Assignment1_v6

April 15, 2023

Enna

Assessment 1 SP2 2023

0.0.1 Introduction

COVID-19 data from five main Australian states: NSW, Vic, QLD, SA, and WA will be analysed in this study. The data includes the number of new cases and deaths reported each day. Due to the Australian government's reporting regulations, the files contain a mix of daily and weekly data, so daily data will be aggregated into weekly data. The data spans the period beginning with the epidemic and ending on September 9, 2022. There are 6 variables and 4935 observations in all.

---

[1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
# To ignore warnings in the notebook
import warnings
warnings.filterwarnings("ignore")
# Set the default style for all plots
sns.set_style("white")

%matplotlib inline
```

Data cleaning

By integrating and cleansing data from several files, I prepared a dataset from **COVID Live** (https://covidlive.com.au/) on COVID-19 cases and deaths in five Australian states. The cleaning procedure sought to standardise data across all states, assuring consistency and allowing study and understanding of Australia's COVID-19 situation.

[2]:
```python
# all data files are in the same folder as the notebook

# Define a list of state abbreviations
states = ['nsw', 'qld', 'vic', 'wa', 'sa']

# Create an empty list to store the dataframes
```

```
dfs = []

# Loop over the state abbreviations and read in the case and death data
for state in states:
    case_file = f'daily_cases_{state}.tsv'
    death_file = f'daily_death_{state}.tsv'
    case_data = pd.read_csv(case_file, sep='\t')
    death_data = pd.read_csv(death_file, sep='\t')
    # Merge the case and death data into a single dataframe
    merged_data = pd.merge(case_data, death_data, on='DATE')

    # Add a state column to the dataframe
    merged_data['state'] = state
    dfs.append(merged_data)

# Concatenate all of the dataframes into a single dataframe
all_data = pd.concat(dfs, axis=0)
all_data = all_data.rename(columns={'VAR_x': 'var_cases', 'VAR_y':␣
 ↪'var_deaths', 'NET_x': 'net_cases', 'NET_y': 'new_deaths'})
# change all column names to lowercase
all_data.columns = all_data.columns.str.lower()
```

The initial stage of data analysis is exploratory data analysis (EDA), which aids in understanding the data before proceeding with the analysis. Before analysing data, it must be cleaned and prepared. In addition to cleaning and preparing the data previously, I cleaned the COVID-19 dataset further to ensure consistency and correctness in analysis.

[3]:
```
# check if all var_cases and var_deaths are null
print(all_data.shape)
# count the number of missing values in var_cases and var_deaths
print(all_data['var_cases'].isnull().sum())
print(all_data['var_deaths'].isnull().sum())
# drop the column var_cases and var_deaths
all_data = all_data.drop(columns=['var_cases', 'var_deaths'])
all_data.info()
```

```
(4935, 9)
4935
4935
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4935 entries, 0 to 986
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   date        4935 non-null   object
 1   new         4927 non-null   object
 2   cases       4935 non-null   object
 3   net_cases   4935 non-null   object
 4   deaths      4935 non-null   object
 5   new_deaths  4935 non-null   object
 6   state       4935 non-null   object
dtypes: object(7)
memory usage: 308.4+ KB
```

The cleaned dataset contains COVID-19 case and death data for five Australian states (New South Wales, Queensland, Victoria, Western Australia, and South Australia) from the beginning of the epidemic until September 9, 2022. The dataset consists of 4935 observations and 7 variables: date, net_cases, new_cases, net_deaths, new_deaths, state, and week.

The data cleaning process included merging the daily case and death data for each state, standardizing the data across states, and filling missing values in the 'new_deaths' column with zeros. The 'var_cases' and 'var_deaths' columns were dropped due to containing all missing values.

[4]:
```
# get all the missing values in 'new' column
all_data[all_data['new'].isnull()]
```

[4]:
```
          date  new      cases net_cases deaths new_deaths state
986  25 Jan 20  NaN          3         -      0          -   nsw
12   30 Dec 22  NaN  1,743,616         0  2,405          0   qld
33   04 Sep 22  NaN  1,610,200         0  1,986          0   qld
34   03 Sep 22  NaN  1,610,200         0  1,986          0   qld
986  25 Jan 20  NaN          0         -      0          -   qld
986  25 Jan 20  NaN          1         -      0          -   vic
986  25 Jan 20  NaN          0         -      0          -    wa
986  25 Jan 20  NaN          0         -      0          -    sa
```

[5]:
```
# get all the 0 values in 'new' column
all_data[all_data['new'] == 0]
# there is no 0 value in 'new' column
# so it could be safely assumed that the missing values in 'new' column are 0
```

[5]:
```
Empty DataFrame
Columns: [date, new, cases, net_cases, deaths, new_deaths, state]
Index: []
```

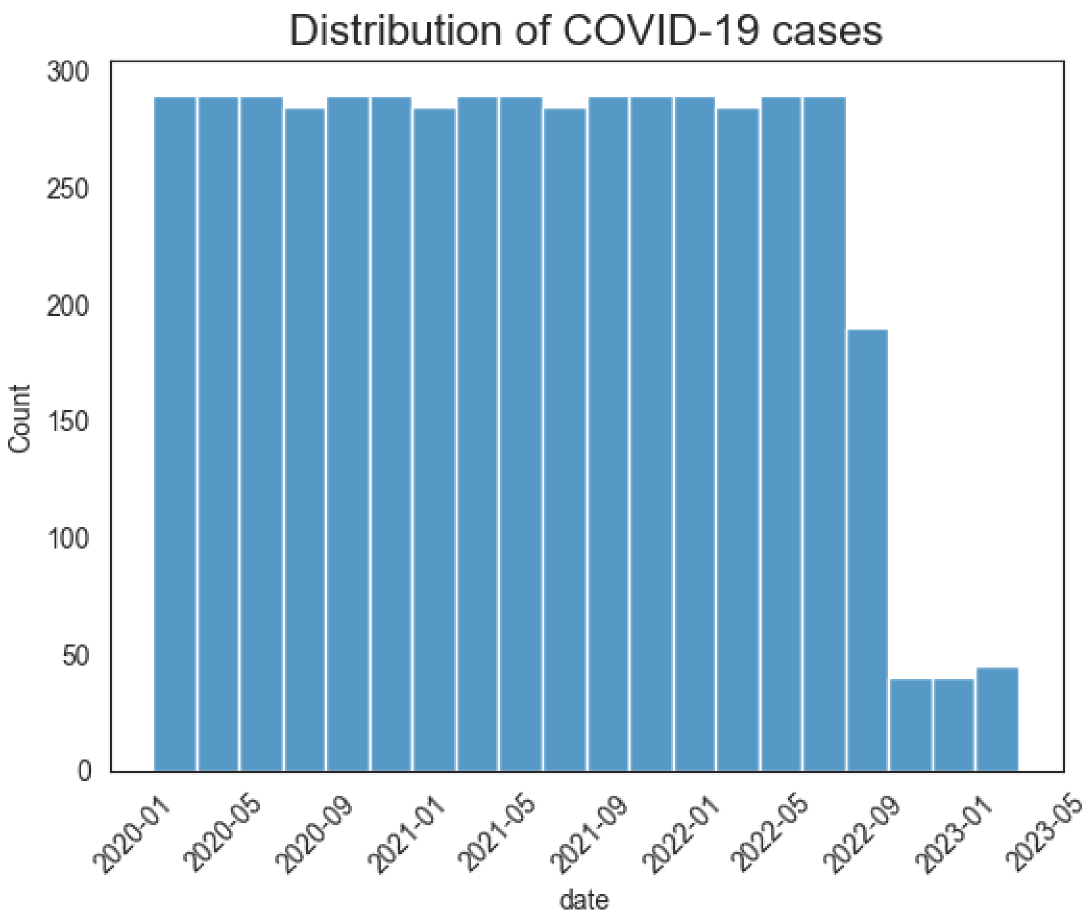
[6]:
```
# replace all the NAN values in 'new' column with 0
all_data['new'] = all_data['new'].fillna(0)
all_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4935 entries, 0 to 986
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   date        4935 non-null   object
 1   new         4935 non-null   object
 2   cases       4935 non-null   object
 3   net_cases   4935 non-null   object
 4   deaths      4935 non-null   object
 5   new_deaths  4935 non-null   object
 6   state       4935 non-null   object
dtypes: object(7)
memory usage: 308.4+ KB
```

[7]:
```
# Change the date column to datetime
all_data['date'] = pd.to_datetime(all_data['date'])

# Change the state column to categorical
all_data['state'] = all_data['state'].astype('category')

# create a new dataframe with only date and state columns
date_state = all_data[['date', 'state']]

# create a new dataframe with only new, cases and deaths columns
new_cases_deaths = all_data[['new', 'deaths', 'cases', 'net_cases',␣
 ↪'new_deaths']]

# Remove commas from the number columns, and remove all non-numeric characters,␣
 ↪replace float to int, NaN to 0
new_cases_deaths = new_cases_deaths.replace(',', '', regex=True).
 ↪replace('[^0-9]', '', regex=True)
new_cases_deaths = new_cases_deaths.replace('', np.nan)
new_cases_deaths = new_cases_deaths.astype(float).fillna(0).astype(int)
new_cases_deaths.tail(10)

merged_data = pd.concat([date_state, new_cases_deaths], axis=1)
# rearrange the dataframe as date from 2020-01-05 to 2020-03-29 from oldest to␣
 ↪newest
merged_data = merged_data.sort_values(by=['date'], ascending=True)

# check the three columns that the instruction asked to drop, which is the last␣
 ↪three columns
# merged_data.tail(10)
```

[8]:
```
# drop the last three columns
merged_data = merged_data.drop(columns=['net_cases', 'cases'])

# Group by state and calculate the cumulative sum of new cases
merged_data['case'] = merged_data.groupby('state')['new'].cumsum()

# Sort the data by date and state
merged_data = merged_data.sort_values(by=['date', 'state'])
```

[9]:
```
# check the mix of data - daily numbers before September 9, 2022, and weekly␣
↪data after that
sns.histplot(data=merged_data, x='date', kde=False)
# Set x-axis labels to 45 degree angle for better readability
plt.xticks(rotation=45)
plt.title('Distribution of COVID-19 cases', fontsize=16)
plt.show()
```



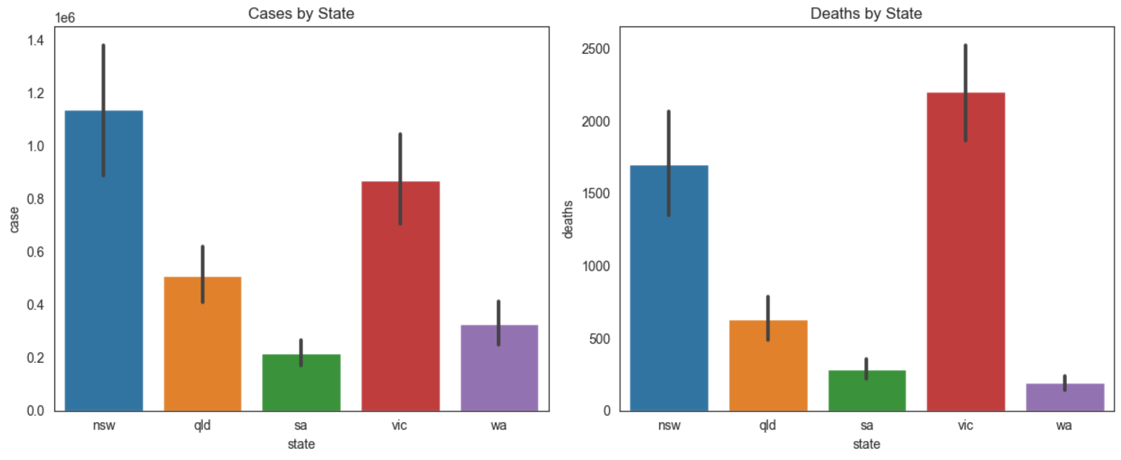
These code generates a histogram to depict the distribution of daily COVID-19 cases and deaths prior to September 9, 2022, and weekly statistics after that date.

[10]:
```
# Create a copy of the original dataframe
week_dis = merged_data.copy()
# Convert the 'date' column to a pandas datetime object
week_dis['date'] = pd.to_datetime(week_dis['date'])

# Create a new column 'week_start' with the start of each week
```

```
week_dis['week_start'] = week_dis['date'] - pd.to_timedelta((week_dis['date'].
 ↪dt.dayofweek), unit='d')
# Group the data by 'week_start' and 'state', and sum the 'new' column
week_new = week_dis.groupby(['week_start', 'state']).agg({'new': 'sum', 'case':␣
 ↪'max', 'deaths': 'max', 'new_deaths':'sum'}).reset_index()
# Rename the 'week_start' column to 'date'
week_new = week_new.rename(columns={'week_start': 'date'})

# Reorder the columns in the dataframe
week_new = week_new[['date', 'state', 'new', 'case', 'deaths', 'new_deaths']]
# Sort the dataframe by date and state
week_new = week_new.sort_values(['date', 'state'])
```

The code above creates a copy of the merged_data dataframe and converts the 'date' column to a pandas datetime object. It then creates a new column 'week_start' that corresponds to the start of each week. The data is then grouped by 'week_start' and 'state', and the 'new' column is summed up. The resulting dataframe is then renamed and reordered, and sorted by date and state. The resulting dataframe contains weekly statistics of COVID-19 cases and deaths for each state.

[11]:
```
# EDA using barplots is performed to understand the data. The barplot shows the␣
 ↪number of new COVID-19 cases and deaths in each state.

fig, axes = plt.subplots(ncols=2, figsize=(12, 5))

# plot the first subplot
sns.barplot(x='state', y='case', data=week_new, ax=axes[0])
axes[0].set_title('Cases by State')

# plot the second subplot
sns.barplot(x='state', y='deaths', data=week_new, ax=axes[1])
axes[1].set_title('Deaths by State')

# adjust the layout and show the plot
plt.tight_layout()
plt.show()
```



These code generate two bar plots to visualize the number of new COVID-19 cases and deaths in each state. The first bar plot shows the number of new cases in each state, while the second bar plot shows the number of new deaths in each state.

Description

date: the date of the observation
state: the state where the observation was made
new: the number of new COVID-19 cases reported on that date
cases: the total number of confirmed COVID-19 cases in the state up to that date
_deaths: the total number of COVID-19 deaths in the state up to that date**
new_deaths**: the number of new COVID-19 deaths reported on that date

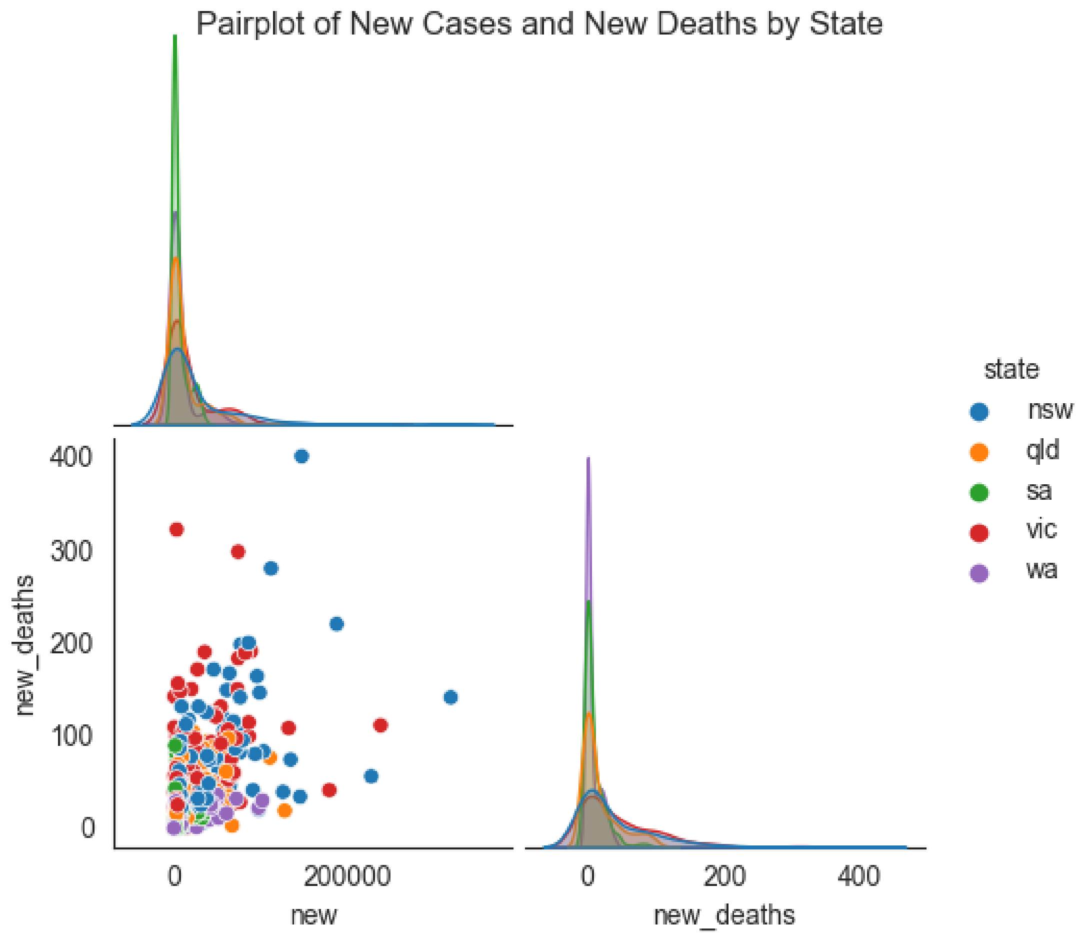
[12]:
```
week_new.tail(10)
```

[12]:
```
           date state   new     case  deaths  new_deaths
820  2023-03-13   nsw  8905  3955025    6551          22
821  2023-03-13   qld  3049  1809086    2818          22
822  2023-03-13    sa  2347   834784    1360           0
823  2023-03-13   vic  3960  2967281    7399          29
824  2023-03-13    wa  2625  1319535     985          29
825  2023-03-20   nsw  8563  3963588    6573          22
826  2023-03-20   qld  3207  1812293    2834          16
827  2023-03-20    sa  2888   837672    1358           0
828  2023-03-20   vic  4467  2971748    7424          25
829  2023-03-20    wa     0  1319535     985           0
```

0.0.2 Distributions of New Cases and Deaths

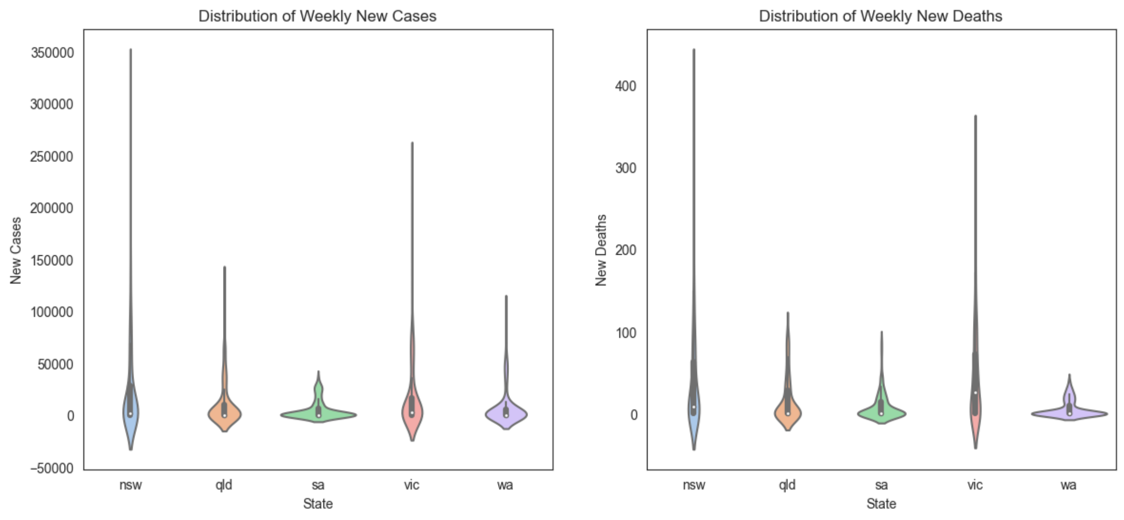
The graphs below illustrate the distributions of new cases and deaths in each state. From the start of the pandemic until September 9, 2022, the graphs show the weekly number of new cases and deaths in each state. NSW had the most mean new cases, whereas Victoria had the most mean new deaths, followed by NSW, Queensland, South Australia, and Western Australia. The number of new cases and deaths climbed dramatically in all states during the second wave in mid-2020, followed by a drop near the end of the year (see chapter Normalized Numbers of New Cases by Population).

[13]:
```python
# Create pairplot with hue = 'state'
g = sns.pairplot(data=week_new, vars=['new', 'new_deaths'], hue='state',␣
 ↪corner=True)

# Add a title to the pairplot
plt.suptitle('Pairplot of New Cases and New Deaths by State', fontsize=12)
plt.show()
```



The pairplot is a useful visualization to compare the distributions across states. It appears that there is significant variation in both the mean and the spread of new cases and deaths across the five states.

[14]:
```
fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(14,6))

# Plot the distribution of new cases by state in the first subplot
sns.violinplot(data=week_new, x="state", y="new", palette="pastel", ax=ax1)
ax1.set_title('Distribution of Weekly New Cases')
ax1.set_xlabel('State')
ax1.set_ylabel('New Cases')

# Plot the distribution of new deaths by state in the second subplot
sns.violinplot(data=week_new, x="state", y="new_deaths", palette="pastel",␣
 ↪ax=ax2)
ax2.set_title('Distribution of Weekly New Deaths')
ax2.set_xlabel('State')
ax2.set_ylabel('New Deaths')
plt.show()
```

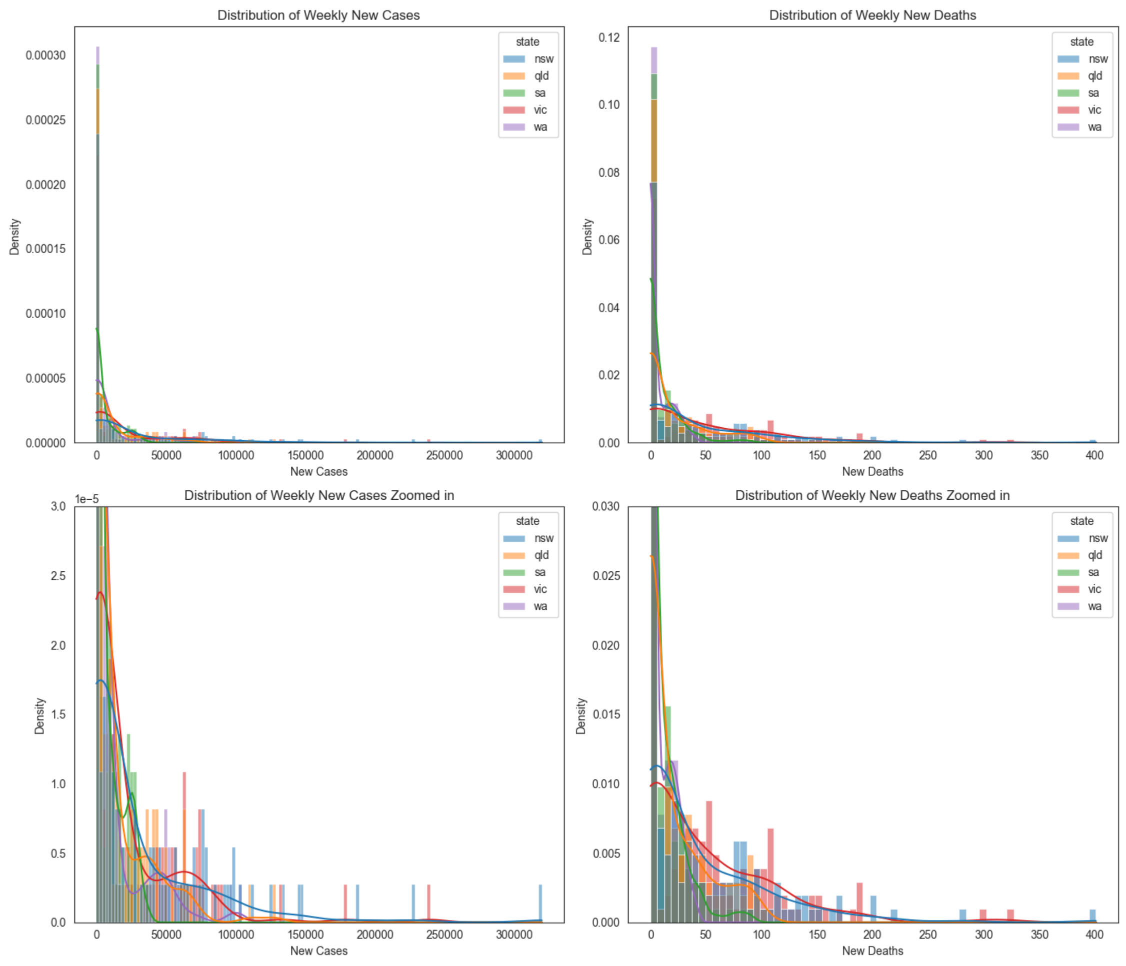


The two violin plots showing the distribution of weekly new cases and new deaths by state. The x-axis shows the five states, and the y-axis represents the number of new cases and new deaths. The wider parts of the violin plots represent the areas where the density of data points is higher, whereas the narrower parts represent the areas where the density is lower.

The patterns observed in the violin plots and the number of new cases and deaths for each state can provide insights into the spread and impact of COVID-19 in each state. In terms of the weekly new cases, it seems that all states have been able to control the spread of the virus to some extent, as the highest density is at 0 new cases. However, it is important to note that some states, such as NSW and Victoria, have had much higher numbers of new cases overall than other states, indicating that the virus has been more widespread and harder to contain in those areas. Additionally, the fact that the density gets wider at higher numbers of new cases suggests that there have been some outbreaks in each state that have resulted in higher numbers of cases.

In terms of the weekly new deaths, it seems that QLD has had a wider distribution of new deaths than other states, suggesting that the virus has been more deadly in that state. Additionally, SA have had lower numbers of new cases than WA overall, but higher numbers of new deaths, suggesting that the virus has had a greater impact on those states' populations.

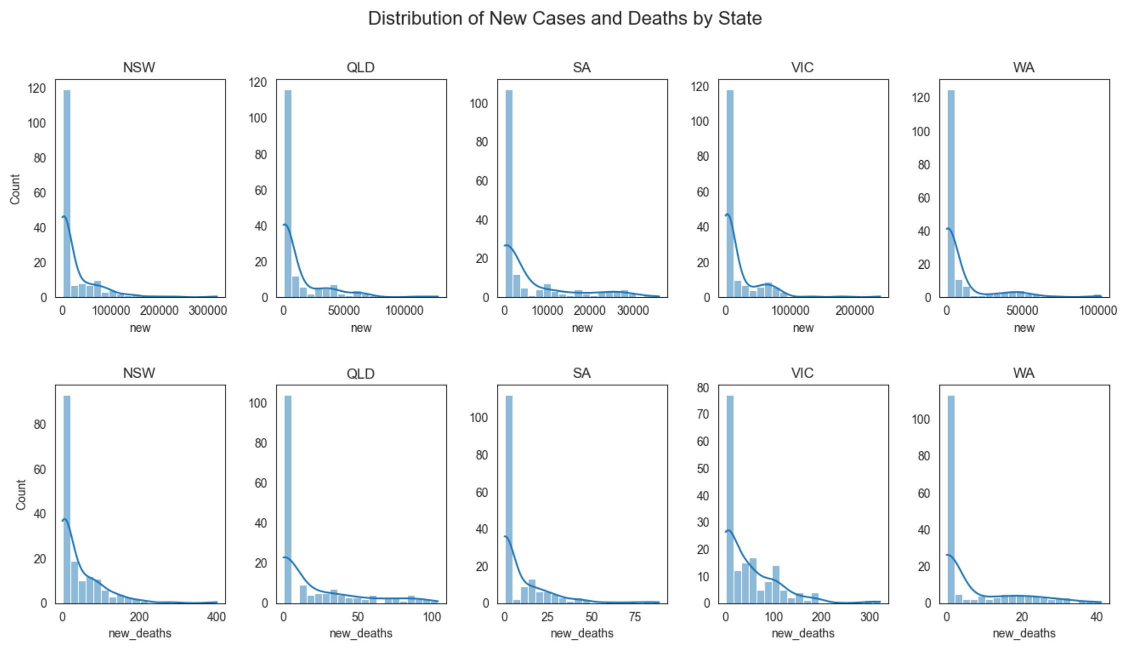
[15]:
```
import matplotlib.gridspec as gridspec

fig = plt.figure(figsize=(14, 12))
gs = gridspec.GridSpec(nrows=2, ncols=2, height_ratios=[2, 2])

# Plot the original distribution of new cases by state in the first subplot
ax1 = fig.add_subplot(gs[0, 0])
sns.histplot(data=week_new, x="new", hue="state", kde=True, stat='density',␣
 ↪common_norm=False, ax=ax1)
ax1.set_title('Distribution of Weekly New Cases')
ax1.set_xlabel('New Cases')
ax1.set_ylabel('Density')

# Plot the original distribution of new deaths by state in the second subplot
ax2 = fig.add_subplot(gs[0, 1])
sns.histplot(data=week_new, x="new_deaths", hue="state", kde=True,␣
 ↪stat='density', common_norm=False, ax=ax2)
ax2.set_title('Distribution of Weekly New Deaths')
ax2.set_xlabel('New Deaths')
ax2.set_ylabel('Density')

# Plot the zoomed-in distribution of new cases by state in the third subplot
ax3 = fig.add_subplot(gs[1, 0])
sns.histplot(data=week_new, x="new", hue="state", kde=True, stat='density',␣
 ↪common_norm=False, ax=ax3)
ax3.set_title('Distribution of Weekly New Cases Zoomed in')
ax3.set_xlabel('New Cases')
ax3.set_ylabel('Density')
ax3.set_ylim(top=0.00003)

# Plot the zoomed-in distribution of new deaths by state in the fourth subplot
ax4 = fig.add_subplot(gs[1, 1])
sns.histplot(data=week_new, x="new_deaths", hue="state", kde=True,␣
 ↪stat='density', common_norm=False, ax=ax4)
ax4.set_title('Distribution of Weekly New Deaths Zoomed in')
ax4.set_xlabel('New Deaths')
ax4.set_ylabel('Density')
ax4.set_ylim(top=0.03)
plt.tight_layout()
plt.show()
```



The above code creates a figure with four subplots, showing the distribution of weekly new cases and deaths by state, with a focus on the range of values with higher density. The first two subplots show the original distribution, while the third and fourth subplots zoom in on the area with the highest density.

The first subplot shows that the distribution of new cases is highly skewed to the right, with a large proportion of states having a mode at zero new cases. The states with the highest mean new cases, NSW and Victoria, also have the highest density at higher values. The second subplot shows that the distribution of new deaths is also highly skewed to the right, with a large proportion of states having a mode at zero new deaths.

[16]:
```python
fig, axes = plt.subplots(nrows=2, ncols=5, figsize=(16,8))

# loop through each state and plot its distribution in a separate subplot
for i, state in enumerate(['nsw', 'qld', 'sa', 'vic', 'wa']):
    state_cases = week_new[week_new['state'] == state]['new']
    state_deaths = week_new[week_new['state'] == state]['new_deaths']
    # plot histogram for new cases in the appropriate subplot
    sns.histplot(state_cases, bins=20, kde=True, ax=axes[0][i])

    # plot histogram for new deaths in the appropriate subplot
    sns.histplot(state_deaths, bins=20, kde=True, ax=axes[1][i])

    # set the title for the subplot
    axes[0][i].set_title(state.upper())
    axes[1][i].set_title(state.upper())

    # remove y-axis label for all subplots except the first one in each row
    if i > 0:
        axes[0][i].set_ylabel('')
        axes[1][i].set_ylabel('')

# set the overall title for the plot
fig.suptitle("Distribution of New Cases and Deaths by State", fontsize=16)
plt.subplots_adjust(wspace=0.3, hspace=0.4)
plt.show()
```



This code creates a 2x5 grid of subplots, each showing the distribution of new cases and new deaths for a different state.

[17]:
```
# Summary statistics of weekly new cases by state
print("Weekly New Cases by State")
week_new.groupby('state')['new'].describe().round(2)
```

```
Weekly New Cases by State
```

```
[17]:        count      mean       std  min    25%     50%       75%       max
      state
      nsw    166.0  23877.04  45638.27  0.0  39.00  1601.5  28310.75  319632.0
      qld    166.0  10917.43  21227.99  0.0   8.00    22.0  10063.75  127914.0
      sa     166.0   5046.22   8830.82  0.0   3.00    13.0   6424.25   36203.0
      vic    166.0  17902.10  33497.71  0.0  25.75  2473.5  16619.00  238588.0
      wa     166.0   7949.01  18016.62  0.0   3.00    12.0   5024.00  102305.0
```

```
[18]: # Summary statistics of weekly new deaths by state
      print("Weekly New Deaths by State")
      week_new.groupby('state')['new_deaths'].describe().round(2)
```

```
Weekly New Deaths by State
```

```
[18]:        count   mean    std  min  25%   50%    75%    max
      state
      nsw    166.0  39.60  60.05  0.0  0.0   8.5  63.75  401.0
      qld    166.0  17.08  27.43  0.0  0.0   0.0  28.00  104.0
      sa     166.0   8.45  15.69  0.0  0.0   0.0  14.00   89.0
      vic    166.0  44.72  57.74  0.0  0.0  26.0  72.75  322.0
      wa     166.0   5.98  10.16  0.0  0.0   0.0   9.75   41.0
```

The mean weekly new cases in NSW were 23877.04, followed by VIC at 17902.10, QLD at 10917.43, WA at 7949.01, and SA at 5046.22. NSW also had the highest standard deviation in weekly new cases (45638.27), indicating that the number of new cases varied more significantly in NSW than in the other states.The maximum weekly new cases in VIC were 238588, followed by 319632 in NSW. VIC had the highest mean of new fatalities at 44.72, followed by NSW at 39.60, QLD at 17.08, SA at 8.45, and WA at 5.98. The maximum weekly new fatalities in VIC were 322, followed by 401 in NSW. Overall, it appears that NSW and VIC have had the largest weekly numbers of new cases and deaths, with NSW having a greater range of variance in new case counts. When compared to NSW and VIC, QLD, SA, and WA had lower mean numbers of new cases and deaths.

It's also worth mentioning that this analysis is based on summary information, and further research, such as hypothesis testing or modeling, would be required to draw more definitive conclusions about variances in the distribution of new cases and deaths across the five states.

0.0.3 History of COVID-19 in Different States

The graphs below depict the history of COVID-19 in each state, beginning with the week following the first 1000 cases reported. The total weekly number of new instances is displayed. The number of new cases climbed fast in all states during the second wave in mid-2020, followed by a decrease in the number of new cases and deaths near the end of the year. The state of New South Wales has the most cumulative instances, followed by Victoria, Queensland, Western Australia, and South Australia.

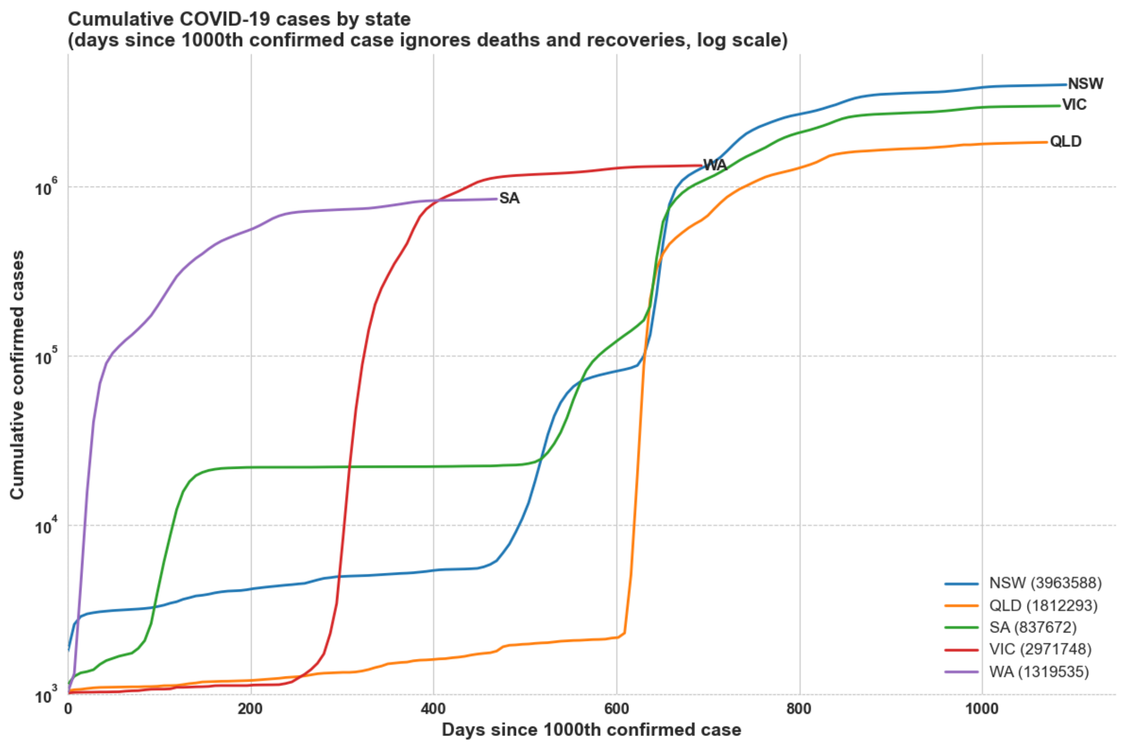
[19]:
```
# Filter the data to include only weeks with 1000 or more cases
week = week_new.loc[week_new['case'] >= 1000]

# Calculate the number of days since the 1000th confirmed case for each state
week['days_since_1000'] = (week['date'] - week.groupby('state')['date'].
 ↪transform(min)).dt.days
week = week.loc[week['days_since_1000'] >= 0].copy()  # Add .copy() to avoid␣
 ↪the warning
```

[20]:
```
# Get the last cumulative confirmed case for each state
last_cases = week.groupby('state').tail(1)
# extract the last cumulative confirmed case for each state
last_cases['case'].values
last_cases
```

[20]:
```
           date state   new     case  deaths  new_deaths  days_since_1000
825 2023-03-20   nsw  8563  3963588    6573          22             1092
826 2023-03-20   qld  3207  1812293    2834          16             1071
827 2023-03-20    sa  2888   837672    1358           0              469
828 2023-03-20   vic  4467  2971748    7424          25             1085
829 2023-03-20    wa     0  1319535     985           0              693
```

[21]:
```
sns.set_style('white')
fig, ax = plt.subplots(figsize=(12, 8))

# Plot the data for each state
for i, state in enumerate(states):
    state_data = week[week['state'] == state]
    plt.plot(state_data['days_since_1000'], state_data['case'],
             label=state.upper(), linewidth=2)

    # Label each line with the name of the state at the end of the line
    plt.text(state_data['days_since_1000'].iloc[-1]+2,
             state_data['case'].iloc[-1],
             state.upper(), fontsize=12, fontweight='bold', ha='left',␣
 ↪va='center')

# Set the y-axis to a logarithmic scale and start at 1000
plt.yscale('log')
plt.ylim([1000, None])

# Set the x-axis to start at 0
plt.xlim([0, None])

# Set the x-axis label
plt.xlabel('Days since 1000th confirmed case', fontsize=14, fontweight='bold')
```

```
# Set the y-axis label
plt.ylabel('Cumulative confirmed cases', fontsize=14, fontweight='bold')

# Customize axes labels and ticks
plt.xticks(fontsize=12, fontweight='bold')
plt.yticks(fontsize=12, fontweight='bold')

# Remove top and right spines
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)

# Customize remaining spines
plt.gca().spines['left'].set_linestyle((0, (2, 2)))
plt.gca().spines['bottom'].set_linestyle((0, (2, 2)))
plt.gca().spines['left'].set_color('#c7c7c7')
plt.gca().spines['bottom'].set_color('#c7c7c7')
plt.grid(axis='y', linestyle='--', color='#c7c7c7')
plt.grid(axis='x', color='#c7c7c7')

# Add legend box with the last cumulative confirmed cases for each state
last_cases = week.groupby('state').tail(1)
last_cases_values = last_cases['case'].values
legend_text = ['{} ({})'.format(state.upper(), last_cases_values[i]) for i,␣
  ↪state in enumerate(last_cases['state'].unique())]
plt.legend(legend_text, loc='lower right', fontsize=12, frameon=False)

# Set title and adjust the layout
plt.title('Cumulative COVID-19 cases by state\n(days since 1000th confirmed␣
  ↪case ignores deaths and recoveries, log scale)', fontsize=15,␣
  ↪fontweight='bold', loc='left')
plt.tight_layout()

plt.show()
```



The graphic depicts the cumulative confirmed cases of COVID-19 in various Australian states over time, beginning with the week following the 1000th confirmed case in each state. The graphic has a logarithmic scale for the y-axis, which begins at 1000. The number of days since the 1000th verified case is represented on the x-axis.

The graphic is useful for visually comparing COVID-19 evolution in different states. The graphic also aids in evaluating the trend of confirmed cases for each state over time. For example, the plot reveals that New South Wales has the most confirmed cases, followed by Victoria and Queensland. South Australia has had fewer confirmed cases than the other states.

Furthermore, it reveals that the number of confirmed cases in SA climbed significantly until a certain point, after which it slowed. The rate of growth, however, varies by state, with some states experiencing a greater reduction in the number of verified cases than others.

One thing to note is that the visualization only shows the cumulative confirmed cases and doesn't show the number of deaths or recoveries. Additionally, it's important to keep in mind that different states may have different testing protocols and capacities, which can affect the number of reported cases.

Overall, the graph shows that all states experienced a significant increase in the number of new cases during the second wave in mid-2020. However, since then, the number of new cases has decreased and remained relatively stable, with some minor fluctuations. It's also interesting to note that New South Wales has the most cumulative cases, followed by Victoria, Queensland, Western Australia, and South Australia.

0.0.4 Normalized Numbers of New Cases by Population

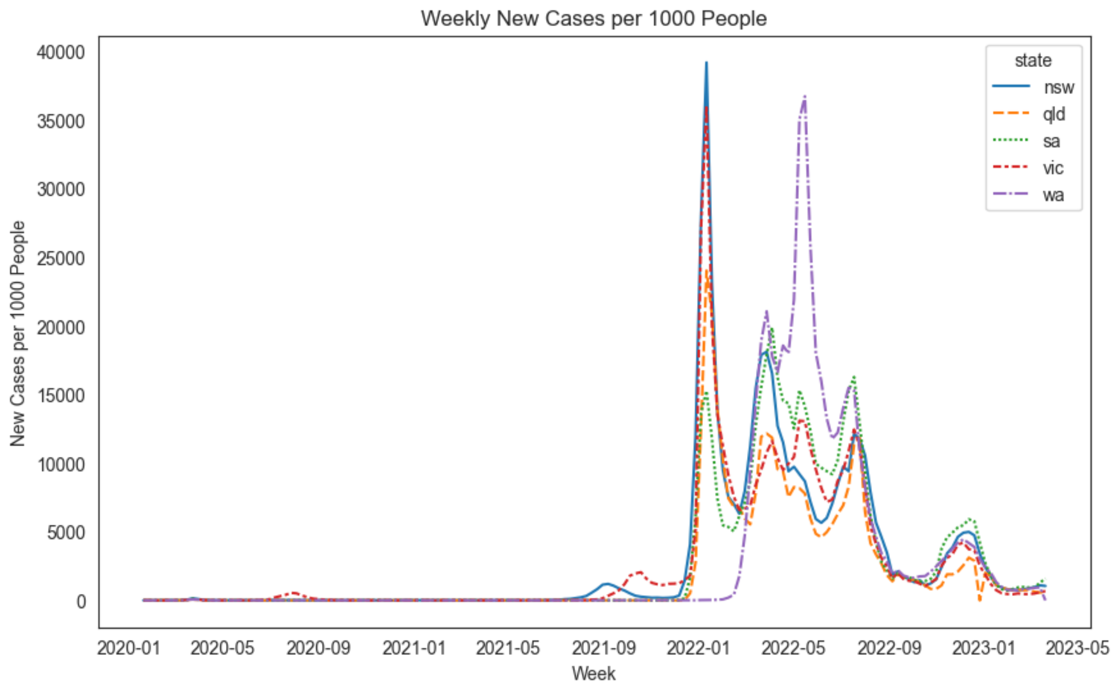
The number of new instances was normalised by population, and a weekly calendar-based history graph was generated. From the commencement of the epidemic until September 9, 2022, the graphs below indicate the weekly number of new cases per 100,000 inhabitants in each state. The trend of new cases per 1,000 persons in each state differs from the trend of total new cases in each state, as can be seen. Western Australia had the most new cases per 1,000 persons, followed by NSW. The number of new cases per 1,000 persons climbed dramatically in all states during the second wave in mid-2020.

[22]:
```
population = pd.read_csv('AustralianBureauofStatistics.csv')
# rename the column unnamed:0 to state
population.rename(columns={'Unnamed: 0': 'state'}, inplace=True)
# remove the last 6 rows
population = population[:-6]

# rename the second column to population by index
population.rename(columns={population.columns[1]: 'population_k'}, inplace=True)

# map state names to their abbreviations
state_abbr = {'new south wales': 'nsw', 'victoria': 'vic', 'queensland': 'qld',
              'western australia': 'wa', 'south australia': 'sa'}
population['state'] = population['state'].str.lower().replace(state_abbr)

# print the updated dataframe
population
```

[22]:
```
  state  population_k  Change over previous year ('000)  \
0   nsw        8153.6                              59.8
1   vic        6613.7                              65.7
2   qld        5322.1                             104.4
3    sa        1820.5                              17.3
4    wa        2785.3                              35.4

   Change over previous year (%)
0                            0.7
1                            1.0
2                            2.0
3                            1.0
4                            1.3
```

[23]:
```
# Merge population and new_cases dataframes
merged_df = pd.merge(week_new, population, on='state')
# drop the last 2 column
merged_df.drop(merged_df.columns[-2:], axis=1, inplace=True)

# Calculate new_cases per capita
```

```
merged_df['new_cases_per_capita'] = merged_df['new'] /␣
↪merged_df['population_k'] * 1000

# Convert date column to datetime format
merged_df['date'] = pd.to_datetime(merged_df['date'])

# Group by week and state
grouped_df = merged_df.groupby([pd.Grouper(key='date', freq='W-MON'), 'state']).
↪sum()

# Reset index and unstack the state column
standardized_df = grouped_df.reset_index().pivot(index='date', columns='state',␣
↪values='new_cases_per_capita')
# Plot the graph
sns.set_style('white')
plt.figure(figsize=(10, 6))
sns.lineplot(data=standardized_df)
plt.title('Weekly New Cases per 1000 People')
plt.xlabel('Week')
plt.ylabel('New Cases per 1000 People')
plt.show()
```



```
[24]: standardized_df
```

[24]:
```
state                nsw         qld          sa         vic          wa
date
2020-01-20      0.000000    0.000000    0.000000    0.000000    0.000000
2020-01-27      0.122645    0.375792    1.098599    0.453604    0.000000
2020-02-03      0.000000    0.563687    0.000000    0.000000    0.000000
2020-02-10      0.000000    0.000000    0.000000    0.000000    0.000000
2020-02-17      0.000000    0.563687    0.549300    0.000000    0.359028
…                    …           …           …           …           …
2023-02-20    802.712912  742.188234  976.105466  461.466350  817.865221
2023-02-27    878.507653  756.844103  933.809393  456.023104  858.076329
2023-03-06    965.340463  626.632344  940.400989  501.837096  914.084659
2023-03-13   1092.155612  572.894158 1289.206262  598.757125  942.447851
2023-03-20   1050.210950  602.581688 1586.377369  675.416182    0.000000

[166 rows x 5 columns]
```

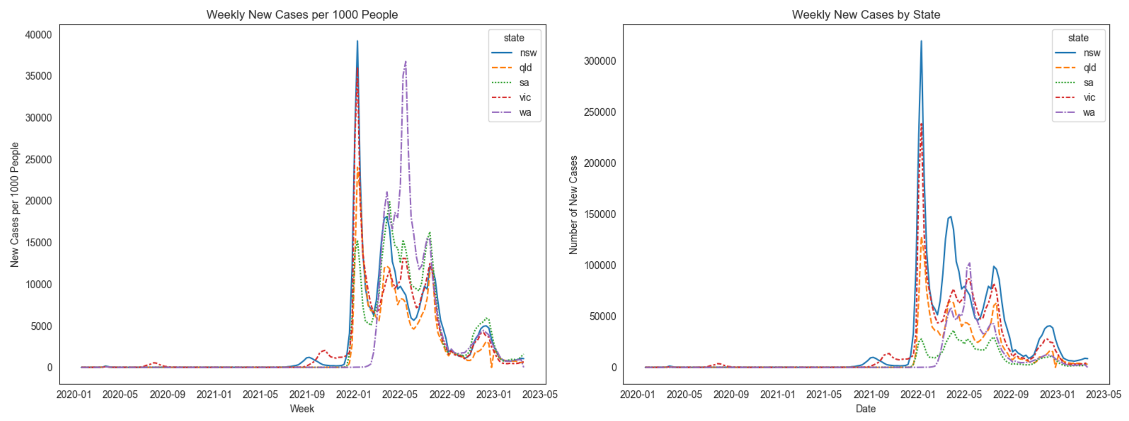
[25]:
```
#Create separate dataframes for new cases and deaths
new_cases =week_new [['date', 'state', 'new']]
# Create a pivot table for new cases
new_cases_pivot = new_cases.pivot(index='date', columns='state', values='new')
```

[26]:
```
new_cases_pivot
```

[26]:
```
state        nsw   qld    sa   vic    wa
date
2020-01-20     0     0     0     0     0
2020-01-27     1     2     2     3     0
2020-02-03     0     3     0     0     0
2020-02-10     0     0     0     0     0
2020-02-17     0     3     1     0     1
…            …     …     …     …     …
2023-02-20  6545  3950  1777  3052  2278
2023-02-27  7163  4028  1700  3016  2390
2023-03-06  7871  3335  1712  3319  2546
2023-03-13  8905  3049  2347  3960  2625
2023-03-20  8563  3207  2888  4467     0

[166 rows x 5 columns]
```

[27]:
```
# Create a subplot with 1 row and 2 columns
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(16, 6))

# Plot the new cases per capita by state in the left subplot
sns.lineplot(data=standardized_df, ax=axs[0])
axs[0].set_title('Weekly New Cases per 1000 People')
axs[0].set_xlabel('Week')
axs[0].set_ylabel('New Cases per 1000 People')
```

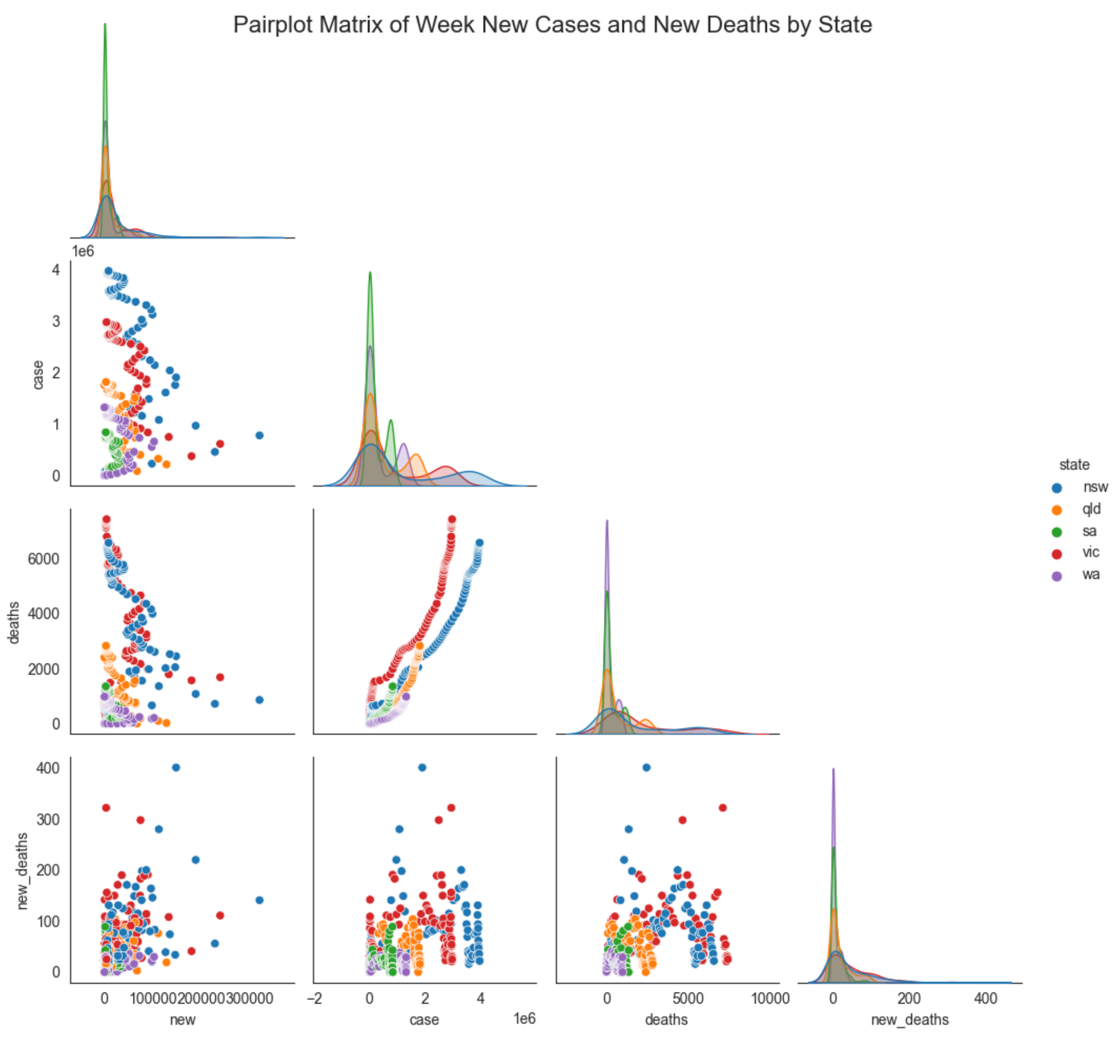
```
# Plot the new cases by state with the custom color palette in the right subplot
sns.lineplot(data=new_cases_pivot, ax=axs[1])
axs[1].set_title('Weekly New Cases by State')
axs[1].set_xlabel('Date')
axs[1].set_ylabel('Number of New Cases')
plt.tight_layout()
plt.show()
```



0.0.5 Relationship Between Number of New Cases and Deaths

In this section, the relationship between the number of new cases and deaths in each state is investigated. The graphs below show the scatterplots of the number of new cases against the number of new deaths in each state from the beginning of the pandemic up to September 9, 2022. We can observe that there is a positive correlation between the number of new cases and new deaths in all states.

[28]:
```
# Create pairplot with hue = 'state'
g = sns.pairplot(week_new, hue='state', corner=True)
# Add a title to the pairplot
plt.suptitle('Pairplot Matrix of Week New Cases and New Deaths by State',␣
 ↪fontsize=16)
plt.show()
```



The pairplot matrix shows the relationship between the number of new cases, total case, total deaths and new deaths in each state. Each scatter plot represents the relationship between new cases and new deaths in a specific state, with the hue indicating the state. From the bottom left plt there is a positive correlation between the number of new cases and new deaths in all states. This suggests that as the number of new cases increases, it can be expected to see a corresponding increase in new deaths. Additionally, the plot reveals that Victoria and New South Wales had the highest number of new cases and new deaths during the pandemic, followed by Queensland and Western Australia. South Australia had the lowest number of new cases and new deaths, indicating that the state was relatively successful in controlling the spread of the virus.

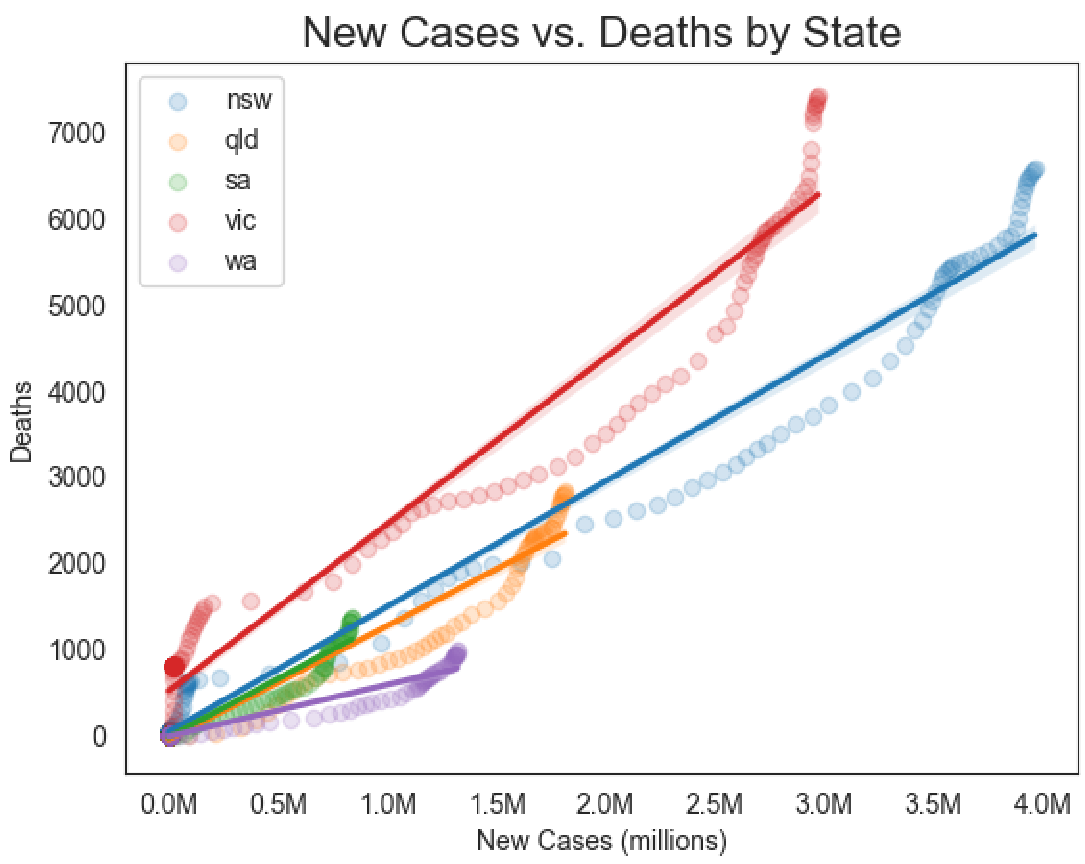
[29]:
```
new_deaths = week_new [['date', 'state', 'new_deaths']]
# Create a pivot table for new deaths
new_deaths_pivot = new_deaths.pivot(index='date', columns='state',␣
 ↪values='new_deaths')
```

[30]:
```
# Calculate the correlation matrix
correlation_matrix = new_cases_pivot.corrwith(new_deaths_pivot)
correlation_matrix
```

[30]:
```
state
nsw    0.649281
qld    0.624834
sa     0.493110
vic    0.499509
wa     0.722223
dtype: float64
```

NSW and WA show the greatest correlations between new cases and new deaths, with correlation values of 0.649 and 0.722, respectively. QLD and VIC similarly have positive correlations between new cases and new fatalities, but the coefficients are slightly lower, at 0.625 and 0.500, respectively. SA has the least positive association between new cases and new deaths, with a value of 0.493. Overall, the correlation coefficients indicate that there is a positive association between the number of new cases and the number of new fatalities in each state, but the strength of the relationship differs by state. However, it is crucial to note that correlation does not always imply causation, and other factors could be impacting the association between new cases and new deaths.

[31]:
```
import matplotlib.ticker as mtick

# Define a function to format x-axis labels in millions
def millions(x, pos):
    'The two args are the value and tick position'
    return '{:.1f}M'.format(x*1e-6)

# Create scatter plot with regression lines for each state
for state in week_new['state'].unique():
    state_data = week_new[week_new['state'] == state]
    sns.regplot(x='case', y='deaths', data=state_data, label=state,␣
 ↪scatter_kws={'alpha':0.2})

# Format x-axis labels in millions
formatter = mtick.FuncFormatter(millions)
plt.gca().xaxis.set_major_formatter(formatter)
plt.title('New Cases vs. Deaths by State', fontsize=16)
plt.xlabel('New Cases (millions)')
plt.ylabel('Deaths')
plt.legend()
plt.show()
```



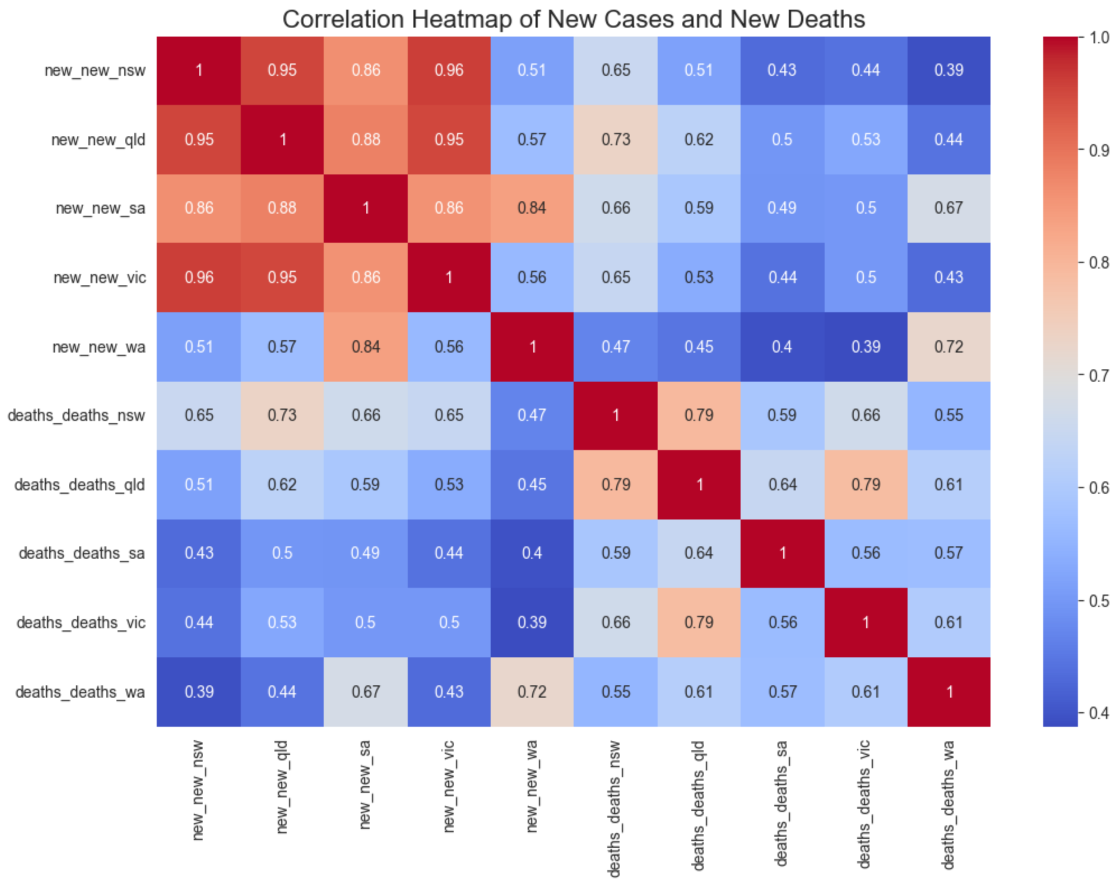
The scatter plot with regression lines shows the relationship between the number of new cases and the number of deaths in each state. The slope of the regression line (which represents the strength of the relationship) varies between states, suggesting that there may be differences in how the two variables are related across different regions. The plot also highlights some outliers in the data, where the number of deaths is much higher or lower than what would be expected based on the number of new cases. These outliers may be worth investigating further to understand what factors could be contributing to the differences in mortality rates between different regions.

[32]:
```
#  Pearson correlation coefficient between the total number of new cases and␣
 ↪deaths for each state across all dates
new_cases_total = new_cases_pivot.sum(axis=0)
new_deaths_total = new_deaths_pivot.sum(axis=0)

np.corrcoef(new_cases_total, new_deaths_total)[0, 1].round(2)
```

[32]:
```
0.91
```

This is a single correlation coefficient that represents the association between the total number of new cases and deaths across all states, and it is quite strong at 0.91. This shows that the number of new cases and deaths are closely connected, which is not surprising given that the number of new cases is directly proportional to the number of new fatalities. It is crucial to remember, however, that correlation does not always imply causation, and other factors could be influencing the link between new cases and new deaths.

[34]:
```
# Rename columns in new_cases_pivot and new_deaths_pivot
new_cases_pivot.columns = [f"new_{col}" for col in new_cases_pivot.columns]
new_deaths_pivot.columns = [f"deaths_{col}" for col in new_deaths_pivot.columns]

# Merge the two tables
merged = pd.concat([new_cases_pivot, new_deaths_pivot], axis=1)

# Create a heatmap
fig, ax = plt.subplots(figsize=(12, 8))
sns.heatmap(data=merged.corr(), annot=True, cmap="coolwarm")
plt.title('Correlation Heatmap of New Cases and New Deaths', fontsize=16)
plt.show()
```



	new_new_nsw	new_new_qld	new_new_sa	new_new_vic	new_new_wa	deaths_deaths_nsw	deaths_deaths_qld	deaths_deaths_sa	deaths_deaths_vic	deaths_deaths_wa
new_new_nsw	1	0.95	0.86	0.96	0.51	0.65	0.51	0.43	0.44	0.39
new_new_qld	0.95	1	0.88	0.95	0.57	0.73	0.62	0.5	0.53	0.44
new_new_sa	0.86	0.88	1	0.86	0.84	0.66	0.59	0.49	0.5	0.67
new_new_vic	0.96	0.95	0.86	1	0.56	0.65	0.53	0.44	0.5	0.43
new_new_wa	0.51	0.57	0.84	0.56	1	0.47	0.45	0.4	0.39	0.72
deaths_deaths_nsw	0.65	0.73	0.66	0.65	0.47	1	0.79	0.59	0.66	0.55
deaths_deaths_qld	0.51	0.62	0.59	0.53	0.45	0.79	1	0.64	0.79	0.61
deaths_deaths_sa	0.43	0.5	0.49	0.44	0.4	0.59	0.64	1	0.56	0.57
deaths_deaths_vic	0.44	0.53	0.5	0.5	0.39	0.66	0.79	0.56	1	0.61
deaths_deaths_wa	0.39	0.44	0.67	0.43	0.72	0.55	0.61	0.57	0.61	1

The correlation table shows the correlation coefficients between the number of new cases and deaths in each state, as well as the correlations between deaths and new cases in different states. Correlation coefficients range from -1 to 1, with values close to 1 indicating a strong positive correlation, values close to -1 indicating a strong negative correlation, and values close to 0 indicating no correlation.

With values ranging from 0.394 to 0.794, the connection between new cases and deaths is frequently positive. This shows that, in general, as the number of new cases increases in a state, so does the number of deaths. The strength of the association varies among states, with Queensland having the strongest correlation (0.794) and Western Australia having the worst (0.468).

There is apparently some association between deaths and new cases in other states. The number of new cases in New South Wales, for example, is positively connected with the number of deaths in Queensland (0.731), South Australia (0.661), and its own state (0.649). It is crucial to emphasize, however, that correlation does not always imply causality. While the connection between new cases and deaths shows a link between the two, it does not necessarily imply that an increase in new cases causes an increase in deaths. Other factors, such as demographic disparities, healthcare infrastructure, and public health policies, may also play a role in the observed patterns.

0.0.6 Conclusion

In conclusion, this analysis examined COVID-19 data from five major Australian states and discovered that NSW had the largest number of new cases and deaths. During the second wave in mid-2020, we saw a strong increase in the number of new cases and deaths, followed by a reduction towards the end of the year. In all states, a positive link between the number of new cases and the number of new deaths are discovered. The results of this study can help policymakers devise effective methods to control the spread of COVID-19 in Australia.

The limitations of this study include the fact that it only analyzes COVID-19 data from five major Australian states, which may not be representative of the entire country. The study also only examines data up to December 2020, which means that the most recent trends in COVID-19 cases and deaths are not accounted for. Additionally, this study does not consider any other variables that may influence the relationship between new cases and deaths, such as demographic factors, healthcare infrastructure, and public health policies. Moreover, the data used in this study may have some errors or omissions, which could affect the accuracy of the results. Finally, it is important to note that correlation does not necessarily imply causation, and the results of this study should be interpreted with caution.

Recommendations for further investigation include including the percentage of the population that has been vaccinated, noting when each state enacted COVID-19 limits, calculating the average daily increase in cases, determining the association between the number of confirmed cases and demographic and socioeconomic data, and using time series analysis techniques to predict future case counts. These analyses can help with forecasting and planning.

[1]:
```
# import os
# os.environ['PATH'] = 'C:/Users/E/AppData/Local/Programs/MiKTeX/miktex/bin/x64;
 ↪' + os.environ['PATH']
```

[2]:
```
# !jupyter nbconvert --to pdf D:
 ↪\2023S1\programing_for_ds\assignment\data\assignment1v6.ipynb
```

```
[NbConvertApp] Converting notebook
D:\2023S1\programing_for_ds\assignment\data\assignment1v6.ipynb to pdf
```

```
[NbConvertApp] Support files will be in assignment1v6_files\
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Making directory .\assignment1v6_files
[NbConvertApp] Writing 110551 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | b had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 1077017 bytes to
D:\2023S1\programing_for_ds\assignment\data\assignment1v6.pdf
```