COVID-19 VACCINE DATA ANALYSIS

INNOVATION

Title:

Exploring advanced machine learning techniques like clustering or time series forecasting to uncover hidden patterns in vaccine distribution and adverse effects data.

Dataset link:

https://www.kaggle.com/datasets/gpreda/covid-world-vaccination-progress

Abstract:

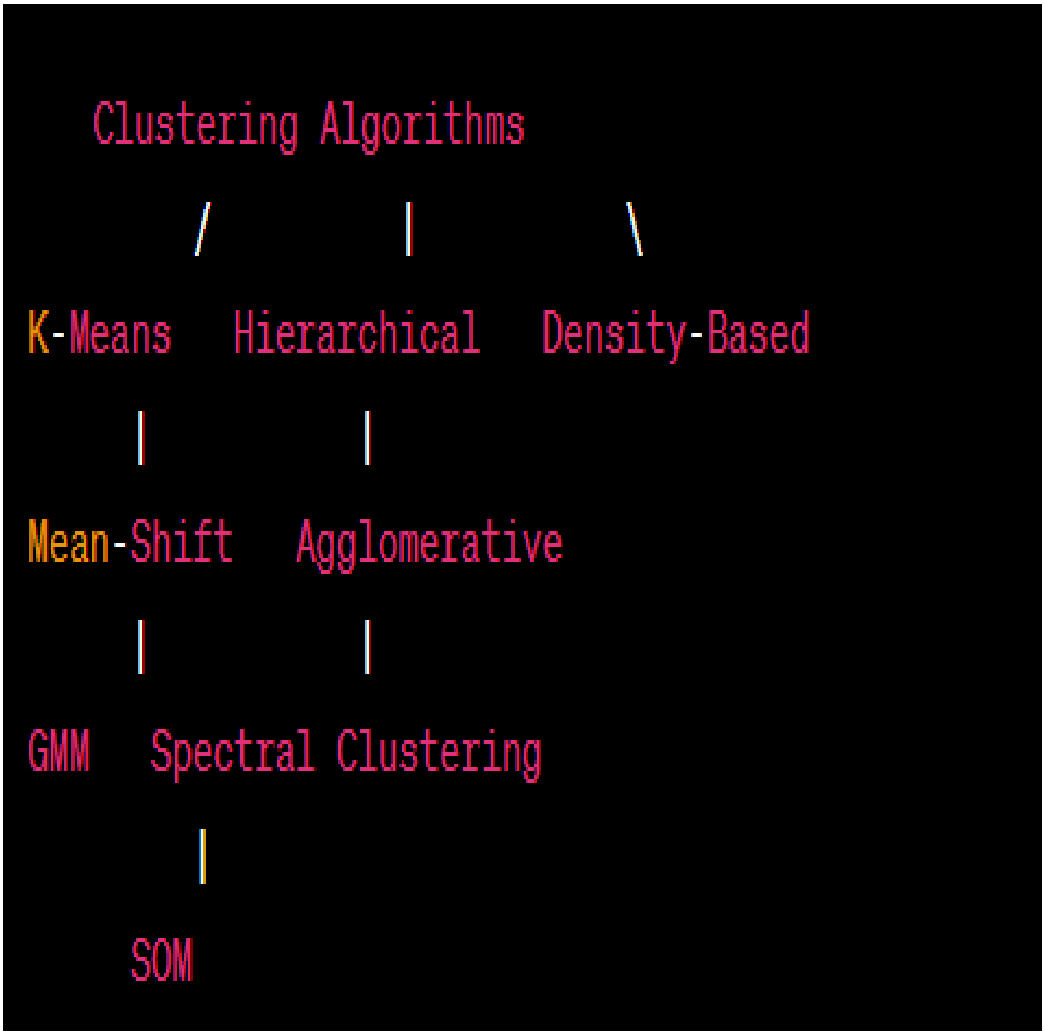
The rapid distribution and administration of vaccines are of paramount importance in the global effort to combat infectious diseases, and the comprehensive analysis of vaccine distribution data can provide valuable insights for optimizing public health strategies. With the ongoing challenges posed by the COVID-19 pandemic and the necessity to manage vaccine supply chains effectively, the utilization of advanced machine learning techniques has emerged as a key innovation.

This document explores the application of cutting-edge machine learning methodologies, particularly clustering and time series forecasting, to analyze vaccine distribution and adverse effects data. We aim to uncover hidden patterns, gain a deeper understanding of distribution dynamics, and enhance decision-making processes.

Key Objectives:

1) **Clustering Analysis:** This document investigates how clustering techniques can help identify unique groups within the vaccine distribution dataset. By segmenting the data into meaningful clusters,

we aim to improve the allocation of resources and optimize vaccine distribution strategies.



2) **Time Series Forecasting:** Time series forecasting enables us to predict future vaccine distribution trends, helping healthcare systems plan for contingencies and allocate resources more efficiently. By analyzing historical data, we can develop predictive models that offer valuable insights into the vaccine distribution process.

Benefits:

- **Improved resource allocation:** By understanding distribution patterns, authorities can allocate vaccines where they are needed most.
- **Enhanced planning:** Time series forecasting can help healthcare systems prepare for future demand and optimize distribution logistics.
- **Informed decision-making:** Data-driven insights enable public health

officials to make well-informed decisions regarding vaccine distribution and management.

Source Code:

1) Clustering:

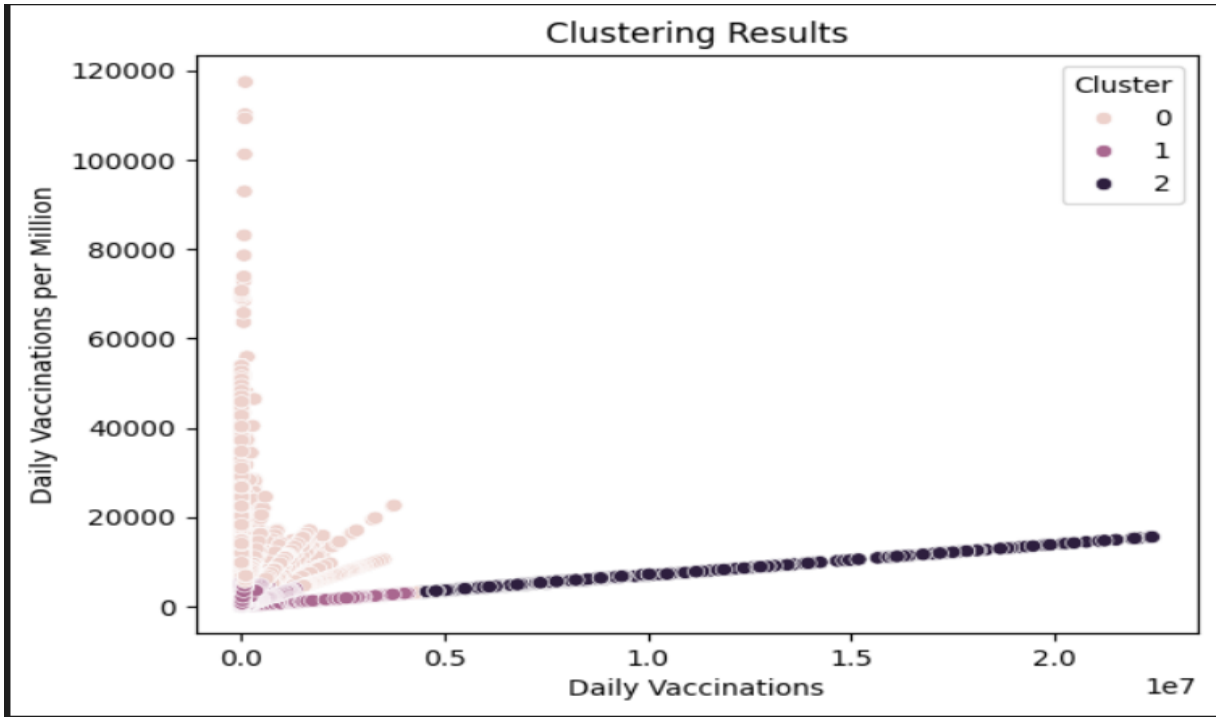
```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import warnings
import matplotlib.pyplot as plt
import seaborn as sns

# Load the vaccine distribution data
data = pd.read_csv('country_vaccinations.csv')
data_copy = data.copy()

# Select relevant features for clustering
features = data[['daily_vaccinations', 'daily_vaccinations_per_million']]

# Data Preprocessing
# 1. Handling missing values (if any)
features.fillna(0, inplace=True)  # Replace missing values with zeros

# 2. Standardization (optional but recommended)
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

# 3. Dimensionality Reduction (optional but recommended for high-dimensional data)
# Using Principal Component Analysis (PCA) to reduce dimensionality
pca = PCA(n_components=2)  # Adjust the number of components as needed
reduced_features = pca.fit_transform(scaled_features)

# At this point, 'reduced_features' contains the preprocessed data suitable for clustering

from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=3)
clusters = kmeans.fit_predict(reduced_features)

# Add cluster labels to the original DataFrame
data['Cluster'] = clusters
selected_attributes = data[['daily_vaccinations', 'daily_vaccinations_per_million', 'Cluster']]
selected_attributes.to_csv('vaccine_distribution_clusters.csv', index=False)
# Print the cluster assignments
print(data[['country', 'daily_vaccinations', 'daily_vaccinations_per_million', 'Cluster']])
```

```
# Visualize the clusters
sns.scatterplot(x='daily_vaccinations', y='daily_vaccinations_per_million', hue='Cluster',
data=data)
plt.xlabel('Daily Vaccinations')
plt.ylabel('Daily Vaccinations per Million')
plt.title('Clustering Results')
plt.show()
```

Output:

```
          country  daily_vaccinations  daily_vaccinations_per_million  \
0     Afghanistan                 NaN                             NaN
1     Afghanistan              1367.0                            34.0
2     Afghanistan              1367.0                            34.0
3     Afghanistan              1367.0                            34.0
4     Afghanistan              1367.0                            34.0
...           ...                 ...                             ...
86507    Zimbabwe             69579.0                          4610.0
86508    Zimbabwe             83429.0                          5528.0
86509    Zimbabwe             90629.0                          6005.0
86510    Zimbabwe            100614.0                          6667.0
86511    Zimbabwe            103751.0                          6874.0
```

```
       Cluster
0            1
1            1
2            1
3            1
4            1
...        ...
86507        1
86508        0
86509        0
86510        0
86511        0

[86512 rows x 4 columns]
```



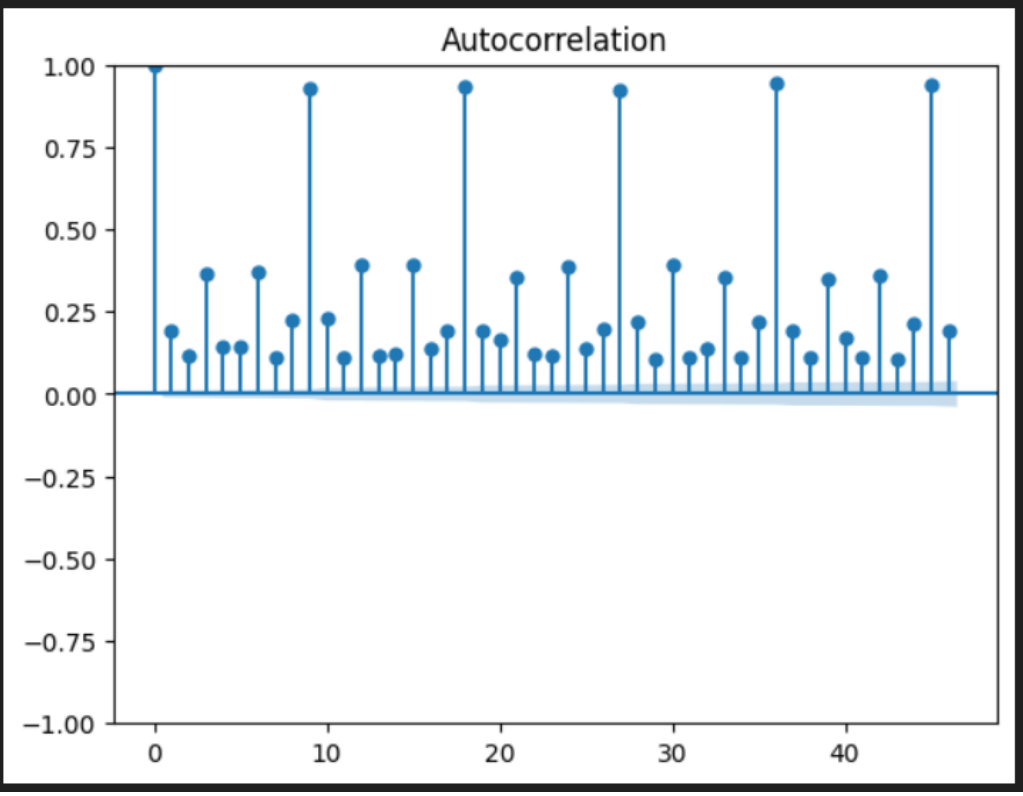
3) Time series forecasting:

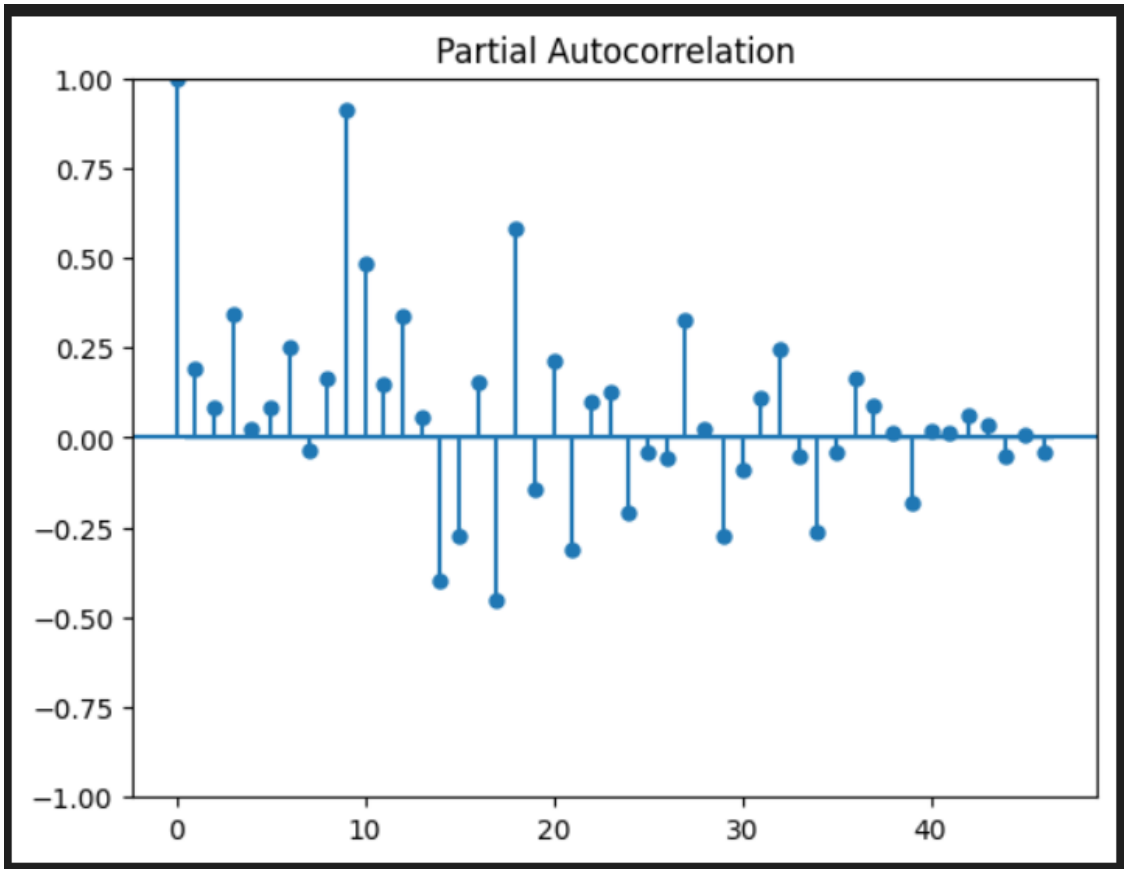
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import adfuller
import warnings
warnings.filterwarnings('ignore')
# Load your time series data
data = pd.read_csv('country_vaccinations_by_manufacturer.csv')
data['date'] = pd.to_datetime(data['date'])
data.set_index('date', inplace=True)

# Check for stationarity
result = adfuller(data['total_vaccinations'])
print(f'ADF Statistic: {result[0]}')
print(f'p-value: {result[1]}')


# If the data is non-stationary, difference it to make it stationary
if result[1] > 0.05:
    data['total_vaccinations'] = data['total_vaccinations'].diff().dropna()

# Plot the ACF and PACF to determine model orders (p and q)
plot_acf(data['total_vaccinations'])
plot_pacf(data['total_vaccinations'])
```

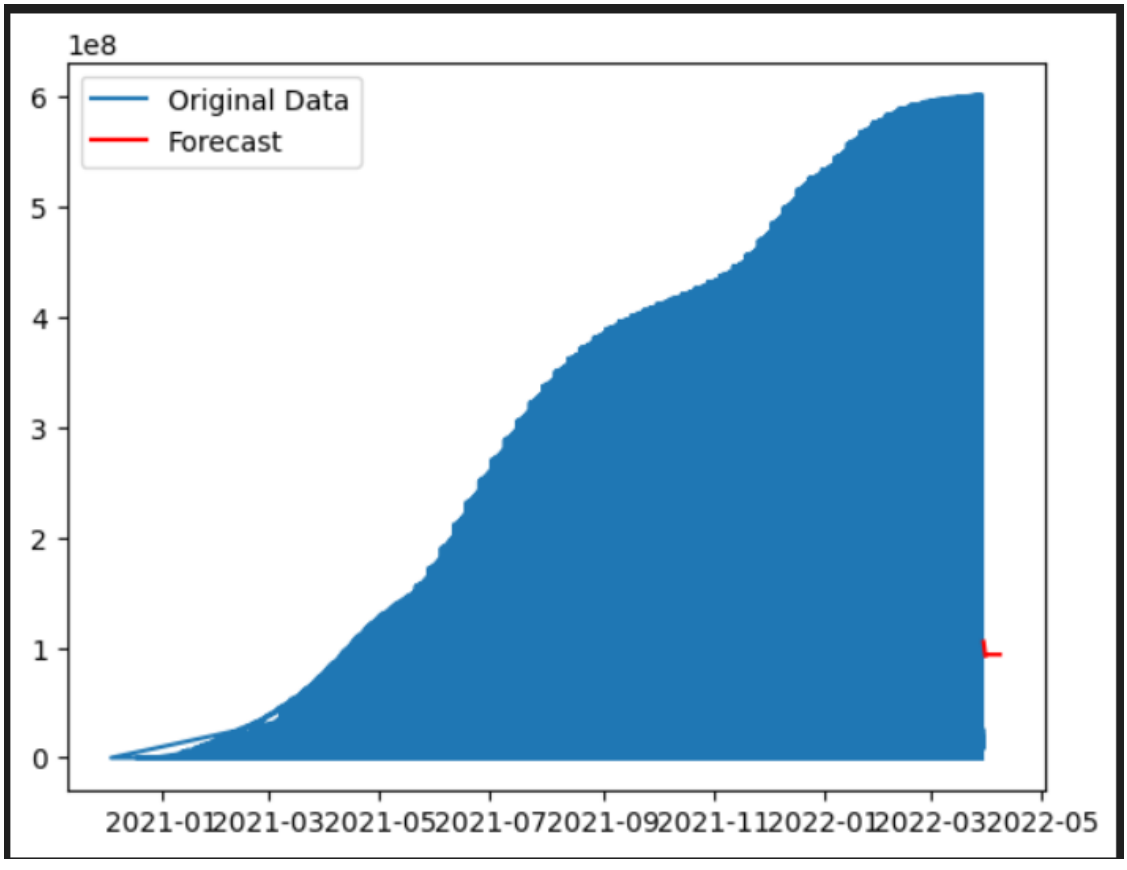
```
plt.show()

# Fit the ARIMA model
model = ARIMA(data['total_vaccinations'], order=(1, 1, 1))  # Adjust p, d, and q as needed
model_fit = model.fit()

# Forecast future values
forecast_steps = 10  # Number of steps to forecast
forecast = model_fit.forecast(steps=forecast_steps)

# Create a date range for the forecasted values
forecast_index = pd.date_range(start=data.index[-1], periods=forecast_steps + 1)

# Plot the original data and forecasted values
plt.plot(data['total_vaccinations'], label='Original Data')
plt.plot(forecast_index[1:], forecast, label='Forecast', color='red')
plt.legend()
plt.show()
```

Output:

```
ADF Statistic: -3.3481016547444016
p-value: 0.012860554109584074
```




Partial Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40


1e8
Original Data
Forecast
6
5
4
3
2
1
0
2021-01
2021-03
2021-05
2021-07
2021-09
2021-11
2022-01
2022-03
2022-05

Conclusion:

In the face of global health crises like the COVID-19 pandemic, advanced machine learning techniques offer innovative solutions to the challenges of vaccine distribution. This document explores the application of clustering and time series forecasting to uncover hidden patterns in vaccine distribution and adverse effects data. By doing so, we aim to contribute to more effective vaccine allocation, better planning, and informed decision-making in the realm of public health.