per thousand people. New_tests_per_thousand: the number of new tests for Covid-19 per thousand people. New_tests_smoothed: the 7-day smoothed average of new tests for Covid-19. New_tests_smoothed_per_thousand: the 7-day smoothed average of new tests for Covid-19 per thousand people. Positive_rate: the share of Covid-19 tests that are positive, given as a rolling 7-day average. Tests_per_case: the number of tests conducted per confirmed case of Covid-19, given as a rolling 7-day average. Tests_units: the units used by the location to report its testing data. Total_vaccinations: the total number of doses of Covid-19

vaccines administered. The data is presented in csv file format.

2.3 Data Exploration

Use spark for data visualization. Load the table data and visualize it as shown in Figure 3. Use a scatter plot to visualize the relationship between the total number of deaths and the total number of cases, as shown in Figure 4.

iso_code	continent	location	date	population	total_cases	new_cases	total_deaths	new_deaths	total_vaccinations	new_cases_smoothed	new_deaths_smoothed	total_cases_per_million	new_cases_per_million	new_cases_smoothed_per_million	total_deaths_per_million	new_deaths_per_million	new_deaths_smoothed_per_million	reproduction_rate	icu_patients	icu_patients_per_million	hosp_patients	hosp_patients_per_million	weekly_icu_admissions	weekly_icu_admissions_per_million	weekly_hosp_admissions	weekly_hosp_admissions_per_million
AFG	Asia	Afghanistan	1/3/2020	41128772	null	0	null	0	null	null	0.0	null	null	null	0.0	null	0.0	null	null	null	null	null	0.0	null	0.0	0.0
AFG	Asia	Afghanistan	1/4/2020	41128772	null	0	null	0	null	null	0.0	null	null	null	0.0	null	0.0	null	null	null	null	null	0.0	null	0.0	0.0
AFG	Asia	Afghanistan	1/5/2020	41128772	null	0	null	0	null	null	0.0	null	null	null	0.0	null	0.0	null	null	null	null	null	0.0	null	0.0	0.0
AFG	Asia	Afghanistan	1/6/2020	41128772	null	0	null	0	null	null	0.0	null	null	null	0.0	null	0.0	null	null	null	null	null	0.0	null	0.0	0.0
AFG	Asia	Afghanistan	1/7/2020	41128772	null	0	null	0	null	null	0.0	null	null	null	0.0	null	0.0	null	null	null	null	null	0.0	null	0.0	0.0
AFG	Asia	Afghanistan	1/8/2020	41128772	null	0	null	0	null	null	0.0	null	null	null	0.0	null	0.0	null	null	null	null	null	0.0	null	0.0	0.0
AFG	Asia	Afghanistan	1/9/2020	41128772	null	0	null	0	null	null	0.0	null	null	null	0.0	null	0.0	null	null	null	null	null	0.0	null	0.0	0.0
AFG	Asia	Afghanistan	1/10/2020	41128772	null	0	null	0	null	null	0.0	null	null	null	0.0	null	0.0	null	null	null	null	null	0.0	null	0.0	0.0

Figure 3 original table

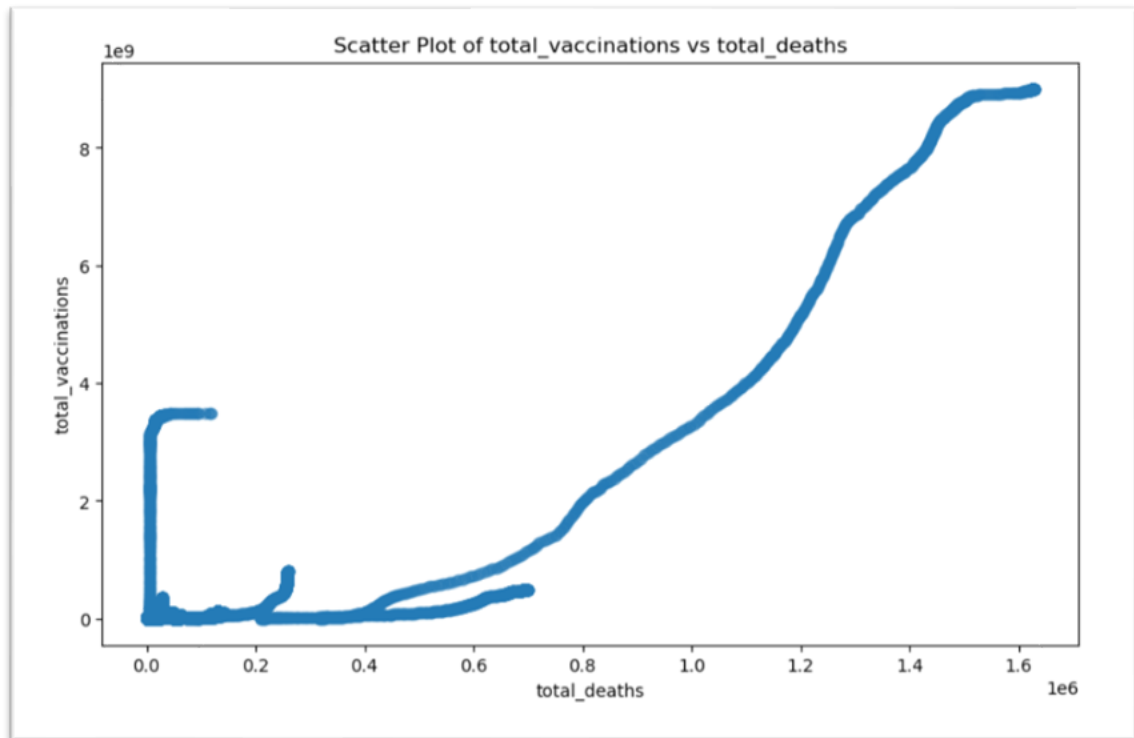


Figure 4 Scatter plot of deaths and cases

The distribution of the total number of deaths can be seen through the histogram, as shown in Figure 5

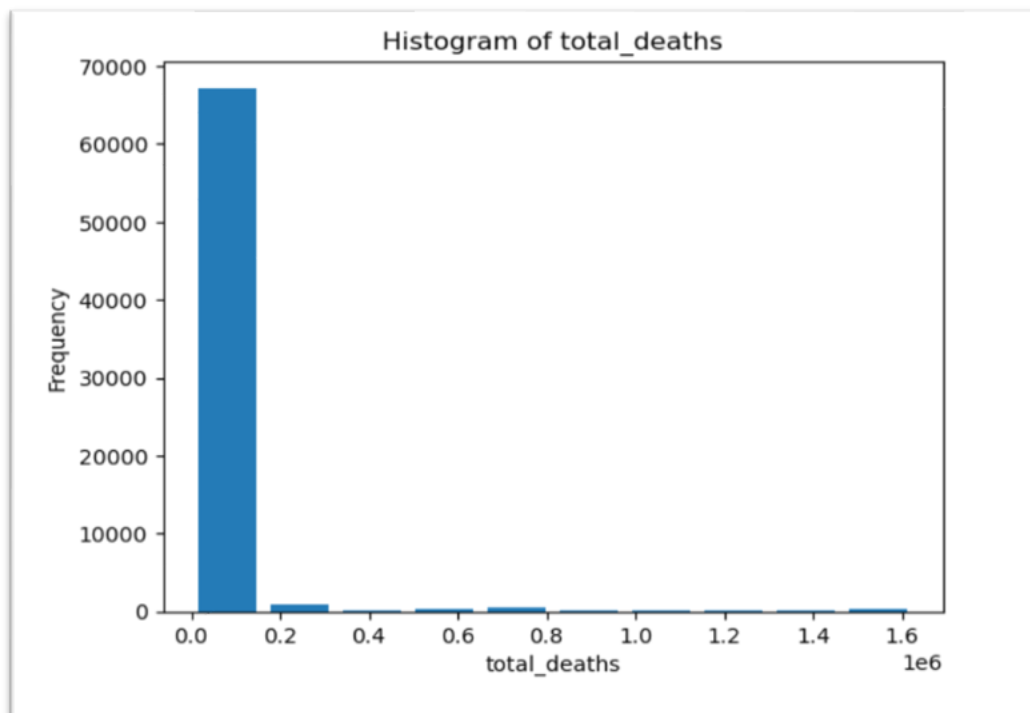


Figure 5 distribution of total deaths

The distribution of the new_deaths can be seen through the histogram, as shown in Figure 6.

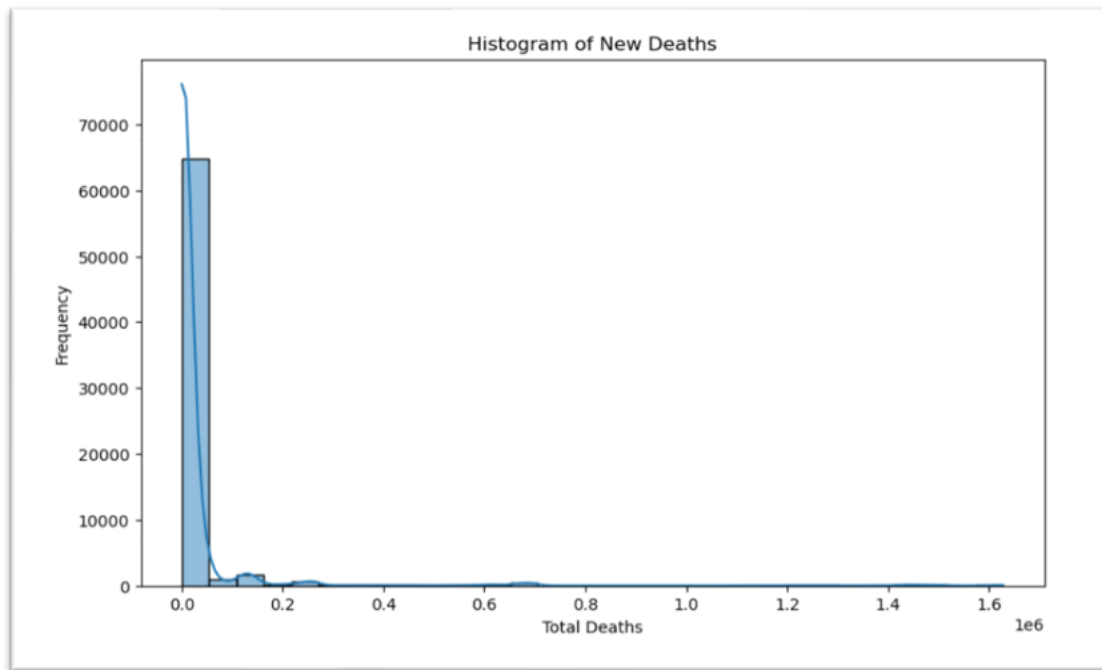


Figure 6 distribution of new deaths

In spark model, the relationship between each data variable and other variable types can be visualized to help better understand the original data. For example, in the death table, the relationship between the total number of deaths and new cases is visualized, etc., while in the vaccination table, the table can be visualized based on the number of vaccinations.

2.4 Verifying Data Quality

The missing values of the data can be seen in the plot.

iso_code	continent	location	date	population	total_cases	new_cases	total_deaths	new_deaths	total_vaccinations	new_cases_smoothed	new_deaths_smoothed	total_cases_per_million	new_cases_per_million	new_cases_smoothed_per_million	total_deaths_per_million	new_deaths_per_million	new_deaths_smoothed_per_million	reproduction_rate	icu_patients	icu_patients_per_million	hosp_patients	hosp_patients_per_million	weekly_icu_admissions	weekly_icu_admissions_per_million	weekly_hosp_admissions	weekly_hosp_admissions_per_million
1511	0	2397	0	0	0	6638	1201	10318	1181	60510	1536	1511	25613	6638	1201	10318	1181	60510	1536	1511	25613	6638	1201	10318	1181	
1511	25613	6638	68098	1201	68098	69849	1536	69849	10318	77767	1181	1511	25613	6638	1201	10318	1181	60510	1536	1511	25613	6638	1201	10318	1181	
77767	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	74243	

Figure 7 missing value

Check data quality by checking data types, as shown in Figure 8 .

```
[('iso_code', 'string'), ('continent', 'string'), ('location', 'string'), ('date', 'string'), ('population', 'bigint'), ('total_cases', 'int'), ('new_cases', 'int'), ('total_deaths', 'int'), ('new_deaths', 'int'), ('total_vaccinations', 'bigint'), ('new_cases_smoothed', 'double'), ('new_deaths_smoothed', 'double'), ('total_cases_per_million', 'double'), ('new_cases_per_million', 'double'), ('new_cases_smoothed_per_million', 'double'), ('total_deaths_per_million', 'double'), ('new_deaths_per_million', 'double'), ('new_deaths_smoothed_per_million', 'double'), ('reproduction_rate', 'double'), ('icu_patients', 'int'), ('icu_patients_per_million', 'double'), ('hosp_patients', 'int'), ('hosp_patients_per_million', 'double'), ('weekly_icu_admissions', 'int'), ('weekly_icu_admissions_per_million', 'double'), ('weekly_hosp_admissions', 'int'), ('weekly_hosp_admissions_per_million', 'double'), ('total_cases_bins', 'string')]
```

Figure 8 Data types

Use descriptive statistics to display basic statistics of a DataFrame, as shown in Figure 9.

summary	iso_code	continent	location	date	population	total_cases	new_cases	total_deaths	n
ew_deaths	total_vaccinations	new_cases_smoothed	new_deaths_smoothed	total_cases_per_million	new_cases_per_million	new_cases_smoothed_per_m	total_deaths_per_million	new_deaths_per_million	new_deaths_smoothed_per_m
reproduction_rate	icu_patients	icu_patients_per_million	hosp_patients	hosp_patients_per_million	weekly_icu_admissions	weekly_icu_admissions_per_million	weekly_hosp_admissions	weekly_hosp_admissions_per_million	
count	79999	77602	79999	79999	79999	73361	78798	69681	784
78818	19489	78463	78488	78488	73361	78798	69681	784	
63	69681	78818	78488	54386	11901	5756			
11901	10150	2232	2232	5756					
5756									
mean	NULL	NULL	NULL	NULL	1.3238519732122901E8	2599885.369487875	6593.51105358004	32717.455777041087	44.44653
505544419	3.790593589407358E8	6619.746225890005	44.6271046656824	78189.16712536696	152.44767911621966	153.06154323			
694045	769.0143559506887	1.06583961785379	1.0697371190500264	0.8781842753649899	433.176203680363	19.632266			
448197566	2153.3720197044336	128.19332571428617	253.34408602150538	16.746282706093176	2130.6002432244613				
103.69756497567762									
stddev	NULL	NULL	NULL	NULL	6.162162797372026E8	1.692044887252384E7	103652.34240168534	151182.30912148565	275.5572
721943506	1.4127079037202768E9	100447.26479401716	265.5331224464298	129543.87963002999	720.3421069716459	489.148650			
624793	1059.6108524433075	5.203168251442458	2.9962354693844975	0.40161470993438086	888.7073628392994	30.065411			
457446473	3336.7568913690848	147.21885034011123	326.9203649667642	18.895641286195882	3207.858228491786				
114.72411869417482									
min	ABW	Africa	Afghanistan	1/1/2020	15877	1	0	1	0.0
0	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	-0.07	0	0.0	0.0	0
0.0	0.0	0	0.0	0.0	0.0	0	0	0.0	0.0
max	VGB	South America	Eritrea	9/9/2022	4721383370	295846519	7214500	1628478	9383.
11447	9003468005	6109992.571	5718.857	731762.14	31227.263	7969	26682	18	
35	5644.685	583.919	87.155	3.68	86.769				
0.675	34336	1526.846	1701						
708.12									

Figure 9 DataFrame describe

Use graphics such as boxplot to explore relationships between, for example, total deaths and total cases to examine data distribution and potential outliers.

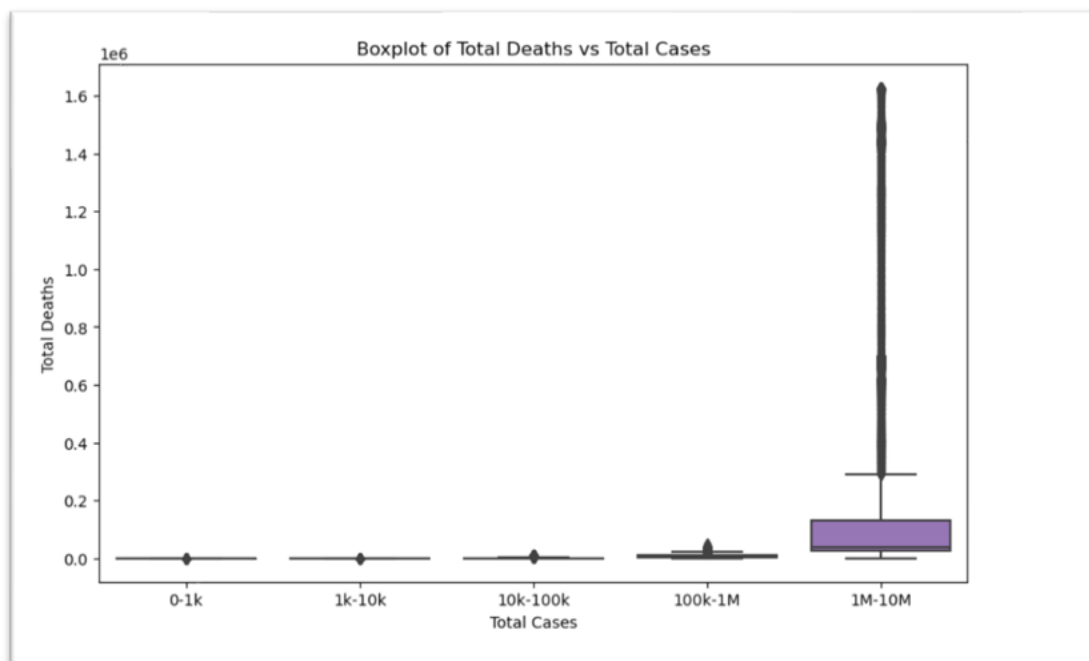


Figure 10 boxplot of total deaths and total cases

3. Data preparation

3.1 Data selection

The research goal is the situation between death and vaccination, which may be affected by region and time. Observe the number of deaths in different places and at different times in the data set and analyze the number of vaccinations in the same region and time constraints, so as to analyze the relationship between the number of deaths and the number of vaccinations. As shown below:

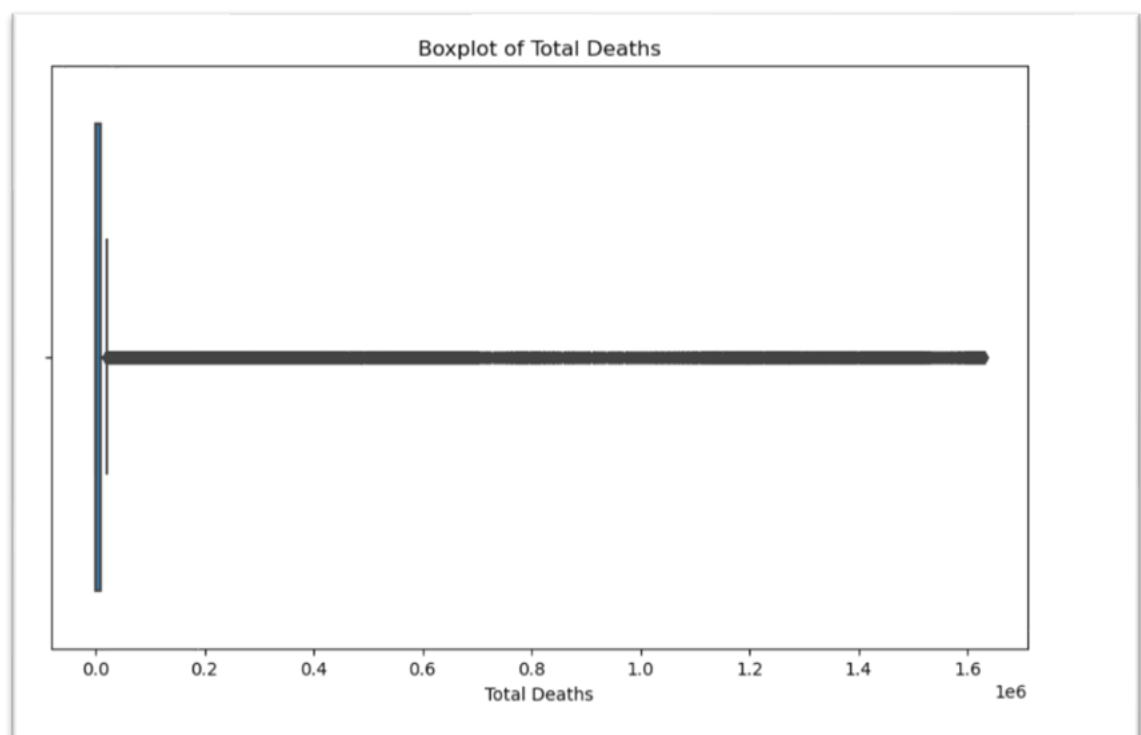


Figure 11 audit in total deaths

In data quality, this topic has been mentioned in the 2.4 process. Use the boxplot to detect outliers in the data set, the skew of the distribution, and the comparison between different groups of data to understand the central trend of the data and Degree of dispersion. Or use a scatterplot to view the relationship

between two variables to explore relationships between variables, observe the distribution of data, and identify outlier observations. You can also compare the indicator with the threshold to determine whether the requirements of the data quality mode are met.

In terms of technical limitations, the huge data set may cause difficulties in calculation and analysis, etc. It is necessary to ensure the efficiency of computing resources to deal with the complexity and volume of data. And ensure enough data storage space to store data.

3.2 Data cleaning

There are many null values in the table, so it needs to be cleaned up. The original data table is shown in the figure 12, and it can be seen that there are 35978 null values in the total cases. Filling missing values by using the median method, as shown in Figure 13 for the target column.

iso_code	continent	location	date	population	total_cases	new_cases	total_deaths	new_deaths	total_vaccinations	new_cases_smoothed	new_deaths_smoothed	total_cases_per_million	new_cases_per_million	new_cases_smoothed_per_million	total_deaths_per_million	new_deaths_per_million	new_deaths_smoothed_per_million	reproduction_rate	icu_patients	icu_patients_per_million	hosp_patients	hosp_patients_per_million	weekly_icu_admissions	weekly_icu_admissions_per_million	weekly_hosp_admissions	weekly_hosp_admissions_per_million
0	2397	0	0	0	6638	1201	10318	1181	60510	1536	1181	1511	25613	68098	68098	69849	69849	77767	74243	74243	74243	74243	74243	74243	74243	74243
1511	1511	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767	7767

Figure 12 original data

total_cases	new_cases	total_deaths	new_deaths	total_vaccinations
0	0	0	0	0
total_cases	new_cases	total_deaths	new_deaths	total_vaccinations
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7
50574.0	0.0	811.0	0.0	1.328689E7

only showing top 20 rows

Figure 13 Cleaned data

Check for outliers and use the IQR method or other methods to handle outliers, as shown in Figure 14.

```
No outliers detected for total_cases.
No outliers detected for new_cases.
No outliers detected for total_deaths.
No outliers detected for new_deaths.
No outliers detected for total_vaccinations.
```

Figure 14 Handle outliers

3.3 Construct the data

Create new variables from existing variables. Create a variable representing the death rate, which is `total_deaths` divided by `total_cases`. The new variable created is shown in Figure 15 .


```

+-----+-----+
| summary | mortality_rate |
+-----+-----+
| count | 79999 |
| mean | 5.439244946005151 |
| stddev | 49.45436514353395 |
| min | 1.977300589235575... |
| max | 811.0 |
+-----+-----+

```

```

+-----+
| mortality_rate |
+-----+
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
| 0.01603590777870052 |
+-----+

```

only showing top 20 rows

Figure 15 node identify

3.4 Data integrating

Use the location shared column in the two data tables as the connection key, and use PySpark's join function to merge the death table and vaccination table. After merging, as shown in the figure below, you can see that 'total_tests','new_tests' in the vaccination table and 'total_cases', 'new_cases' in the death table have been merged into one table.

iso_code	continent	location	date	population	total_cases	new_cases	total_deaths	new_deaths	total_vaccinations	new_cases_smoothed	new_deaths_smoothed	total_cases_per_million	new_cases_per_million	new_cases_smoothed_per_million	total_deaths_per_million	new_deaths_per_million	new_deaths_smoothed_per_million	reproduction_rate	icu_patients	icu_patients_per_million	hosp_patients	hosp_patients_per_million	weekly_icu_admissions	weekly_icu_admissions_per_million	weekly_hosp_admissions	weekly_hosp_admissions_per_million	total_cases_bins	mortality_rate	total_tests	new_tests
----------	-----------	----------	------	------------	-------------	-----------	--------------	------------	--------------------	--------------------	---------------------	-------------------------	-----------------------	--------------------------------	--------------------------	------------------------	---------------------------------	-------------------	--------------	--------------------------	---------------	---------------------------	-----------------------	-----------------------------------	------------------------	------------------------------------	------------------	----------------	-------------	-----------

Figure 16 merge data

Perform missing value processing on total_vaccinations and use the mean to fill the missing values in the total_vaccinations column. By using the IOR method to handle outliers, first calculate the IQR, define limits to identify outliers, then identify outliers, and then choose the method of replacing outliers with medians to handle outliers. Checking outliers and missing values is shown in Figures 17 and 18. The final result after processing is shown in Figure 19.

```

population number of missing value: 0
total_cases number of missing value: 0
new_cases number of missing value: 0
total_deaths number of missing value: 0
new_deaths number of missing value: 0
total_vaccinations number of missing value: 0

```

Figure 17 after missing imputing

area-num	iso_code_numeric
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
AFG	10.0
only showing top 20 rows	

Figure 20 define type and string conversion

Output the data type check, and you can see the newly added columns and type conversions in the table, as shown in Figure 21.

```
[('area-num', 'string'), ('continent', 'string'), ('location', 'string'), ('date', 'string'), ('population', 'bigint'), ('total_cases', 'double'), ('new_cases', 'double'), ('total_deaths', 'double'), ('new_deaths', 'double'), ('total_vaccinations', 'double'), ('new_cases_smoothed', 'double'), ('new_deaths_smoothed', 'double'), ('total_cases_per_million', 'double'), ('new_cases_per_million', 'double'), ('new_cases_smoothed_per_million', 'double'), ('total_deaths_per_million', 'double'), ('new_deaths_per_million', 'double'), ('new_deaths_smoothed_per_million', 'double'), ('reproduction_rate', 'double'), ('icu_patients', 'int'), ('icu_patients_per_million', 'double'), ('hosp_patients', 'int'), ('hosp_patients_per_million', 'double'), ('weekly_icu_admissions', 'int'), ('weekly_icu_admissions_per_million', 'double'), ('weekly_hosp_admissions', 'int'), ('weekly_hosp_admissions_per_million', 'double'), ('mortality_rate', 'double'), ('total_tests', 'int'), ('new_tests', 'int'), ('iso_code_numeric', 'double')]
```

Figure 21 Type of merged table

4. Data transformation

4.1 Data reducing

According to the selection of feature selection to reduce the complexity of the model, improve the performance and generalization ability of the model, and reduce the risk of overfitting. Horizontal reduction involves reducing the number of features in a dataset by selecting features that are most relevant to the predictor variables. First use statistical methods to find the features most relevant to the target variable. Use Pearson or Spearman correlation coefficients to evaluate the relationship between features and target variables. Use feature selection methods, such as recursive feature elimination (RFE), to find the most important features. As shown in Figure 22.

```
Correlation between total_deaths and total_deaths: 1.0
Correlation between total_deaths and population: 0.30719110047649373
Correlation between total_deaths and total_cases: 0.9045693426476185
Correlation between total_deaths and new_cases: 0.6196742829602869
Correlation between total_deaths and new_deaths: 0.5977172297419507
Correlation between total_deaths and total_vaccinations: 0.12103421040283768
```

Figure 22 model of feature selection

To ensure that the proportions of certain categorical variables in the sample are the same as in the original data set, stratified sampling can be used. Use

'stat.sampleBy' to stratify sampling to reduce size, and use the select function in pyspark to keep the columns you need and remove some obviously irrelevant or redundant columns. As shown in Figure 23.

population	total_cases	new_cases	total_deaths	new_deaths	total_vaccinations	mortality_rate	iso_code_numeric
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0
41128772	50574.0	0.0	811.0	0.0	3.790593589407358E8	0.01603590777870052	10.0

only showing top 20 rows

Figure 23 reducing model

4.2 Project the data

First select the features that need to be transformed, use total_deaths as the feature column, make sure there are no zero or negative values in the 'total_deaths' column, apply logarithmic transformation to the selected features, and then check the effect of the transformation through data visualization, as shown in Figure 24 . Logarithmic transformation, etc. are shown in Figure 25. Vectorize the features according to the VectorAssembler and assembler.transform functions, divide the data set into test and training sets, build a linear regression model to train the model, and then test it on the test set. The prediction results output the RMSE and Coefficients evaluation model, as shown in Figure 26 Show.

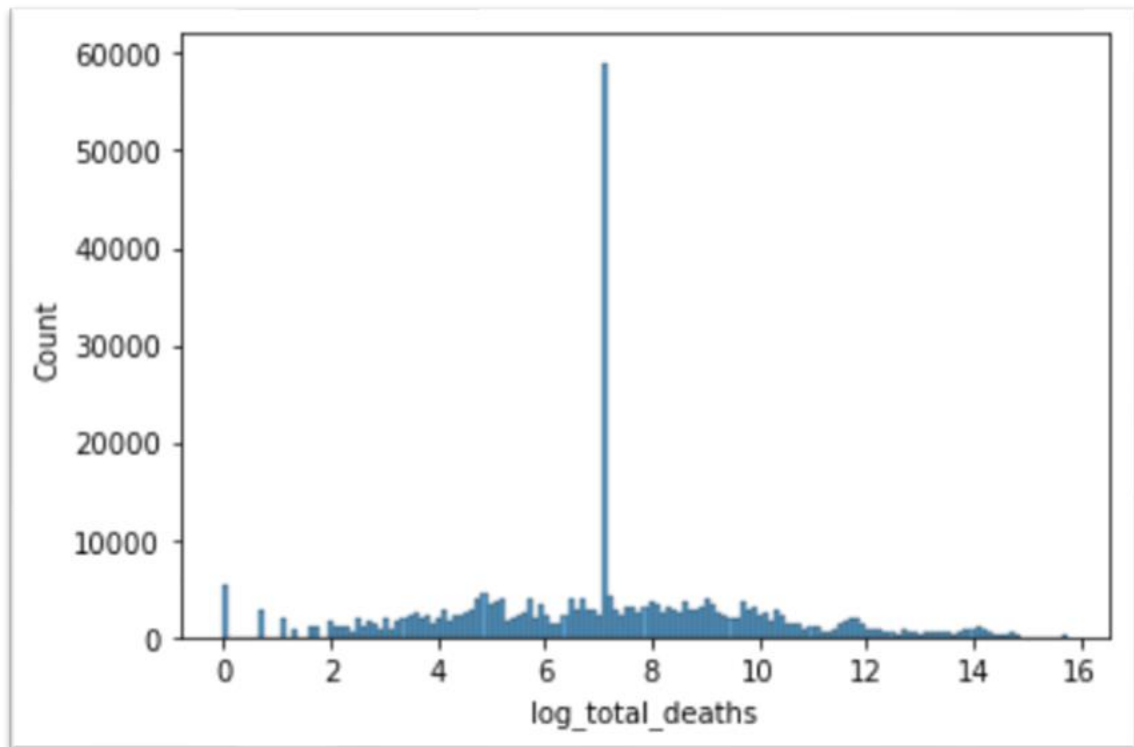


Figure 24 hisplot of threshold

```
selected_columns = ["total_deaths", "population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
df_cleaned = df_cleaned.select(selected_columns)

for col_name in ["total_cases", "new_cases", "new_deaths", "total_vaccinations", "population"]:
    df_cleaned = df_cleaned.withColumn(f"log_{col_name}", log(col(col_name) + 1)) # 添加一个小值以避免负无穷大

feature_columns = ["log_population", "log_total_cases", "log_new_cases", "log_new_deaths", "log_total_vaccinations"]
assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")
df_features = assembler.transform(df_cleaned)

train_data, test_data = df_features.randomSplit([0.8, 0.2], seed=42)

lr = LinearRegression(featuresCol="features", labelCol="total_deaths")

lr_model = lr.fit(train_data)

|
predictions = lr_model.transform(test_data)
```

Figure 25 Model prediction

```
Root Mean Squared Error (RMSE): 4287.103025836265
Coefficients: [462.1758965043022, 943.4033274369991, -74.53980353357201, 1857.40503461671, -135.7747585942073]
Intercept: -11117.901534451128
```

Figure 26 Prediction results

5. Data-mining method selection

5.1 Match and discuss

The purpose of this work is to explore the relationship between deaths and vaccination, as well as other variables, and to predict the correlation between these variables. In supervised ML, regression algorithms are used to predict continuous-valued outputs. In supervised learning, a model is trained using known input features and corresponding output labels to learn the relationship between features and outputs. James and Witten et al(2013) said that , a regression algorithm is an algorithm that predicts continuous variables. Basically, it attempts to find the relationship between one variable and one or more other variables. Linear regression is a suitable method to explore the relationship between the number of deaths and the number of vaccinations in an outbreak. Linear models were built to predict the number of deaths as a function of the number of vaccinations administered and in relation to other variables.

Bishop(2006) concluded, that A classification algorithm is an algorithm for predicting discrete variables and it is the process of assigning a given data point to a predefined class or category. Provost and Fawcett(2013) occurred that, A classification algorithm is a method of supervised learning that predicts the classification label of new instances based on labeled instances in a training data set. Classification algorithms are used to assign input data to predefined categories or labels, and the outputs are discrete categories. Models classify by learning relationships between features and categories in the training data. Commonly used algorithms include Logistic Regression, Decision Trees, Random

Forests, etc. The goal of this kind of mining is numerical output, which is also suitable for this mining goal..

Xu, Rui, and Donald(2010) elaborated "a clustering algorithm is an algorithm for grouping data points, where points within a group are more similar than points between groups." Clustering algorithms group input data into similar clusters, where data within each cluster are more similar and data between different clusters are less similar. This is an unsupervised learning method. It groups data based on similarity. Commonly used algorithms include K-Means Clustering, Hierarchical Clustering, etc. Anomaly detection algorithms are used to identify outlier data points that are significantly different from other data points and can be supervised or unsupervised depending on the presence of labeled outlier data. Correlation analysis algorithms are used to discover correlations between items in a data set and find common sets of items. The algorithm is also better suited to the relationship between the number of vaccinations and the number of deaths.

5.2 Data-mining method(s) selection

The goal of this work is to study and predict the relationship between the number of deaths during the epidemic and other variables such as the number of vaccinations. Other variables such as the total number of cases and their output results are of numeric type. Classify according to the learned relationship between features and categories in the training data. Assume that the independent variable `total_deaths` is assigned a predefined class, such as low and high death, to explore and predict the relationship between these variables. The predictive relationship

between them can be well derived by observing the significant coefficients between them. It is reasonable in this study to examine the relationship between the number of deaths and other variables such as the number of vaccinations, so that an increase in the number of vaccinations may be inversely related to a decrease in the number of deaths. Provost et al. (2013) agreed that the coefficients of this algorithmic model represent the relationship between independent variables and dependent variables, making the model results easy to interpret, and important coefficients can be analyzed to understand the relationship between them. The algorithm provides reliable results and interpretability. So choose the Supervised ML - Decision Tree algorithm.

6. Data-mining algorithm(s) selection

6.1 Conduct exploratory analysis and discuss

By selecting feature selection, you can select the most representative and relevant features from the original feature set for model construction and prediction, which helps to reduce the complexity of the model, improve the performance and generalization ability of the model, and reduce the risk of over-fitting. Data preparation and preprocessing, followed by exploratory data analysis, using histograms to understand the distribution of data, as shown in Figure 27. Scatter plots to understand the relationship between two variables, such as the relationship between the total number of deaths and the number of vaccinations. As shown in Figure 28. Finally, feature selection is performed. `total_deaths` is the

target variable and also used as feature selection. As shown in Figure 29. The features related to the target variable are shown in Figure 30.

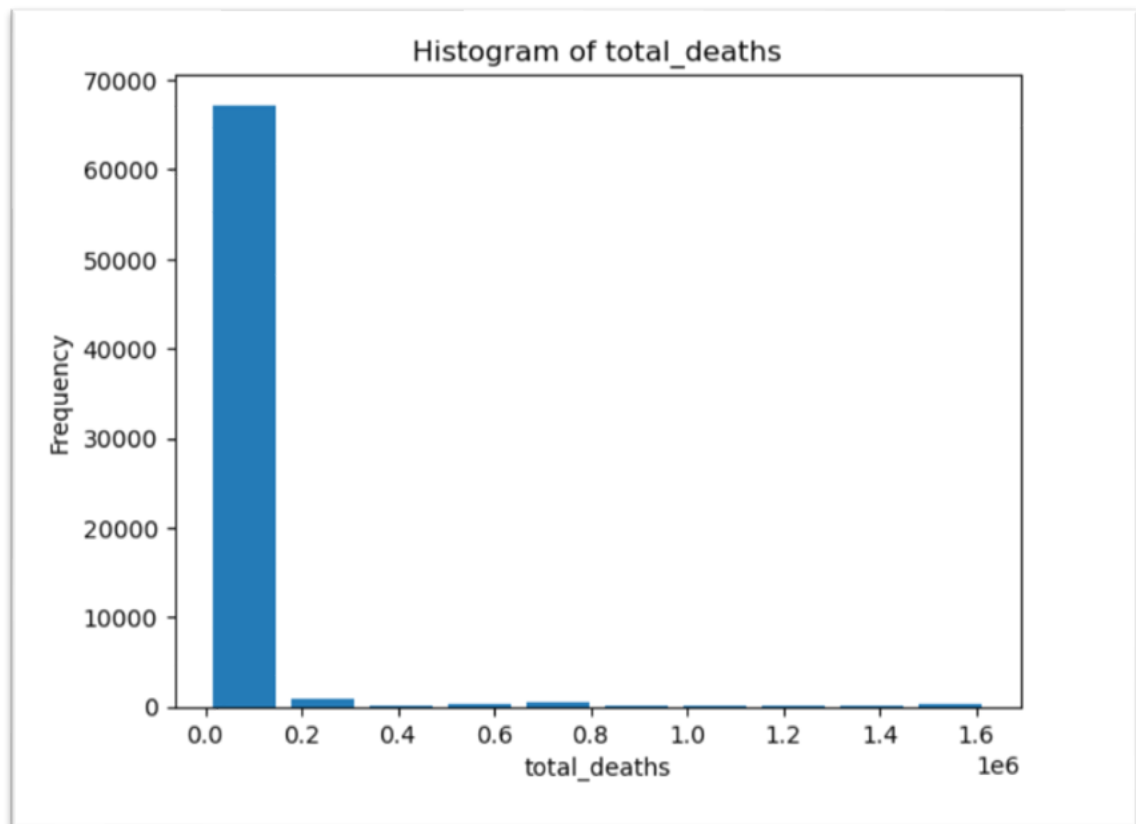


Figure 27 histplot of total deaths

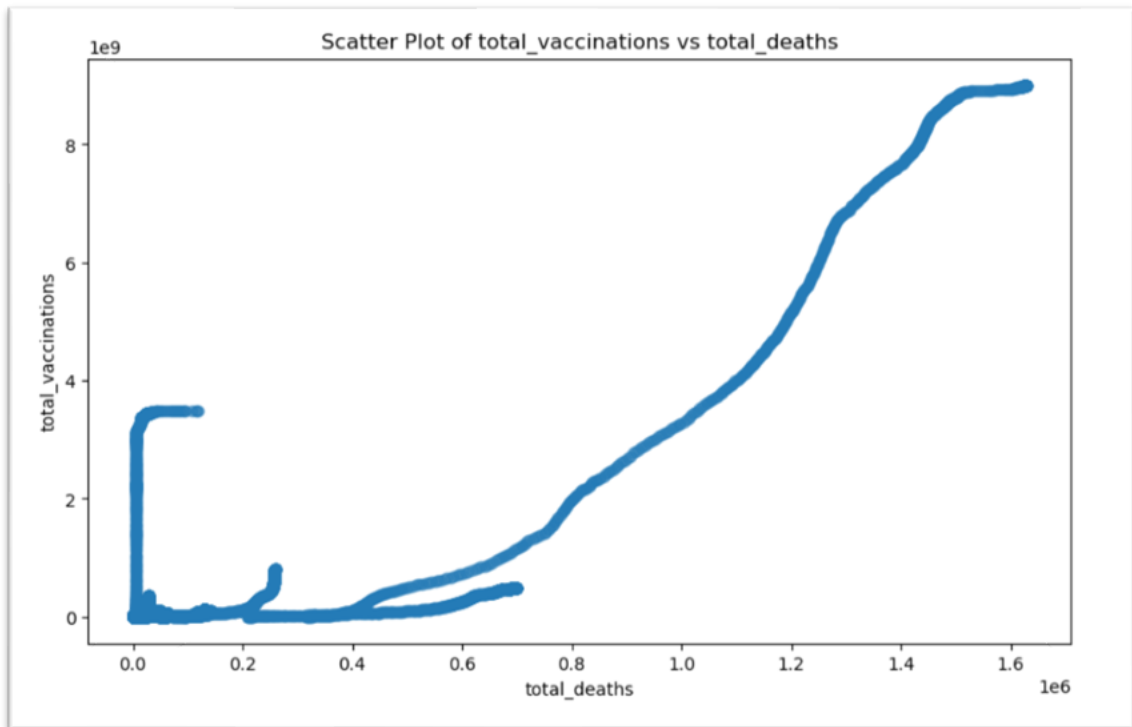


Figure 28 scatterplot of vaccinations and deaths

```
for i, col in enumerate(numerical_cols):
    print(f"Correlation between total_deaths and {col}: {corr_matrix[i][numerical_cols.index('total_deaths')]}")
```

Figure 29 feature selection

```
Correlation between total_deaths and total_deaths: 1.0
Correlation between total_deaths and population: 0.30719110047649373
Correlation between total_deaths and total_cases: 0.9045693426476185
Correlation between total_deaths and new_cases: 0.6196742829602869
Correlation between total_deaths and new_deaths: 0.5977172297419507
Correlation between total_deaths and total_vaccinations: 0.12103421040283768
```

Figure 30 Feature variable

Secondly, James and Witten et al. (2013) said that the regression algorithm is an algorithm for predicting continuous variables. Basically, it tries to find the relationship between one variable and one or more other variables. We can use this algorithm to explore the relationship between the predicted total number of deaths and other variables. John et al. (2001) concluded in regression algorithms

that continuous-valued outputs can be predicted and trained the model using known input features and corresponding output labels to learn the relationship between features and outputs. Set the number of vaccinations as a trait and the number of deaths as input to explore and predict the relationship between them, or set other variables as input to predict the relationship between them.

As shown in Figure 31.

```
from pyspark.sql import SparkSession
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.regression import LinearRegression
import pandas as pd

selected_columns = ["total_deaths", "population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
df_cleaned = df_cleaned.select(selected_columns)

feature_columns = ["population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
vector_assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")
df_assembled = vector_assembler.transform(df_cleaned)

df_final = df_assembled.select("total_deaths", "features")

train_data, test_data = df_final.randomSplit([0.8, 0.2], seed=123)

lr = LinearRegression(featuresCol="features", labelCol="total_deaths")

lr_model = lr.fit(train_data)

coefficients = lr_model.coefficients
intercept = lr_model.intercept

print("Coefficients: " + str(coefficients))
print("Intercept: " + str(intercept))

predictions = lr_model.transform(test_data)

predictions.select("total_deaths", "prediction").show()
```

Figure 31 Regression algorithm

In the classification algorithm, Bishop(2006) concluded, a classification algorithm is an algorithm used to predict discrete variables, it is the process of assigning a given data point to a predefined class or category. Classification algorithms are used to assign input data to predefined categories or labels, and the outputs are

discrete categories. Models classify by learned relationships between features and categories in the training data. It provides flexibility in handling discrete features and handling missing values, and is used in classification and regression analyses. Commonly used algorithms include logistic regression, decision trees, random forests, etc. The goal of this kind of mining is numerical output, which is also suitable for the goal. As shown in Figure 32.

```
from pyspark.sql import SparkSession
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.classification import DecisionTreeClassifier
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from sklearn.metrics import accuracy_score

selected_columns = ["total_deaths", "population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
df_cleaned = df_cleaned.select(selected_columns)

feature_columns = ["population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
vector_assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")
output = vector_assembler.transform(df_cleaned)

output = output.withColumn("label", (output["total_deaths"] > 1000).cast("double"))
final_data = output.select("features", "label")

train_data, test_data = final_data.randomSplit([0.85, 0.15], seed=123)

dt_classifier = DecisionTreeClassifier(labelCol="label", featuresCol="features")

dtModel_classifier = dt_classifier.fit(train_data)

predictions = dtModel_classifier.transform(test_data)

evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")
accuracy = evaluator.evaluate(predictions)
print("Accuracy: {}".format(accuracy))

feature_importances = dtModel_classifier.featureImportances
for feature, importance in zip(feature_columns, feature_importances):
    print("{} Importance: {}".format(feature, importance))

y_true = predictions.select("label").rdd.map(lambda x: x[0])
y_pred = predictions.select("prediction").rdd.map(lambda x: x[0])
accuracy_sklearn = accuracy_score(y_true.collect(), y_pred.collect())
print("Accuracy (Scikit-learn): {}".format(accuracy_sklearn))
```

Figure 32 Decision Tree algorithm

In the clustering algorithm, it is an algorithm that groups data points where points within groups are more similar than points between groups. Cluster analysis algorithms group input data into similar clusters, where data within each cluster is more similar and data between different clusters is less similar. This is an unsupervised learning method. It groups data based on similarities. Commonly

used algorithms include K-means clustering, hierarchical clustering, etc. Anomaly detection algorithms are used to identify outlier data points that are significantly different from other data points and can be supervised or unsupervised based on the presence of labeled outlier data. Correlation analysis algorithms are used to discover correlations between items in a data set and find common sets of items. As shown in Figure 33.

```
from pyspark.sql import SparkSession
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.clustering import KMeans

selected_columns = ["total_deaths", "population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
df_cleaned = df_cleaned.select(selected_columns)

feature_columns = ["population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
vector_assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")
output = vector_assembler.transform(df_cleaned)

final_data = output.select("features")

# 使用 K-Means 聚类算法
kmeans = KMeans().setK(3)
model = kmeans.fit(final_data)

predictions = model.transform(final_data)

predictions.show()

|
evaluator = ClusteringEvaluator()
silhouette_score = evaluator.evaluate(predictions)
print("Silhouette Score: ", silhouette_score)
```

Figure 33 Clustering algorithm

6.2 Selection of data-mining algorithm

Based on the discussion in 6.1, in order to better explore the impact of predicting vaccine deaths and number of vaccinations as well as other variables, we chose the decision tree algorithm. Total_deaths are classified according to the learned relationship between features and categories in the training data (low, medium, high), and other variables are used as features to predict these categories. This approach helps us understand how various features relate to

different levels of total_deaths. By looking at the feature importance, or coefficient, of a classification algorithm, you can derive the correlation between the feature and the target variable. The decision tree algorithm provides high interpretability through a clear tree structure, allowing us to intuitively understand how features affect the target variable. It can provide the importance score of each feature, which helps us understand which features are more important for predicting total_deaths. It can capture non-linear relationships between features and target variables, which is useful for exploring patterns in complex data sets.

6.3 Build/Select appropriate model(s) and choose relevant parameter(s)

First select specific columns to generate a new DataFrame, use VectorAssembler to combine multiple feature columns into one feature vector column, and call the transform method to combine the feature columns into one feature vector column "features". Add a new column "label" via the withColumn method. Determine whether the value of the "total_deaths" column is greater than 1000. If so, return True (converted to 1.0), otherwise return False (converted to 0.0). The "label" column will be used as the target or response variable for the machine learning model. Then select features and target variables. We need to create a partition to divide the dataset into different parts for different purposes such as training, validation and testing. Set test size to 0.15, use 15% of the data as the test set, and the remaining 85% as the training set to train the parameters and weights of the model. The model learns patterns and relationships in the

data on the training set so that it can make accurate predictions or classifications. Set 15% of the test set for final evaluation of model performance. The test set was chosen to be 15% due to the relatively large, diverse, and representative nature of the dataset. After model development and parameter tuning are complete, use the test set to check how the model performs on unseen data. As shown in Figure 34.

```
selected_columns = ["total_deaths", "population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
df_cleaned = df_cleaned.select(selected_columns)

feature_columns = ["population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
vector_assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")
output = vector_assembler.transform(df_cleaned)

output = output.withColumn("label", (output["total_deaths"] > 1000).cast("double"))
final_data = output.select("features", "label")
train_data, test_data = final_data.randomSplit([0.85, 0.15], seed=123)
```

Figure 34 Partition parameter adjustment

Secondly, select classification death_class as the feature variable and select other variables as input, such as total_vaccinations, total_cases, etc., to ensure that the model is evaluated on different data subsets; create a decision tree classifier model, implement the decision tree algorithm, and use the fitting method Train the model. Create a DecisionTreeClassifier object and specify the names of the label column and feature column. Call the fit method to train the model using the training data set. Call the transform method to use the trained model to predict the test data set. The prediction results are stored in the predictions DataFrame, as shown in Figure 35. And calculate the correct accuracy and precision, that is, the proportion of correctly classified samples to the total samples and the proportion of samples that are truly positive among all samples

labeled as positive. Create a MulticlassClassificationEvaluator object for evaluating the model's performance. Call the evaluate method to calculate the accuracy of the model, as shown in Figure 36. The decision tree algorithm is shown in Figure 37.

```
train_data, test_data = final_data.randomSplit([0.85, 0.15], seed=123)

dt_classifier = DecisionTreeClassifier(labelCol="label", featuresCol="features")

dtModel_classifier = dt_classifier.fit(train_data)

predictions = dtModel_classifier.transform(test_data)
```

Figure 35 Model training and prediction

```
evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")
accuracy = evaluator.evaluate(predictions)
print("Accuracy: {}".format(accuracy))

feature_importances = dtModel_classifier.featureImportances
for feature, importance in zip(feature_columns, feature_importances):
    print("{} Importance: {}".format(feature, importance))
```

Figure 36 Model evaluation

```
from pyspark.sql import SparkSession
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.classification import DecisionTreeClassifier
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from sklearn.metrics import accuracy_score

selected_columns = ["total_deaths", "population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
df_cleaned = df_cleaned.select(selected_columns)

feature_columns = ["population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
vector_assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")
output = vector_assembler.transform(df_cleaned)

output = output.withColumn("label", (output["total_deaths"] > 1000).cast("double"))
final_data = output.select("features", "label")

train_data, test_data = final_data.randomSplit([0.85, 0.15], seed=123)

dt_classifier = DecisionTreeClassifier(labelCol="label", featuresCol="features")

dtModel_classifier = dt_classifier.fit(train_data)

predictions = dtModel_classifier.transform(test_data)

evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")
accuracy = evaluator.evaluate(predictions)
print("Accuracy: {}".format(accuracy))

feature_importances = dtModel_classifier.featureImportances
for feature, importance in zip(feature_columns, feature_importances):
    print("{} Importance: {}".format(feature, importance))
```

Figure 37 Regression Algorithm

7. Data Mining

7.1 Create and justify test designs

As mentioned in 6.3, create a logical test selection partition, divide the data set into training set, test set and evaluate the performance of the model. Set `test_size` to 0.15, use 15% of the data as the test set, and the remaining 85% as the training set to train the parameters and weights of the model. The model learns patterns and relationships in the data on the training set so that it can make accurate predictions or classifications. Set 15% of the test set for final evaluation of model performance. Since the data set is relatively large, diverse, and representative, the test set is selected to be 15%, which ensures that there is enough data to train the model, while leaving some data to verify the performance of the model. As shown in Figure 36. In the subsequent model evaluation, the accuracy reached 98%, indicating that the design of the training and test sets is more reasonable.

```
train_data, test_data = final_data.randomSplit([0.85, 0.15], seed=123)
```

Figure 36 Training and testing design

7.2 Conduct data mining

To explore and predict relationships between variables such as deaths and vaccinations, first define a list of column names 'selected_columns' containing the column names to be selected from the original DataFrame. Use the select method to create a new DataFrame 'df_cleaned' containing only the columns specified in

'selected_columns' ("total_deaths", "population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"). Second, define a list of feature column names 'feature_columns'. Create a 'VectorAssembler' object specifying the names of the predictor input and output columns. Call the 'transform' method to combine the feature columns into a feature vector column "features". Add a new column "label" via the 'withColumn' method. Determine whether the value of the "total_deaths" column is greater than 1000. If so, return True (converted to 1.0), otherwise return False (converted to 0.0). Use the 'select' method to select the "features" and "label" columns as input to the machine learning model. Use the 'randomSplit' method to randomly split the dataset into training and test datasets. Then select the partition for data mining training method to create the decision tree 'DecisionTreeClassifier' object model. Select 'DecisionTreeClassifier()', which is a class in the 'scikit-learn' library. It is a class used to implement the decision tree classification algorithm. It divides the samples into different classes based on different values of the input features, and then calls the 'fit' method to train the model with the training data set. As shown in Figure 37. The trained model is shown in Figure 38. Observe the model's accuracy and results by printing feature importance and accuracy for each feature. As shown in Figure 39.

```

selected_columns = ["total_deaths", "population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
df_cleaned = df_cleaned.select(selected_columns)

feature_columns = ["population", "total_cases", "new_cases", "new_deaths", "total_vaccinations"]
vector_assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")
output = vector_assembler.transform(df_cleaned)

output = output.withColumn("label", (output["total_deaths"] > 1000).cast("double"))

final_data = output.select("features", "label")

train_data, test_data = final_data.randomSplit([0.85, 0.15], seed=123)

dt_classifier = DecisionTreeClassifier(labelCol="label", featuresCol="features")

dtModel_classifier = dt_classifier.fit(train_data)

predictions = dtModel_classifier.transform(test_data)

evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")
accuracy = evaluator.evaluate(predictions)

```

Figure 37 Field setting of decision tree

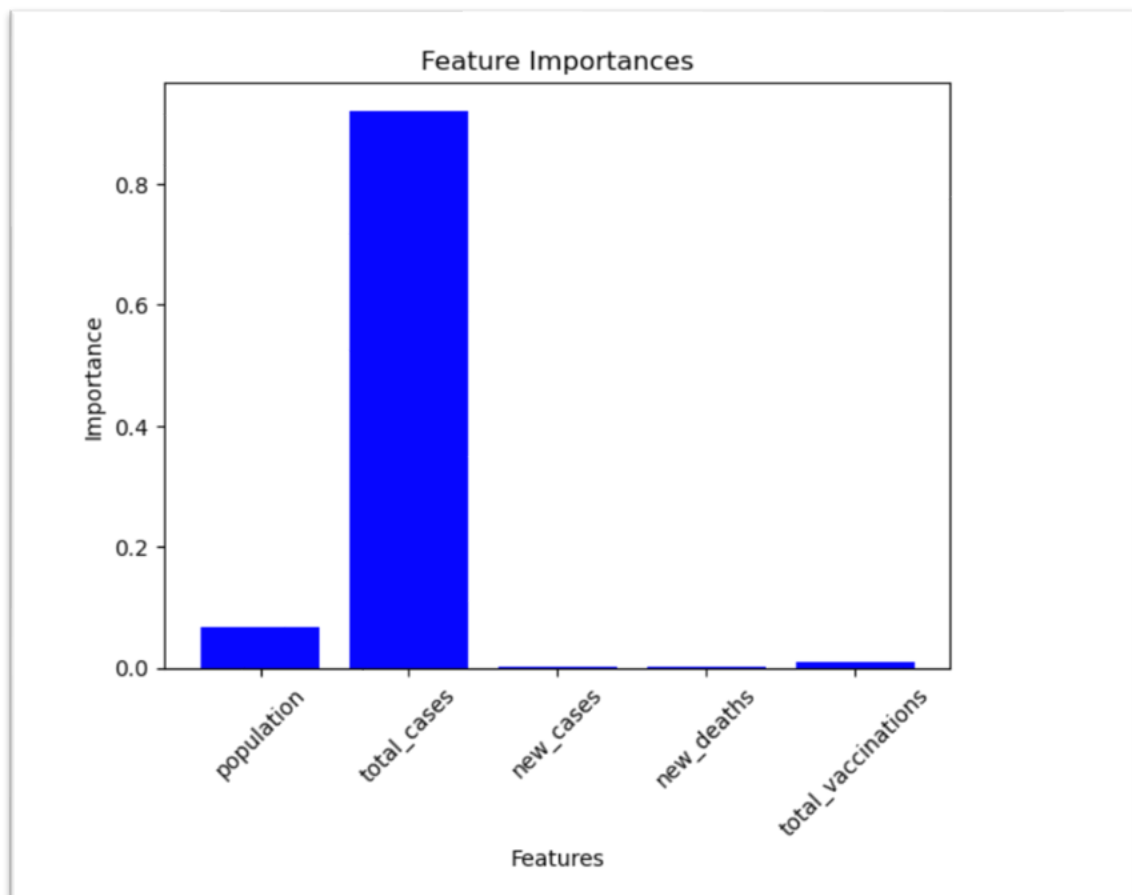


Figure 38 model of decision tree

```
Accuracy: 0.9841797654567421
population Importance: 0.06802314337624597
total_cases Importance: 0.9219125368935314
new_cases Importance: 0.0005872076432077867
new_deaths Importance: 0.0005837687369369529
total_vaccinations Importance: 0.008893343350077782
```

Figure 39 Importance and prediction

7.3 Search for patterns

According to the results in Figure 40, it can be seen that "total_cases" has the highest importance score of about 0.92, indicating that it contributes the most to explaining the "total_deaths" variable. showed that this was a very strong predictive factor. Secondly, the importance coefficient of "population" is 0.07, which is highly predictive. The significance of "total_vaccinations" is 0.008, indicating that there is a weak relationship and that in cases of high mortality, people may need to be vaccinated to reduce death. "new_cases" has an importance of 0.0005 and "new_deaths" has an importance of 0.005. Their prediction-test relationships are relatively weak. The high correlation in new deaths may indicate some correlation between epidemic deaths and case numbers. This could mean there is some correlation between the total number of cases and the number of deaths. Related. The lower significance of "total_cases" and "total_vaccinations" may indicate a weaker association between case numbers and vaccinations.

```
population Importance: 0.06802314337624597
total_cases Importance: 0.9219125368935314
new_cases Importance: 0.0005872076432077867
new_deaths Importance: 0.0005837687369369529
total_vaccinations Importance: 0.008893343350077782
```

Figure 40 Feature Importance

Based on the results in Figure 41, use the `accuracy_score` function to calculate the accuracy of the model. Accuracy is the proportion of correctly predicted observations. It is the proportion of correctly predicted samples to all samples. The accuracy result is 0.984, which means that the prediction accuracy of the model is very high.

```
Accuracy: 0.9841797654567421
```

Figure 41 Accuracy of the model

By calculating the confusion matrix, it is used to describe the performance of the classification model, showing information such as true and false positives for each category. The rows of the matrix represent the true categories and the columns represent the predicted categories. According to the results, In thresholds greater than 1000, 97632 class samples are correctly classified, in thresholds less than 1000,25117 class samples are correctly classified, and other off-diagonal numbers indicate misclassifications.

```
Confusion Matrix:  
[[ 97632      0]  
 [      0 205117]]
```

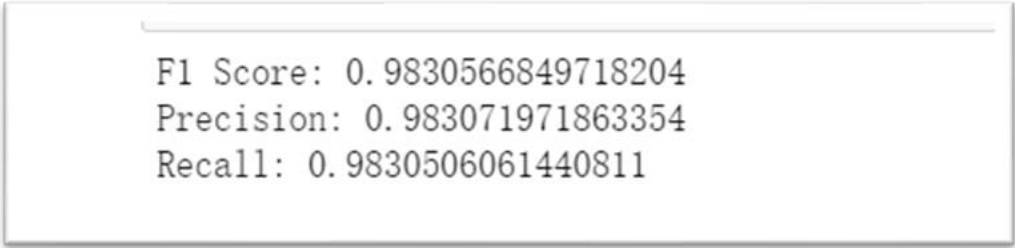
Figure 42 Analysis of Confusion Matrix

Use the `classification_report` function to print a classification report, including Precision, Recall, and F1-Score.

Precision: It is the proportion of correctly predicted positive samples to all predicted positive samples. In the results, The precision is also 0.983, which means the model is very accurate in predicting the positive class.

Recall: It is the proportion of correctly predicted positive samples to all actual positive samples. The recall rate is also 0.98, which means that the model is able to accurately identify most of the positive samples.

F1-Score: It is the harmonic mean of precision and recall, and its value is between precision and recall. It provides a unified standard for evaluating models. Through F1-score, precision and recall can be taken into account at the same time. The F1 score is 0.98, which means the model performs very well and can classify the samples almost perfectly. So the model has a high F1 score, precision, and recall, which usually means that the model performs very well and is able to accurately predict both positive and negative class samples.



```
F1 Score: 0.9830566849718204  
Precision: 0.983071971863354  
Recall: 0.9830506061440811
```

Figure 43 Analysis of the model

8. Interpretation

8.1 Study and discuss the mined patterns

In decision tree classification algorithm, it is a supervised learning algorithm used to learn decision rules from data features to predict the value of a target variable given an input data point. Decision trees divide data by selecting optimal features. Decision trees split the data recursively starting from the root node until certain stopping conditions are met (such as reaching a maximum depth or the number of samples at a node is less than a predefined threshold). The mining data pattern of the decision tree algorithm is as follows: Classification rules, that is, each path from the root to the leaves of the decision tree can be regarded as a classification rule, representing the relationship between features and target variables. Feature importance, or feature importance, represents the importance of each feature in classification decisions. Feature importance can be used to identify and understand the relationship between features and target variables. Analyze the relationship between different predictor variables (such as the total number of cases, etc.) and the target variable. Detect and analyze outliers before they impact model performance and results. Use the magnitude of performance

metrics such as importance to understand how different predictors affect the model.

The trained model and output results are shown in Figure 44. It can be seen that "total_cases" has the highest importance score of 0.92, indicating that it has the greatest impact on the "total_deaths" variable. The importance of "Population" is also relatively high at 0.07, and the relationship between them is equally large. The importance score of "total_vaccinations" is 0.008, and its prediction-test relationship is relatively weak, indicating that in cases of high mortality, people may need to be vaccinated to reduce death. The high importance of "total_cases" may indicate a significant correlation between the number of deaths in an outbreak and the total number of cases, possibly because widespread disease causes more deaths. "Population" also has a higher importance, which may mean there is some correlation between the number of deaths and the number of population density. The lower significance of "new_cases", "total_vaccinations" and "new_deaths" may indicate a weaker association between the number of new cases and vaccinations and new deaths.

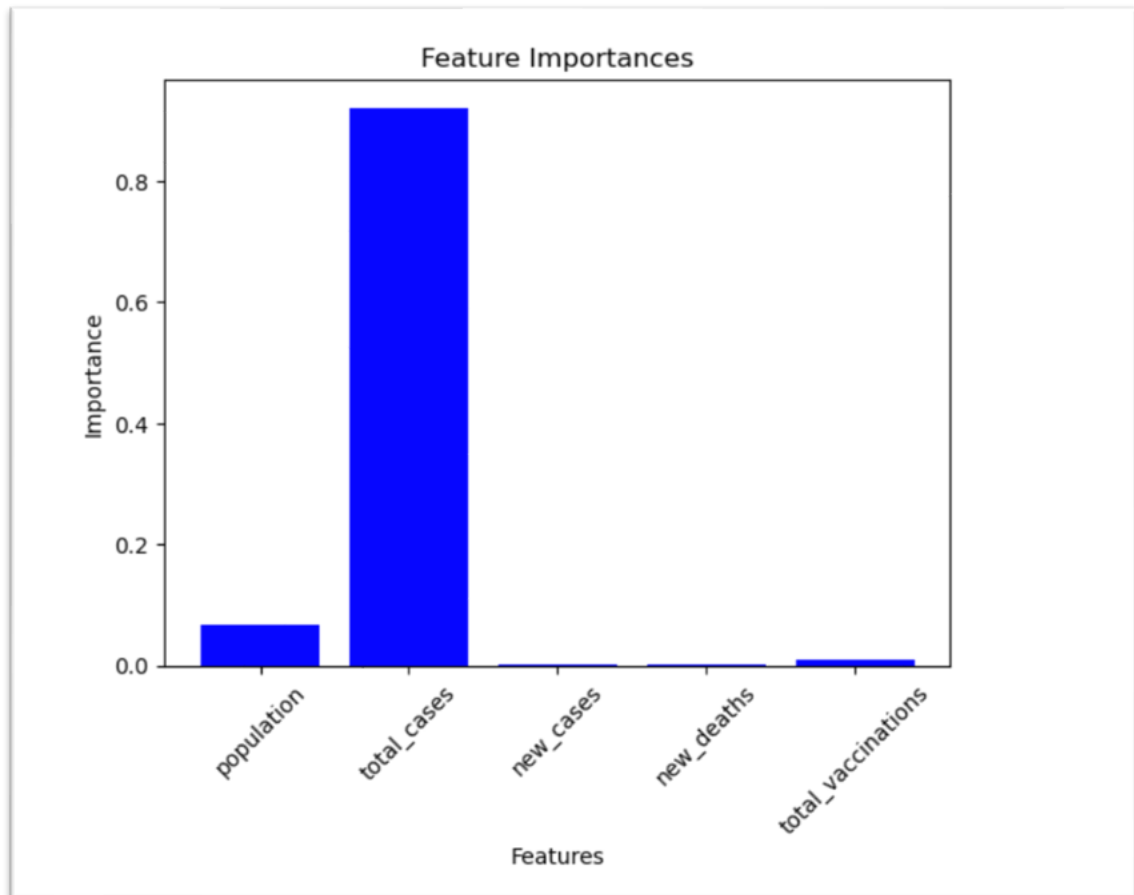


Figure 44 Exported model

8.2 Visualize the data, results, models, and patterns

The trained model is obtained by running the decision trees algorithm, as shown in Figures 45 and 46. You can see the relationship between the total number of deaths and the number of total cases and other variables.

```

DecisionTreeClassificationModel: uid=DecisionTreeClassifier_0e73e3e48d16,
If (feature 1 <= 78343.0)
  If (feature 1 <= 50582.0)
    If (feature 4 <= 8.5)
      Predict: 0.0
    Else (feature 4 > 8.5)
      If (feature 1 <= 27860.0)
        Predict: 0.0
      Else (feature 1 > 27860.0)
        If (feature 0 <= 5730703.5)
          Predict: 0.0
        Else (feature 0 > 5730703.5)
          Predict: 1.0
    Else (feature 1 > 50582.0)
      If (feature 0 <= 1573576.5)
        Predict: 0.0
      Else (feature 0 > 1573576.5)
        If (feature 0 <= 3.7021662E7)
          If (feature 4 <= 0.5)
            Predict: 0.0
          Else (feature 4 > 0.5)
            Predict: 1.0
        Else (feature 0 > 3.7021662E7)
          Predict: 1.0
  Else (feature 1 > 78343.0)
    If (feature 0 <= 1008429.0)
      If (feature 1 <= 434239.5)
        Predict: 0.0
      Else (feature 1 > 434239.5)
        Predict: 1.0
    Else (feature 0 > 1008429.0)
      If (feature 1 <= 100553.0)
        If (feature 0 <= 3.7021662E7)
          If (feature 0 <= 2.8037545E7)
            Predict: 1.0
          Else (feature 0 > 2.8037545E7)
            Predict: 0.0
        Else (feature 0 > 3.7021662E7)
          Predict: 1.0
      Else (feature 1 > 100553.0)
        If (feature 0 <= 1573576.5)
          If (feature 1 <= 251248.0)
            Predict: 0.0
          Else (feature 1 > 251248.0)
            Predict: 1.0
        Else (feature 0 > 1573576.5)
          Predict: 1.0

```

Figure 45 Training of the model

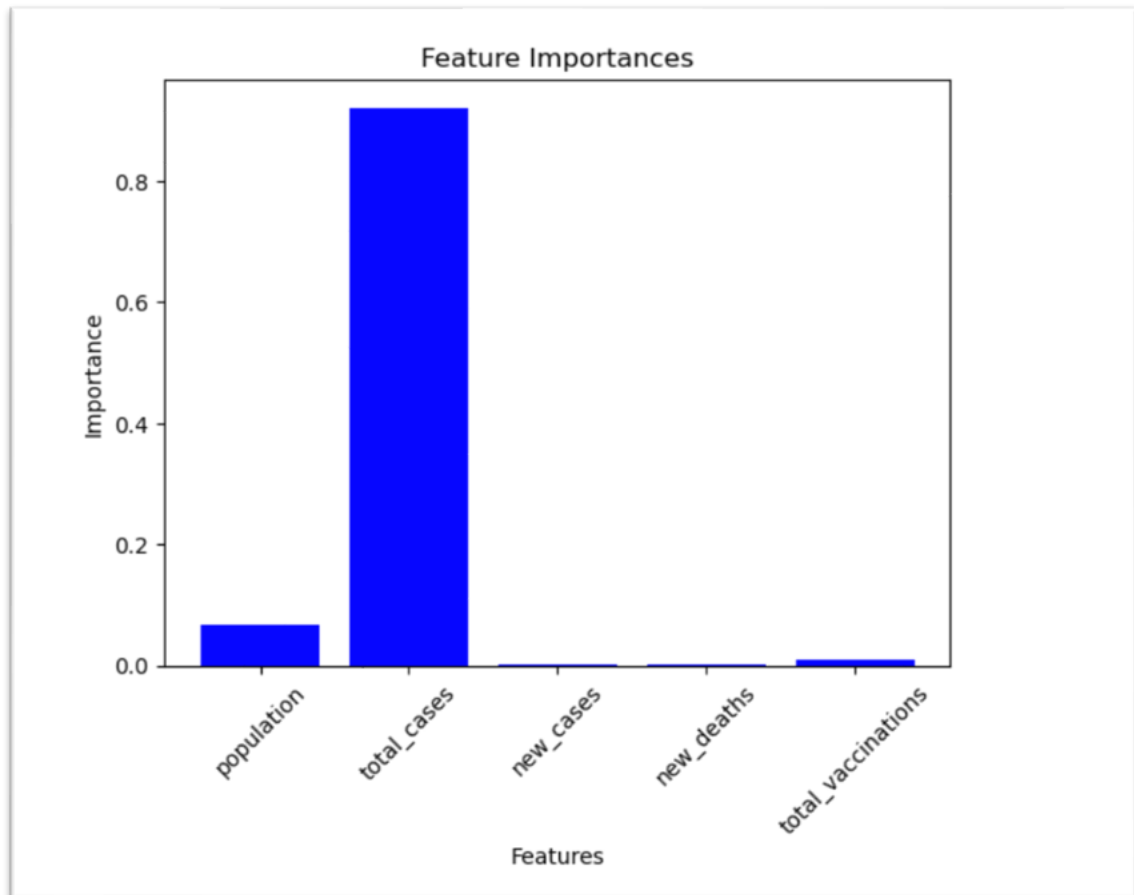


Figure 46 Feature Importance of the model

Through analysis, perform various statistical analysis and data mining tasks and visualize confusion matrices to make them easier to interpret and understand. As shown in Figure 47.

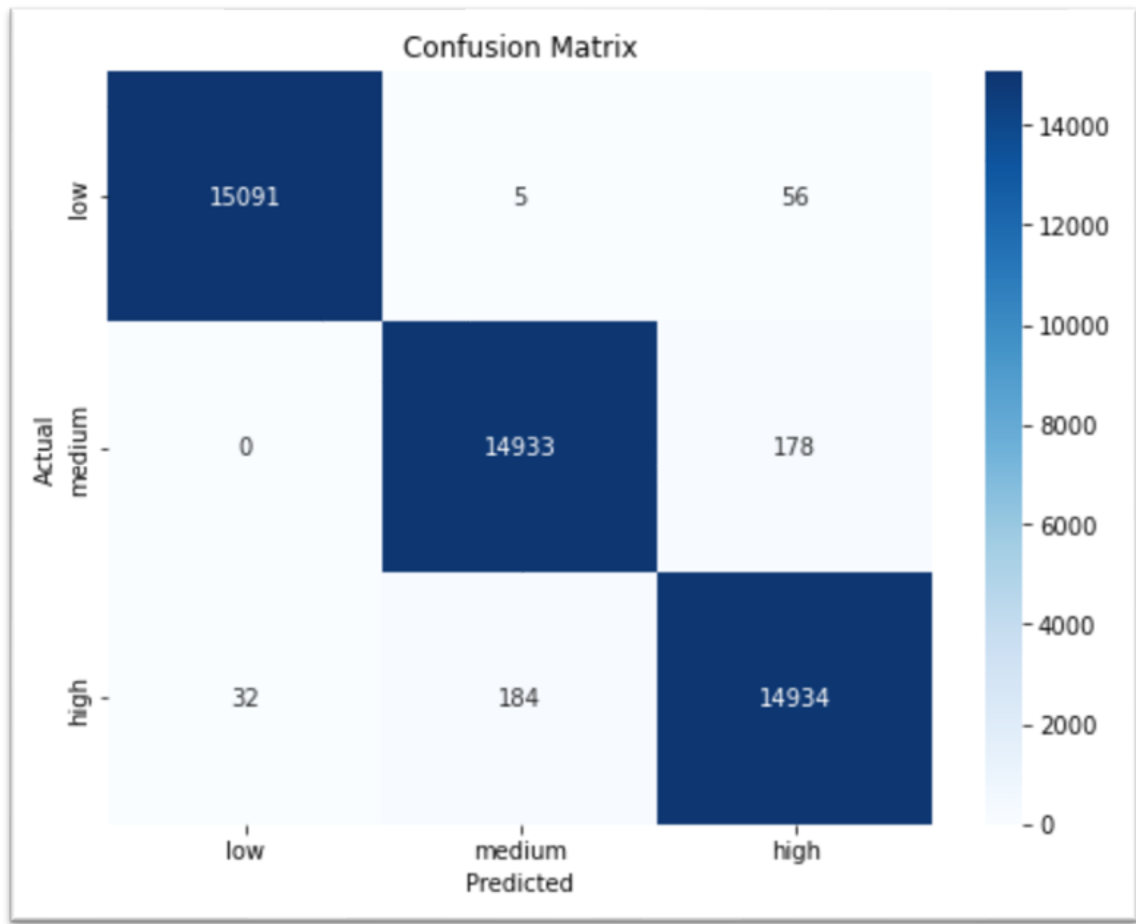


Figure 47 Confusion Matrix of the model

By setting parallel coordinates to visualize the relationship between variables, you can observe the relationship between various numerical variables and see patterns, trends, and correlations between them. As shown in Figure 48.

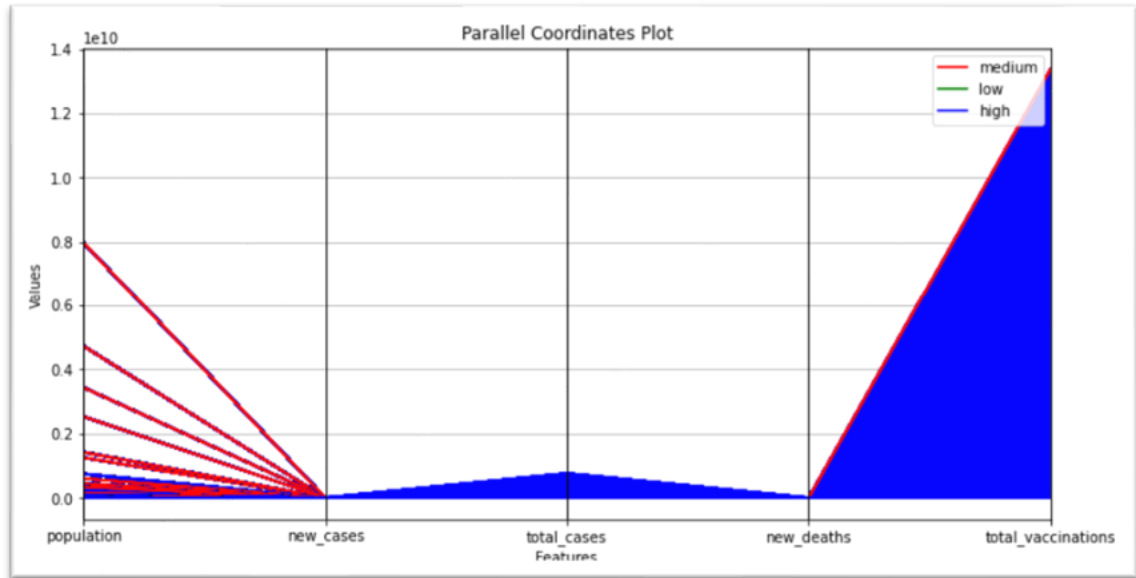


Figure 48 Parallel of the model

Use the "dtModel_classifier.toDebugString" text to visualize the structure and decision paths of the decision tree. This allows you to intuitively understand the inner workings of the model, including the judgment conditions of each node, the depth of the node, and other information. As shown in Figure 49.

```

DecisionTreeClassificationModel: uid=DecisionTreeClassifier_0e73e3e48d16, depth=5, numNodes=31, numClasses=2, numFeatures=5
If (feature 1 <= 78343.0)
  If (feature 1 <= 50582.0)
    If (feature 4 <= 8.5)
      Predict: 0.0
    Else (feature 4 > 8.5)
      If (feature 1 <= 27860.0)
        Predict: 0.0
      Else (feature 1 > 27860.0)
        If (feature 0 <= 5730703.5)
          Predict: 0.0
        Else (feature 0 > 5730703.5)
          Predict: 1.0
  Else (feature 1 > 50582.0)
    If (feature 0 <= 1573576.5)
      Predict: 0.0
    Else (feature 0 > 1573576.5)
      If (feature 0 <= 3.7021662E7)
        If (feature 4 <= 0.5)
          Predict: 0.0
        Else (feature 4 > 0.5)
          Predict: 1.0
      Else (feature 0 > 3.7021662E7)
        Predict: 1.0
  Else (feature 1 > 78343.0)
    If (feature 0 <= 1008429.0)
      If (feature 1 <= 434239.5)
        Predict: 0.0
      Else (feature 1 > 434239.5)
        Predict: 1.0
    Else (feature 0 > 1008429.0)
      If (feature 1 <= 100553.0)
        If (feature 0 <= 3.7021662E7)
          If (feature 0 <= 2.8037545E7)
            Predict: 1.0
          Else (feature 0 > 2.8037545E7)
            Predict: 0.0
        Else (feature 0 > 3.7021662E7)
          Predict: 1.0
      Else (feature 1 > 100553.0)
        If (feature 0 <= 1573576.5)
          If (feature 1 <= 251248.0)
            Predict: 0.0
          Else (feature 1 > 251248.0)
            Predict: 1.0
        Else (feature 0 > 1573576.5)
          Predict: 1.0

```

Figure 49 Decision tree visualization

8.3 Interpret the results, models, and patterns

As mentioned in 8.1, "total_cases" has the highest importance score of 0.92, indicating that it has the greatest impact on the "total_deaths" variable. The importance of "Population" is also relatively high at 0.07, and the relationship between them is equally large. The importance score of "total_vaccinations" is 0.008, and its prediction-test relationship is relatively weak, indicating that in cases of high mortality, people may need to be vaccinated to reduce death. The high importance of "total_cases" may indicate a significant correlation between the

number of deaths in an outbreak and the total number of cases, possibly because widespread disease causes more deaths. "Population" also has a higher importance, which may mean there is some correlation between the number of deaths and the number of population density. The lower significance of "new_cases", "total_vaccinations" and "new_deaths" may indicate a weaker association between the number of new cases and vaccinations and new deaths.

After model training, by observing the confusion matrix results, in thresholds greater than 1000, 97632 class samples are correctly classified, in thresholds less than 1000, 25117 class samples are correctly classified, and other off-diagonal numbers indicate misclassifications. Observe the F1-Score: it is the harmonic mean of precision and recall, and its value is between precision and recall. It provides a unified standard for evaluating models. Through the F1 score, both precision and recall can be considered, and the result is 0.98-1, indicating that the model is relatively accurate.

8.4 Assess and evaluate results, models, and patterns

First check the data quality, check for errors/missing values etc., by means of data cleaning etc. As shown in Figure 50.

```
population number of missing value: 0
total_cases number of missing value: 0
new_cases number of missing value: 0
total_deaths number of missing value: 0
new_deaths number of missing value: 0
total_vaccinations number of missing value: 0
```

Figure 50 Quality after handling missing values

For predictive models, Use the function 'metricName="accuracy"' in 'MulticlassClassificationEvaluator' to calculate the accuracy of the model. Accuracy is the proportion of observations correctly predicted. It is the ratio of correctly predicted samples to all samples. The accuracy is 0.984, which means the model's prediction accuracy is very high. As shown in Figure 51. By calculating and visualizing the confusion matrix, as shown in Figure 52, in thresholds greater than 1000, 97632 class samples are correctly classified, in thresholds less than 1000, 25117 class samples are correctly classified, and other off-diagonal numbers indicate misclassifications. Evaluate the model by calculating precision, recall, and F1 scores. Precision: It is the ratio of correctly predicted positive samples to all predicted positive samples. In the results, the accuracy is also 0.983, which means the model is very accurate in predicting the positive class. Recall: It is the ratio of correctly predicted positive samples to all actual positive samples. The recall rate is also 0.98, which means that the model is able to accurately identify most positive samples. F1-Score: It is the harmonic mean of precision and recall, and its value is between precision and recall. It provides a unified standard for evaluating models. Through the F1 score, both precision and recall can be considered. The result is 0.98-1, which means the model is more accurate. Overall, the model has better performance and higher accuracy. Overall, the model has better

performance and higher accuracy.

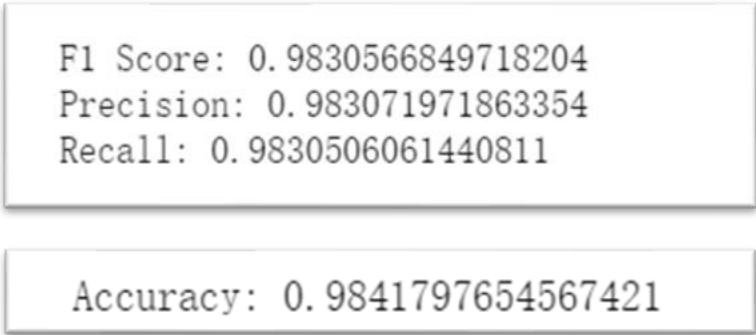


Figure 51 Model evaluation

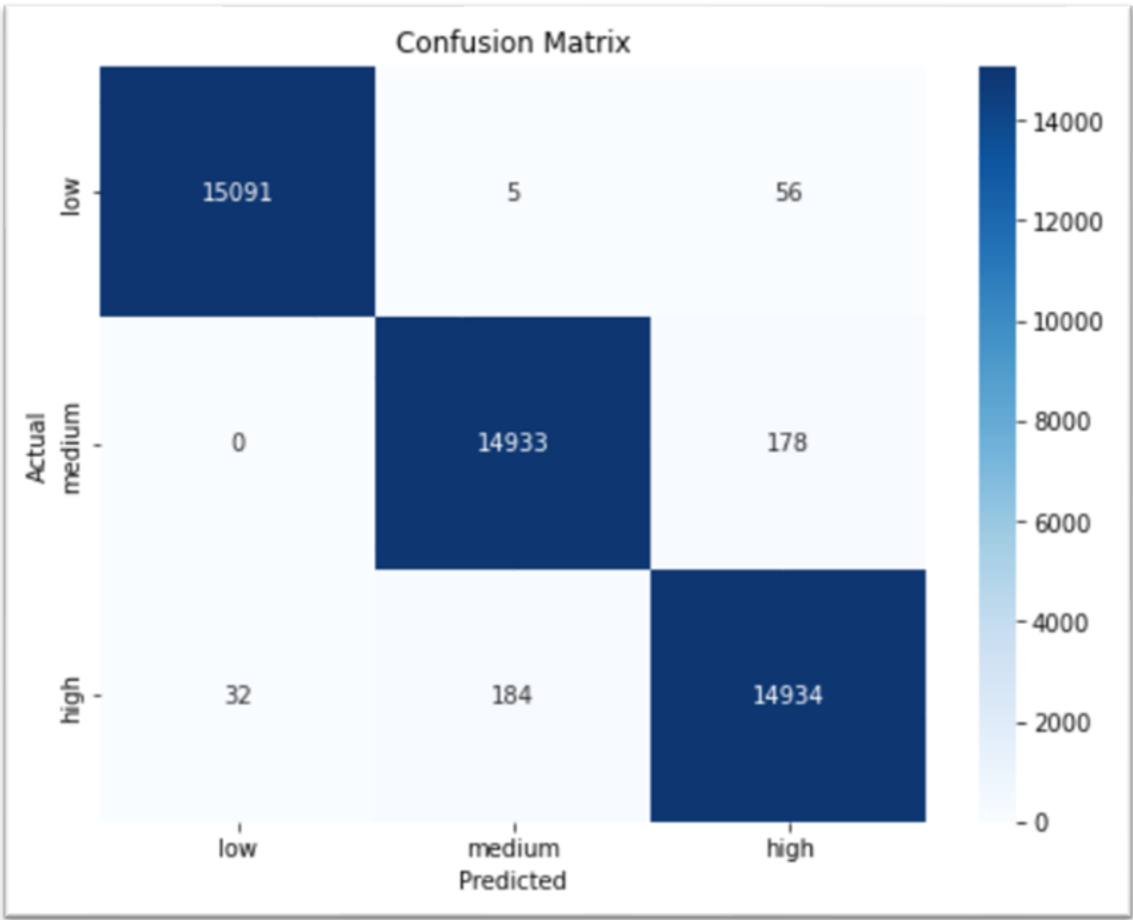


Figure 52 Confusion matrix visualization

8.5 Iterate prior

First select the data and perform data cleaning, handle missing values, etc. Then, the data of the table is integrated, using the merge method, as shown in Figure

53and 54.

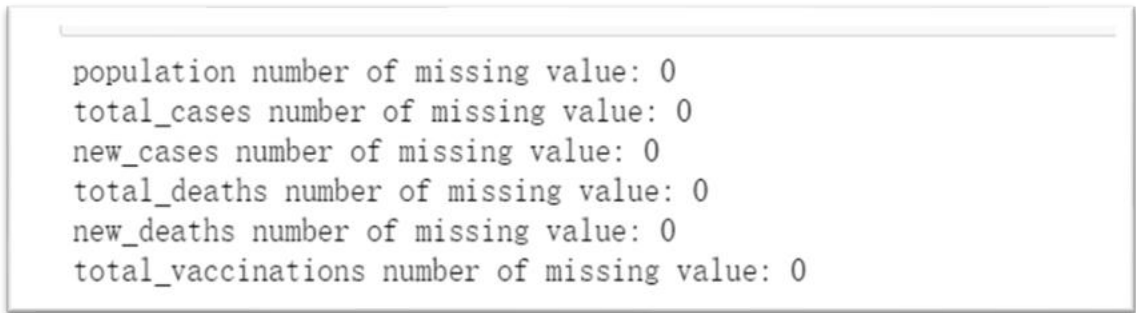


Figure 53 After cleaning data

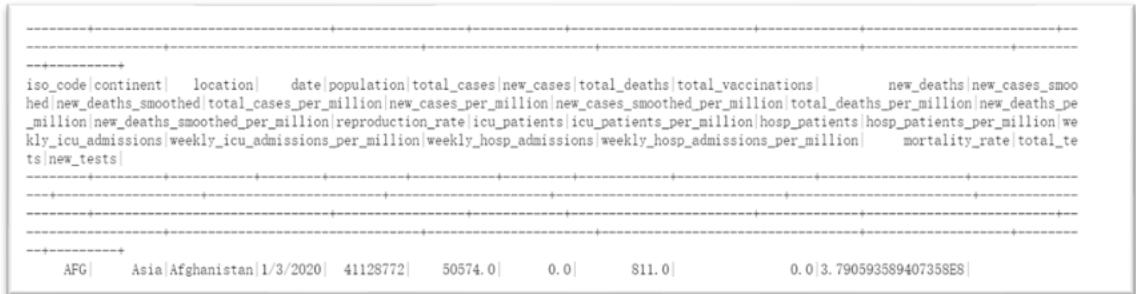


Figure 54 After integration data

Feature selection is chosen to reduce the complexity of the model, improve the performance and generalization ability of the model, and reduce the risk of overfitting, as shown in Figure 55, and then delete irrelevant columns and retain only the required columns to mitigate the bias of class imbalance, thereby reducing Most categories of impact. As shown in Figures 56.

```
for i, col in enumerate(numerical_cols):
    print(f"Correlation between total_deaths and {col}: {corr_matrix[i][numerical_cols.index('total_deaths')]}")
```

```
Correlation between total_deaths and total_deaths: 1.0
Correlation between total_deaths and population: 0.30719110047649373
Correlation between total_deaths and total_cases: 0.9045693426476185
Correlation between total_deaths and new_cases: 0.6196742829602869
Correlation between total_deaths and new_deaths: 0.5977172297419507
Correlation between total_deaths and total_vaccinations: 0.12103421040283768
```

Figure 55 model of feature selection

[illegible]

Figure 56 reducing model

To select a decision tree algorithm for data mining based on the task objectives, a partition needs to be established to divide the data set into different parts for different purposes such as training, validation, and testing. 85% of the training set is used to train the parameters and weights of the model. The model learns patterns and relationships in the data on the training set so that it can make accurate predictions or classifications. Set 15% of the test set for final evaluation of model performance. After model development and parameter

tuning are complete, use the test set to check the model's performance on unseen data. Then set up the decision tree algorithm to mine the trained model after multiple iterations, as shown in Figures 57 and 58. The conclusion is that the number of deaths is positively correlated with the total number of cases, and in areas with more deaths, the number of people who get sick is likely to be higher. Evaluate the fit of the model by observing Accuracy and precision, recall, f1-score, etc. in the model, as shown in Figure 59 and 60.

```
DecisionTreeClassificationModel: uid=DecisionTreeClassifier_0e73e3e48d16, depth=5, numNodes=31, numClasses=2, numFeatures=5
If (feature 1 <= 78343.0)
  If (feature 1 <= 50582.0)
    If (feature 4 <= 8.5)
      Predict: 0.0
    Else (feature 4 > 8.5)
      If (feature 1 <= 27860.0)
        Predict: 0.0
      Else (feature 1 > 27860.0)
        If (feature 0 <= 5730703.5)
          Predict: 0.0
        Else (feature 0 > 5730703.5)
          Predict: 1.0
    Else (feature 1 > 50582.0)
      If (feature 0 <= 1573576.5)
        Predict: 0.0
      Else (feature 0 > 1573576.5)
        If (feature 0 <= 3.7021662E7)
          If (feature 4 <= 0.5)
            Predict: 0.0
          Else (feature 4 > 0.5)
            Predict: 1.0
        Else (feature 0 > 3.7021662E7)
          Predict: 1.0
  Else (feature 1 > 78343.0)
    If (feature 0 <= 1008429.0)
      If (feature 1 <= 434239.5)
        Predict: 0.0
      Else (feature 1 > 434239.5)
        Predict: 1.0
    Else (feature 0 > 1008429.0)
      If (feature 1 <= 100553.0)
        If (feature 0 <= 3.7021662E7)
          If (feature 0 <= 2.8037545E7)
            Predict: 1.0
          Else (feature 0 > 2.8037545E7)
            Predict: 0.0
        Else (feature 0 > 3.7021662E7)
          Predict: 1.0
      Else (feature 1 > 100553.0)
        If (feature 0 <= 1573576.5)
          If (feature 1 <= 251248.0)
            Predict: 0.0
          Else (feature 1 > 251248.0)
            Predict: 1.0
        Else (feature 0 > 1573576.5)
          Predict: 1.0
```

Figure 57 Training model

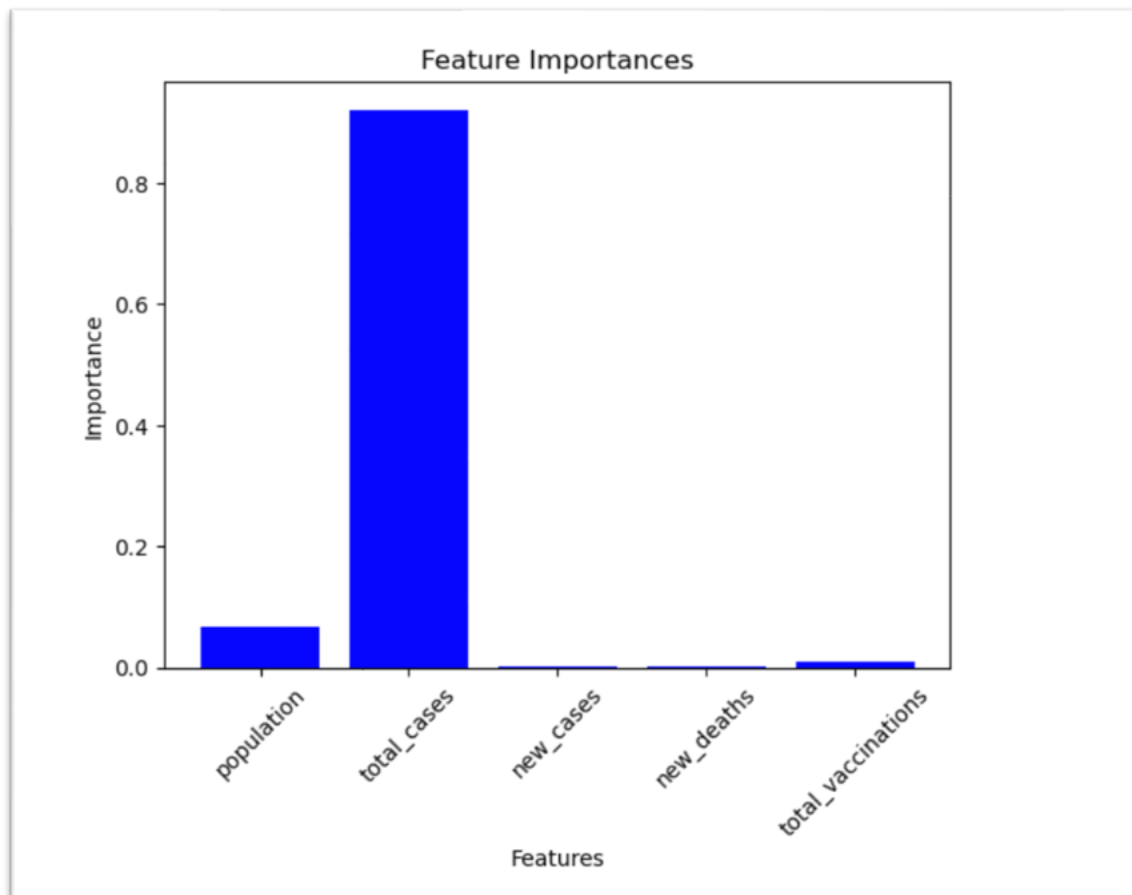


Figure 58 Model of decision tree

```
Accuracy: 0.9841797654567421
population Importance: 0.06802314337624597
total_cases Importance: 0.9219125368935314
new_cases Importance: 0.0005872076432077867
new_deaths Importance: 0.0005837687369369529
total_vaccinations Importance: 0.008893343350077782
```

```
F1 Score: 0.9830566849718204
Precision: 0.983071971863354
Recall: 0.9830506061440811
```

Figure 59 Model evaluation analysis

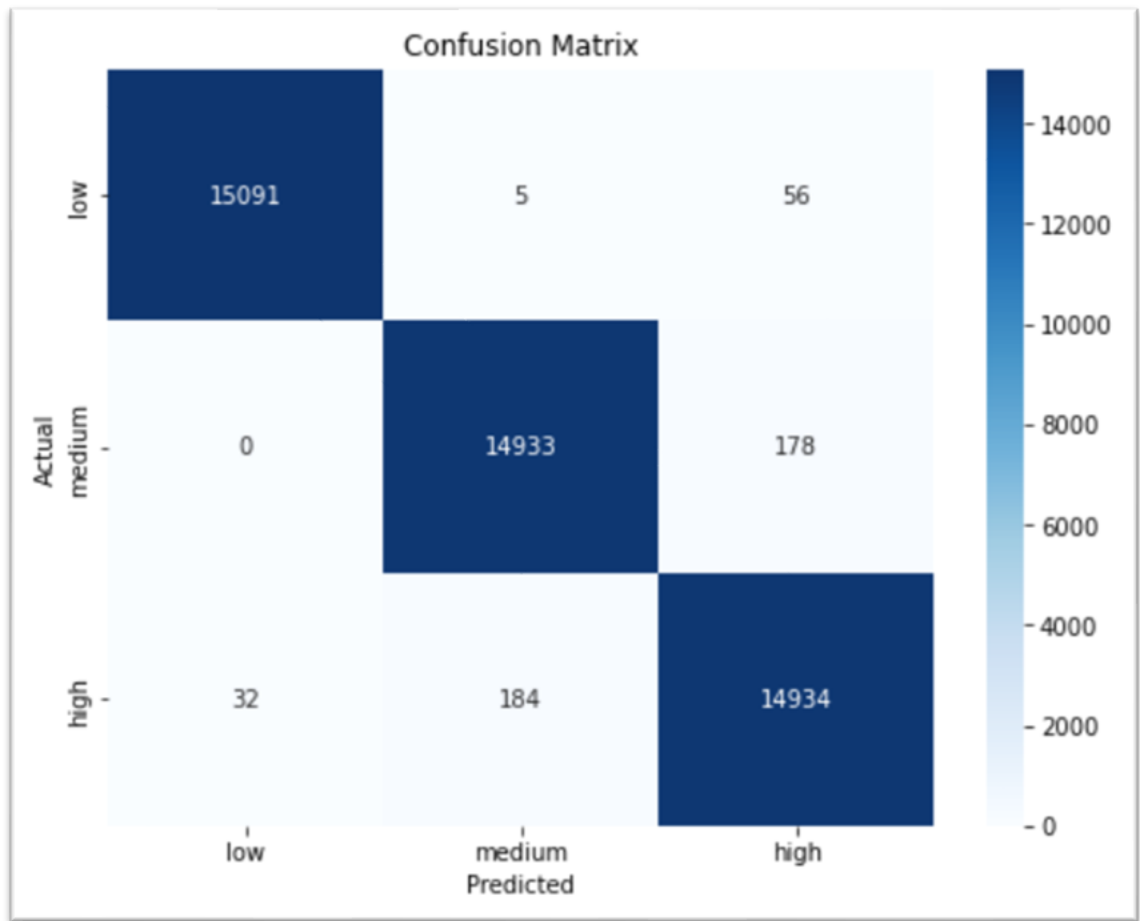


Figure 60 Model evaluation analysis

References

Tohid khan Bagani. (2023). Covid-19 deaths and vaccinations Dataset. <https://www.kaggle.com/datasets/tohidkhanbagani/covid-19-deaths-and-vaccinations-dataset>

Bartsch, S. M., O'Shea, K. J., Ferguson, M. C., et al. (2021). Vaccine Efficacy Needed for a COVID-19 Coronavirus Vaccine to Prevent or Stop an Epidemic as the Sole Intervention. *American Journal of Preventive Medicine*, 60(4), 490-502.

Viboud, C., Sun, K., Gaffey, R., et al. (2020). The RAPIDD Ebola forecasting challenge: synthesis and learning in public health responses. *Epidemics*, 30, 100378.

Douglas C. Montgomery, Elizabeth A. Peck, G. Geoffrey Vining.(2012). An Introduction to Linear Regression Analysis. <https://www.wiley.com/en-us/Introduction+to+Linear+Regression+Analysis%2C+5th+Edition-p-9781118471463>

John O. Rawlings, Sastry G. Pantula, David A. Dickey. (2001). Applied Regression Analysis: A Research Tool. <https://www.springer.com/gp/book/9780387984542>

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning. Springer.

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer. ISBN 0-387-31073-8.

Provost, F., & Fawcett, T. (2013). Data Science for Business: What You Need to Know about Data Mining and Data -Analytic Thinking. O'Reilly Media, Inc.

Xu, Rui, and Donald Wunsch, II. (2010). Clustering algorithms in biomedical research: a review. *IEEE Reviews in Biomedical Engineering* 3: 120-154.

I acknowledge that the submitted work is my own original work in accordance with the University of Auckland guidelines and policies on academic integrity and copyright. (See: <https://www.auckland.ac.nz/en/students/forms-policies-and-guidelines/student-policies-and-guidelines/academic-integrity-copyright.html>).

I also acknowledge that I have appropriate permission to use the data that I have utilised in this project. (For example, if the data belongs to an organisation and the data has not been published in the public domain then the data must be approved by the rights holder.) This includes permission to upload the data file to Canvas. The University of Auckland bears no responsibility for the student's misuse of data.