FINAL VERSION — COVID-19 DATA ANALYSIS REPORT

Title:COVID-19 Global Data Analysis and Visualization

Course: DataAnalysisAlgorithms

DatasetSource: OurWorldinData(OWID)

Tools: Python,PowerBI

TABLE OF CONTENTS

1. Introduction

The COVID-19 pandemic generated complex, large-scale, and highly skewed global data. This project aims to analyze COVID-19 country-level data to identify trends in infections, deaths, testing behavior, and policy responses.

The project combines:

- **Python-based data analysis** for statistical rigor
- **Power BI dashboards** for interactive visualization

The goal is to provide meaningful insights while applying robust data preprocessing and exploratory techniques.

2. Dataset Description

- **Source:** Our World in Data (OWID) COVID-19 Dataset
- **Granularity:** Daily, country-level time series
- **Time Period:** 2020 – 2024
- **Key Dimensions:**
  - Date
  - Country
  - Continent
- **Key Variables:**
  - New cases and deaths
  - Per-million indicators
  - Testing and positivity rate
  - Government stringency index

3. Tools & Technologies

Python Libraries

- pandas, numpy – data manipulation
- matplotlib, seaborn – visualization
- scipy – statistical analysis

Machine Learning

- scikit-learn (StandardScaler, PCA, Lasso, RFE)

Visualization

- Power BI – interactive dashboards

4. Project Workflow (Flowchart)

Raw COVID-19 Data

↓

Initial Data Inspection

↓

EDA (Before Cleaning)

↓

Feature Selection

↓

Data Cleaning

(Missing Values + Date Handling)

↓

Outlier Handling (IQR Clipping)

↓

Feature Engineering

↓

Statistical Analysis & PCA

↓

Power BI Visualization

5. Exploratory Data Analysis (EDA)

5.1 EDA Before Cleaning

EDA was performed to:

- Understand data distributions
- Identify missing values
- Detect extreme outliers
- Analyze categorical imbalance

Selected Code Example:

```
numeric_cols = df.select_dtypes(include="number").columns

df[numeric_cols].hist(figsize=(16,10))

plt.show()
```

Key Observations

- Most variables are **right-skewed**
- Testing-related variables contain **high missingness**
- Daily cases and deaths show **extreme peaks**

6. Data Cleaning & Preprocessing

6.1 Date Handling

Dates were converted to proper datetime format to enable time-series analysis.

```
df["date"] = pd.to_datetime(df["date"], errors="coerce")

df = df.dropna(subset=["date"])
```

6.2 Country-Level Filtering

Aggregate records (World, continents, income groups) were excluded.

```
df = df[df["continent"].notna()]
```

6.3 Missing Value Treatment

- **Numerical variables:** Median
- **Categorical variables:** Mode

This approach is robust against extreme values and skewed distributions.

7. Outlier Handling

7.1 IQR-Based Clipping

Outliers were handled using the Interquartile Range (IQR) method.

```
def clip_iqr(series):

    q1 = series.quantile(0.25)

    q3 = series.quantile(0.75)

    iqr = q3 - q1

    return series.clip(q1 - 1.5*iqr, q3 + 1.5*iqr)
```

Why Clipping?

- Preserves time-series continuity
- Avoids deleting valid epidemic peaks
- Reduces noise caused by reporting anomalies

Clipping was applied **only to daily and rate-based variables**, not cumulative totals.

8. Feature Engineering

Additional indicators were created:

- Case Fatality Rate (CFR)
- Growth Rate
- Active case proxies

These features enhance interpretability beyond raw case counts.

9. Statistical Analysis & PCA

9.1 Correlation Analysis

Correlation matrices were used to examine relationships between cases, deaths, testing, and policy measures.

9.2 Principal Component Analysis (PCA)

PCA was applied after standardization to:

- Reduce dimensionality
- Reveal latent patterns across countries

scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)

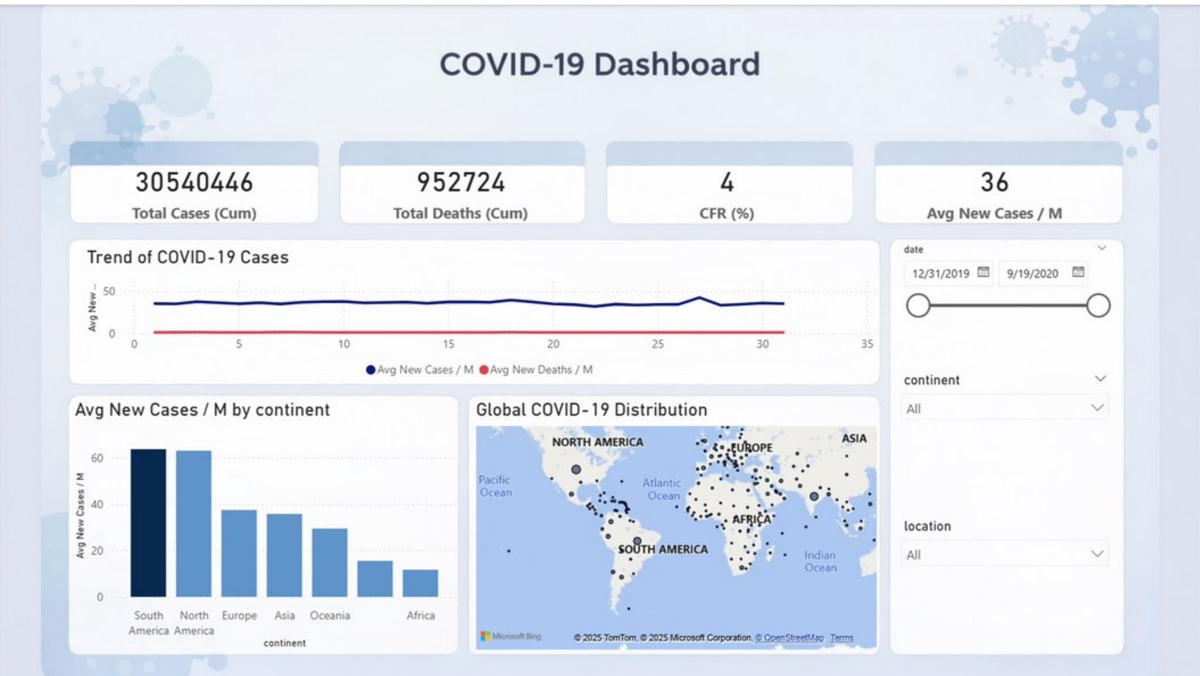
pca = PCA(n_components=2)

X_pca = pca.fit_transform(X_scaled)

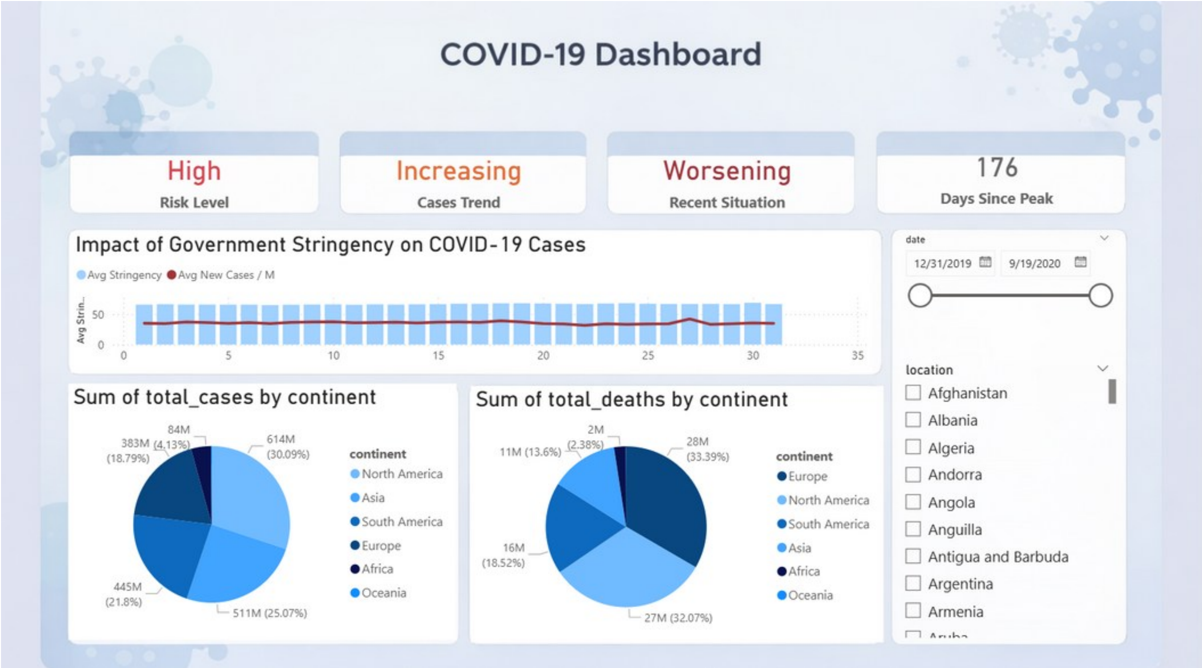
10. Power BI Dashboard

Dashboard Pages

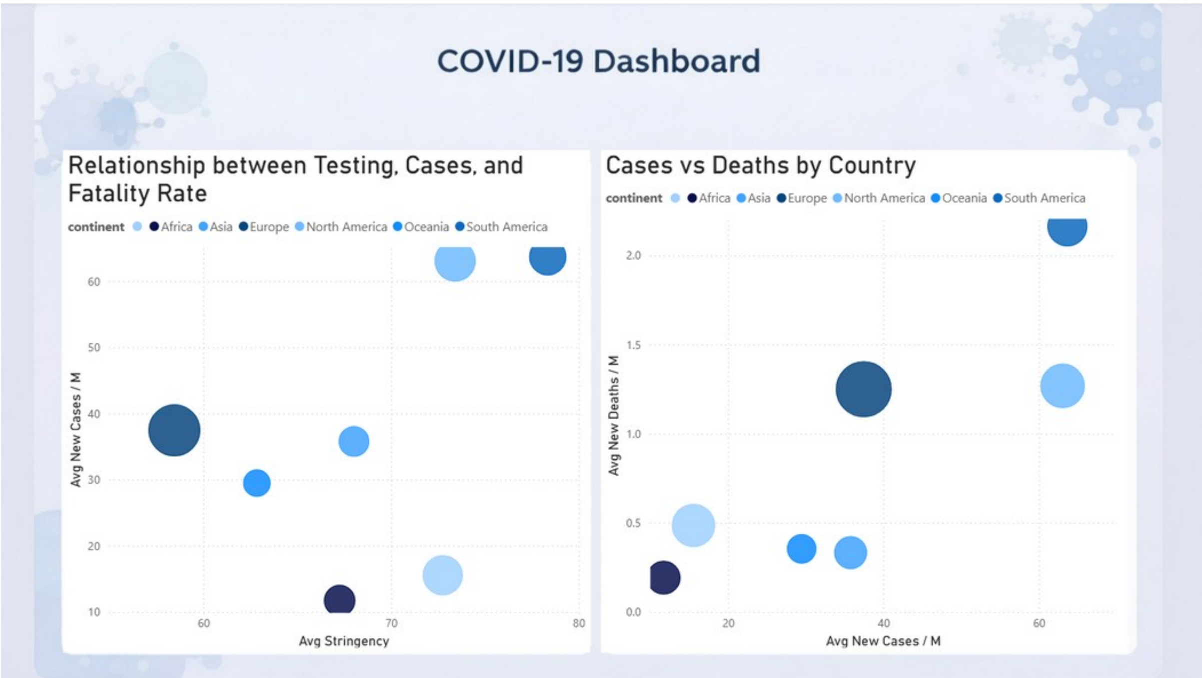
1. **Global Overview**



2.Country-Level Trends



3.Testing & Positivity Analysis



4.Ranking & Comparison Table

location	First continent	Avg New Cases / M	Avg Stringency	CFR (%)	Cases Trend	Total Cases (Cum)	Total Deaths (Cum)	Avg New Deaths / M
Vatican	Europe	1,648.13	72.76	0.00	Stable	12	0	0.00
Falkland Islands	South America	622.07	74.69	0.00	Increasing	13	0	0.00
Aruba	North America	395.21	57.13	0.60	Increasing	3460	23	2.51
Montserrat	North America	371.50	65.89	3.51	Stable	13	1	0.00
San Marino	Europe	242.09	70.59	7.30	Decreasing	723	42	11.05
Turks and Caicos Islands	North America	239.27	61.85	0.95	Decreasing	667	5	1.08
Qatar	Asia	215.47	76.15	0.14	Increasing	122917	209	0.37
Sint Maarten (Dutch part)	North America	209.15	72.76	5.77	Decreasing	574	20	6.19
Andorra	Europe	196.52	46.01	5.09	Increasing	1564	53	6.03
Bahrain	Asia	188.50	67.82	0.33	Increasing	63189	219	0.63

Each dashboard supports interactive filtering by:

- Date
- Country
- Continent

11. Key Insights

- COVID-19 indicators exhibit **heavy-tailed distributions**
- Testing intensity strongly affects reported case counts
- Government stringency shows varying effectiveness across regions
- PCA reveals clusters of countries with similar pandemic dynamics

12. Limitations

- Incomplete testing data for some countries
- Reporting inconsistencies across regions
- Aggregated indicators may hide sub-national variation

13. Conclusion

This project demonstrates how robust preprocessing, careful outlier handling, and combined statistical and visual analysis can extract meaningful insights from complex global health data.

14. References

- Our World in Data – COVID-19 Dataset
- James et al., *An Introduction to Statistical Learning*
- Montgomery & Runger, *Applied Statistics and Probability for Engineers*

15. Appendix – Selected Code Snippets

A.1 Correlation Analysis

```
latest_df = (
    df.sort_values("date")
    .groupby("location")
    .tail(1)
)

corr_cols = [
    "total_cases_per_million",
    "total_deaths_per_million",
    "new_cases_per_million",
    "new_deaths_per_million",
    "stringency_index",
    "cfr"
]

corr_df = latest_df[corr_cols].dropna()

plt.figure(figsize=(10,7))
sns.heatmap(
    corr_df.corr(),
    annot=True,
    fmt=".2f",
    cmap="Purples",
    linewidths=0.5
)
plt.title("Correlation Matrix")
plt.tight_layout()
plt.show()
```

Problem

The dataset contains multiple COVID-19 indicators (cases, deaths, policy measures), and it is not clear how these variables are related to each other at the country level.

Solution

Compute a correlation matrix using the **latest available observation per country** to avoid time-series duplication and obtain a clean country-level comparison.

Why this solution

- Avoids bias caused by repeated daily records
- Quantifies linear relationships between key indicators
- Provides a foundation for feature selection and interpretation

A.2 Feature Selection (Lasso Regression)

```
features = ["new_cases_per_million","stringency_index"]
target = "new_deaths_per_million"

fs_df = latest_df[features + [target]].dropna()

X = StandardScaler().fit_transform(fs_df[features])
y = fs_df[target]

lasso = Lasso(alpha=0.01, max_iter=5000)
lasso.fit(X, y)

pd.Series(lasso.coef_, index=features)
```

Problem

Not all available features contribute equally to predicting COVID-19 outcomes, and using irrelevant variables may reduce model interpretability and performance.

Solution

Apply **Lasso regression**, which performs feature selection by shrinking less important coefficients toward zero.

Why this solution

- Reduces overfitting through regularization
- Automatically highlights influential predictors
- Improves model transparency

A.3 Probability & Distribution Fitting

```
country = "United States"
country_df = df[df["location"] == country]

data = country_df["new_cases"]
data = data[(data.notna()) & (data > 0)]

mu, sigma = stats.norm.fit(data)
shape, loc, scale = stats.lognorm.fit(data, floc=0)
```

Problem

Daily COVID-19 case counts exhibit skewed and heavy-tailed behavior, making it unclear which probability distribution best represents the data.

Solution

Fit **Normal** and **Lognormal** distributions to daily case counts for a selected country and compare their parameters.

Why this solution

- Allows probabilistic interpretation of case counts
- Supports uncertainty and risk analysis
- Helps assess model assumptions

A.4 Hypothesis Testing

```
cutoff_date = pd.to_datetime("2020-06-01")

before = df[df["date"] < cutoff_date]["stringency_index"].dropna()
after = df[df["date"] >= cutoff_date]["stringency_index"].dropna()

t_stat, p_value = stats.ttest_ind(before, after, equal_var=False)

print("t-statistic:", t_stat)
print("p-value:", p_value)
```

```
t-statistic: -8.072775195002984
p-value: 7.101485327797302e-16
```

Problem

It is unclear whether government policy stringency changed significantly after a specific time period.

Solution

Use an **independent two-sample t-test** to compare policy stringency values before and after a defined cutoff date.

Why this solution

- Provides statistical evidence for temporal changes
- Goes beyond visual inspection
- Supports data-driven conclusions

A.5 PCA (Dimensionality Reduction)

```
pca_features = [
    "total_cases_per_million",
    "total_deaths_per_million",
    "new_cases_per_million",
    "new_deaths_per_million",
    "cfr",
    "stringency_index"
]

pca_df = latest_df.dropna(subset=pca_features + ["continent"]).copy()

X = StandardScaler().fit_transform(pca_df[pca_features])

pca = PCA(n_components=2, random_state=42)
components = pca.fit_transform(X)

pca_df["PC1"] = components[:,0]
pca_df["PC2"] = components[:,1]
```

Problem

The dataset contains multiple correlated indicators, making it difficult to visualize and compare countries effectively.

Solution

Apply **Principal Component Analysis (PCA)** after standardization to reduce the feature space to two principal components.

Why this solution

- Reduces dimensionality while preserving variance
- Reveals latent structure and country clusters
- Enables clear visual comparison