30759307_a1

September 9, 2019

Name-Dishi Jain Assignment-Data Science FIT5145 Student ID-30759307

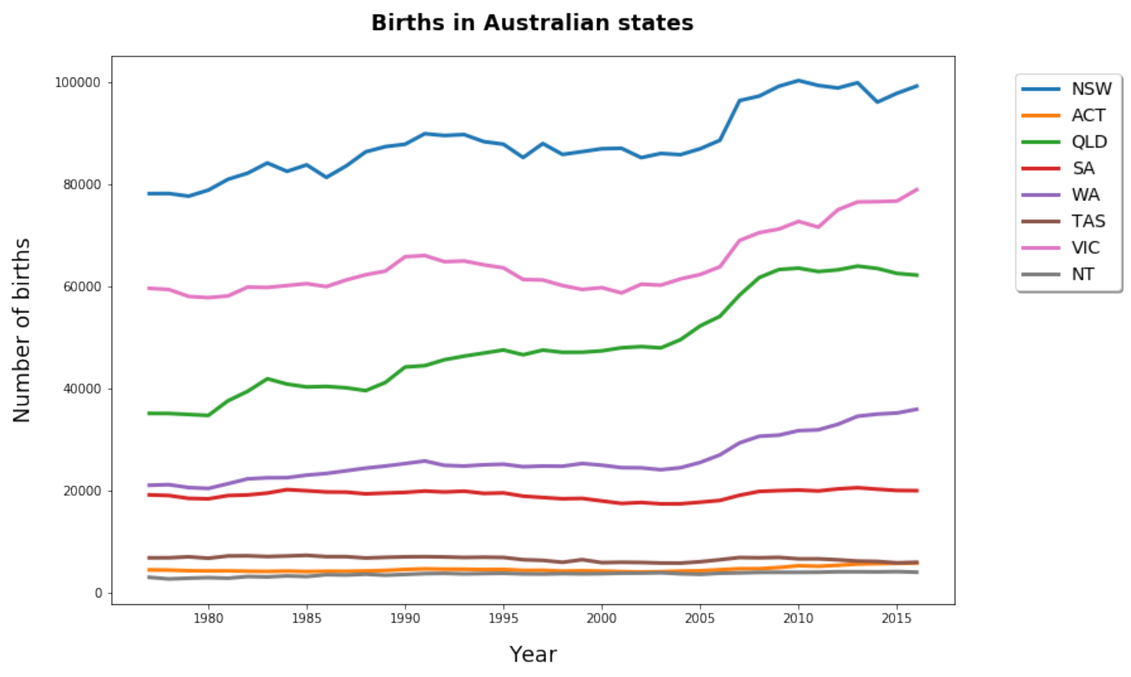
0.1 Task A: Investigating Natural Increase in Australia's population

A1. Investigating the Births, Deaths and TFR Data

1. Using Python, plot the number of births recorded in each state/territory for different Australian states over different years.

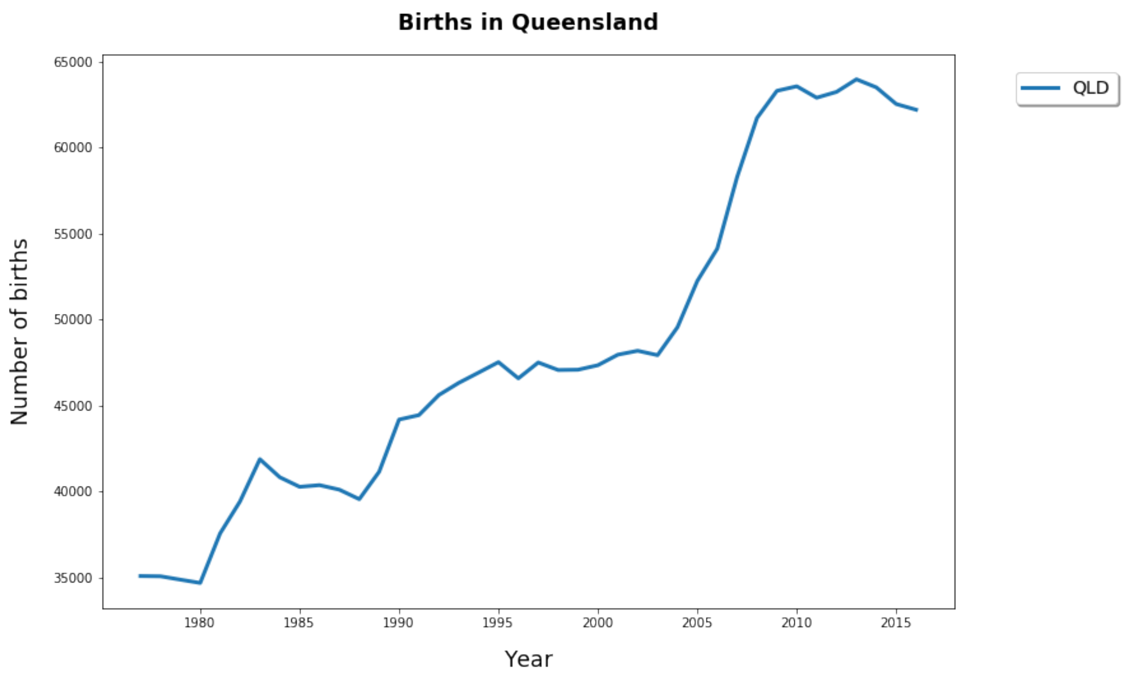
[14]:
```
import pandas as pd
births=pd.read_csv('Births.csv')
births = births.set_index('Year')
```

[15]:
```
#A1-1
import matplotlib.pyplot as plt
plt.figure(figsize=(12, 8))

for i in births.columns:
    plt.plot(births.index,births[i],label=i,linewidth=3)

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Number of births",labelpad=15,color='black',fontsize='xx-large')
plt.title("Births in Australian states",fontsize='xx-large',fontweight␣
 ↪=600,pad=20)

leg=plt.legend(bbox_to_anchor=(1.05, 1, 1,0.), loc=2,ncol=1, borderaxespad=1.
 ↪,fontsize=14,shadow=True,fancybox=True)

plt.show()
```



a. Describe the trend in number of births for Queensland and Tasmania for the period 1977 to 2016?

[16]:
```
#A1-1a

import matplotlib.pyplot as plt
plt.figure(figsize=(12, 8))



for i in births.columns:
    if i=='QLD':
        plt.plot(births.index,births[i],label=i,linewidth=3)

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Number of births",labelpad=15,color='black',fontsize='xx-large')
plt.title("Births in Queensland",fontsize='xx-large',fontweight =600,pad=20)

#plt.legend(bbox_to_anchor=(0., 1.02, 1., 1), loc=3,ncol=4, mode="expand",␣
 ↪fontsize=14,borderaxespad=2.5,shadow=True)
leg=plt.legend(bbox_to_anchor=(1.05, 1, 1,0.), loc=2,ncol=1, borderaxespad=1.
 ↪,fontsize=14,shadow=True,fancybox=True)

plt.show()
```

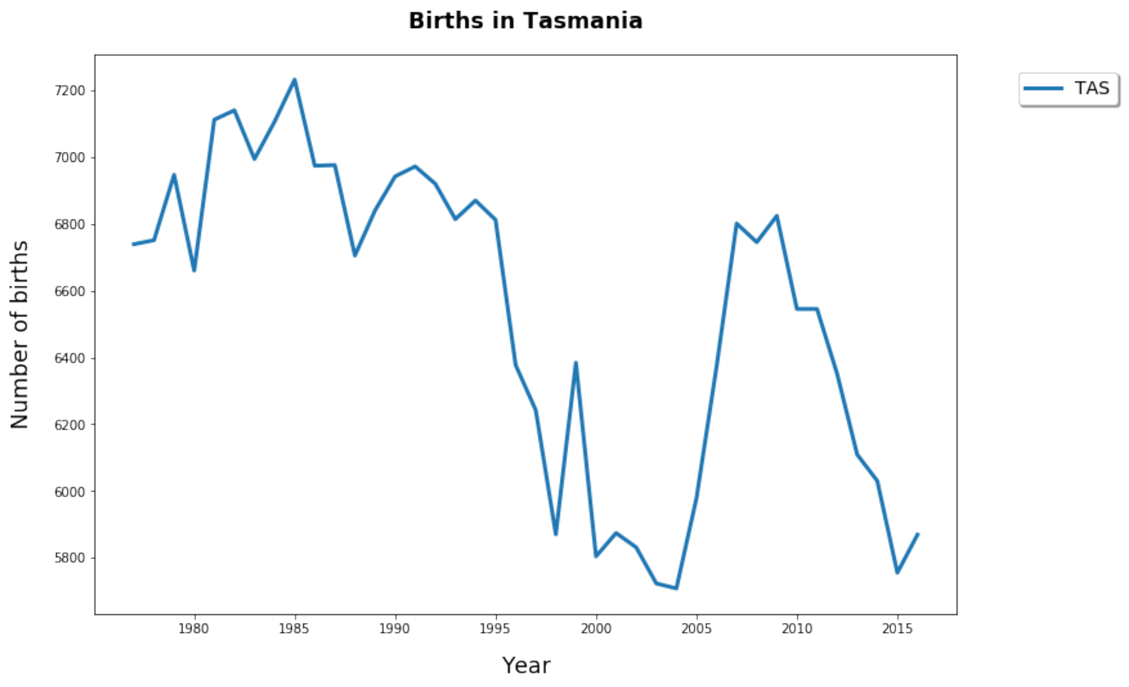


TREND -

The above line graph illustrates the trend between the Births in Queensland over the time period of 1977 to 2016. The trend observed in the above line graph is an increasing trend. Over the time many fluctuations are observed which are illustrated in the line graph. The maximum value of birth is recorded in the year 2016 with a value of around 65,000 and minimum value is observed in the year 1980 with a value of around 35,000 From the year 1980 a sharp increase in the number of births is seen till approximately 1984. A sudden drop to approximately 40,000 in the number of births is then observed in the year of approximately 1988. The line graph then increases gradually from the years 1988 with births of 40,000 till the year of around 2004 with births of around 50,000. After 2004 a huge steep upwards is observed till the year of approximately 2010 where number of births is around 65,000 From the year 2010 the value of number of births remains almost constant till the year 2016.

[17]:
```
#A1-1a

import matplotlib.pyplot as plt
plt.figure(figsize=(12, 8))



for i in births.columns:
    if i=='TAS':
        plt.plot(births.index,births[i],label=i,linewidth=3)

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Number of births",labelpad=15,color='black',fontsize='xx-large')
plt.title("Births in Tasmania",fontsize='xx-large',fontweight =600,pad=20)
```

```
leg=plt.legend(bbox_to_anchor=(1.05, 1, 1,0.), loc=2,ncol=1, borderaxespad=1.
↪,fontsize=14,shadow=True,fancybox=True)

plt.show()
```

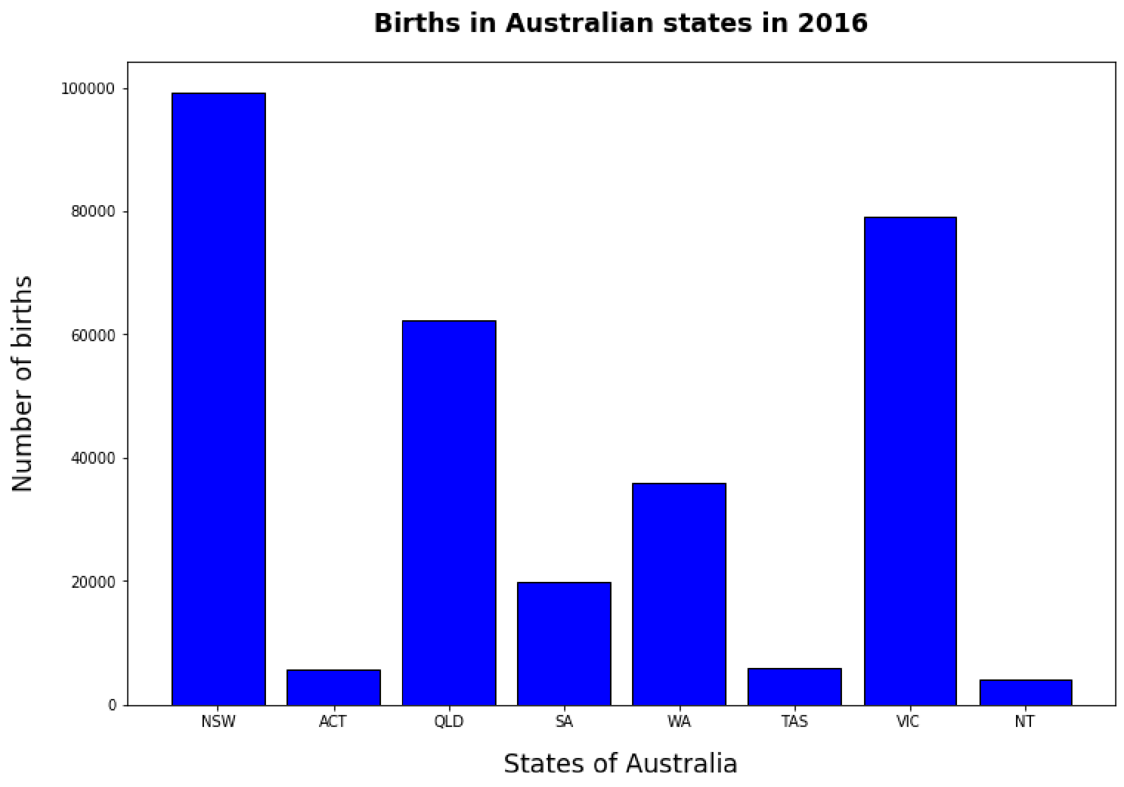


TREND -

The above line graph illustrates the trend between the Births in Tasmania over the time period of 1977 to 2016. The trend observed in the above line graph is a fluctuating trend over time. The maximum value of births is seen in the year around 1985 with a value of 7,200 approximately and th eminimum is seen in the year 2004 with a value of around 5,600. From the year 1977 to around 1985 we can observe fluctuations three times ranging from births with a value between 6,600 and 7,200. After around 1985, a sudden drop is observed till the year 1988 approximately, with a value of 6,700 approximately. Eventually after 1985 an increase is observed till 1995, after which the trend starts to reduce till 2005. After 2005 a sudden increase is seen in the births till the year 2010 after which births suddenly drop till the year 2015 with a vlaue of around 5,700

b. Draw a bar chart to show the number of births in each Australian state in 2016.

[18]:
```
#A1-1b

plt.figure(figsize=(12, 8))

plt.bar(births.columns,births.loc[2016],width=0.
↪8,align='center',color='blue',edgecolor='k',linewidth=1)

plt.xlabel("States of Australia",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Number of births",labelpad=15,color='black',fontsize='xx-large')
plt.title("Births in Australian states in 2016",fontsize='xx-large',fontweight␣
 ↪=600,pad=20)

plt.show()
```



A1

2. We will now investigate the trend in the total number of births over different years. For this, you will need to aggregate the total number of births registered in Australia by year.

[19]:
```
birth_sum_yearwise=births.sum(axis=1)
birth_sum_yearwise.head()
```

[19]:
```
Year
1977    226954
1978    226359
1979    223370
1980    223664
1981    230920
dtype: int64
```

a. Fit a linear regression using Python to the above aggregated data (i.e., total number of births registered in Australia over time) and plot the linear fit.

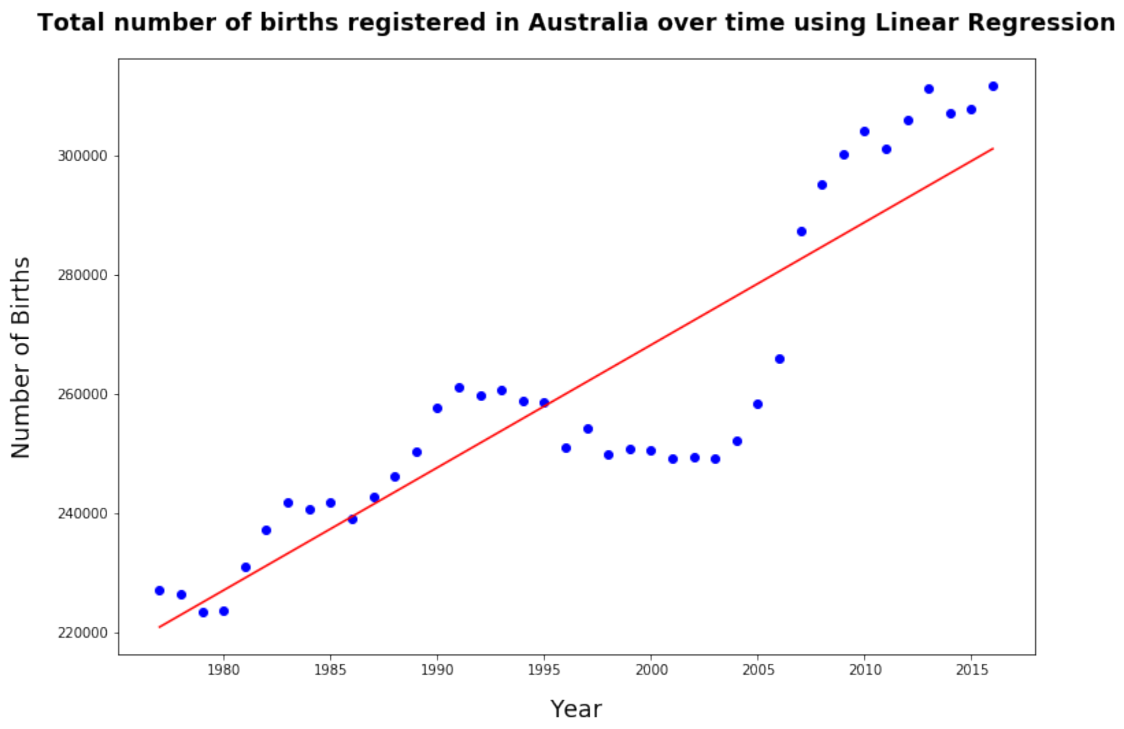
[20]:
```
#A1-2a
import pandas as pd
lp = pd.DataFrame(birth_sum_yearwise)
lp.reset_index(inplace = True)
lp.rename(columns = {0 : 'Birth Sum'} , inplace = True)
```

[21]:
```
X = lp.iloc[:, 0:1].values
y = lp.iloc[:, 1].values
```

[22]:
```
# Fitting Linear Regression to the dataset
import numpy as np
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.linear_model import LinearRegression
linn = LinearRegression()

linn.fit(X, y)
y_pred = linn.predict(X)
rmse = np.sqrt(mean_squared_error(y,y_pred))
print(rmse)
r2 = r2_score(y,y_pred)
print(r2)
```

```
11928.82804501033
0.7983479933953936
```

[24]:
```
# Visualising the Linear Regression results
plt.figure(figsize=(12, 8))

plt.scatter(X, y, color = 'blue')

plt.plot(X, linn.predict(X), color = 'red')

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Number of Births",labelpad=15,color='black',fontsize='xx-large')
plt.title("Total number of births registered in Australia over time using␣
 ↪Linear Regression",fontsize='xx-large',fontweight =600,pad=20)


plt.show()
```



b. Does it look like a good fit to you? Identify the period time having any unusual trend(s) in your plot

#A1-2b

The linear regression above is a good fit to predict the numbers of Birth Sum in Australia over time. However, it's not the best fit. We can still predict the numbers in a better way by plotting other models like polynomial regression. This Linear regression is a good fit because the number of points i.e. (x,y) coordinates plotted on the graph are more in quantity and also are close to each other. This helps in plotting the line for linear progression easy and more accurate.

The time period having an unusal trend is between the years 1988 and 2007. During these years the scatter points plotted are far away from the other scatter points and hence become the cause of increased error percentage for the graph.

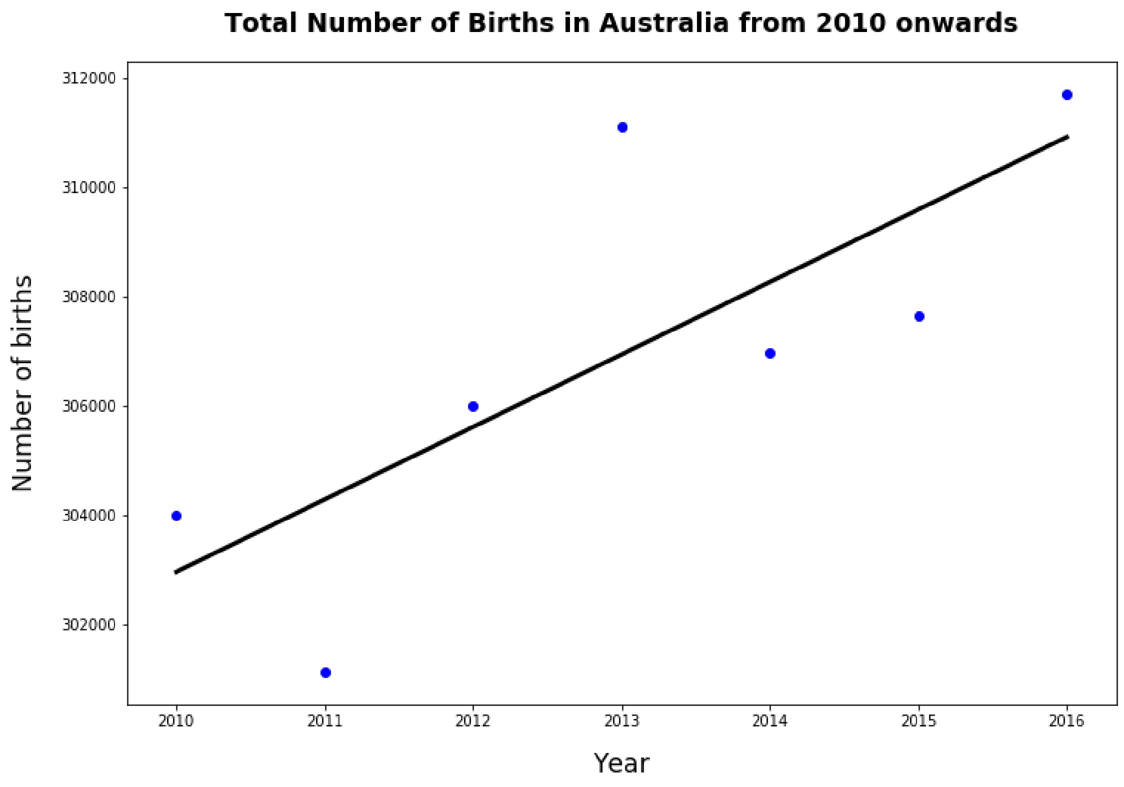
c. Use the linear fit to predict the total births in Australia for the years 2050 and 2100.

[26]:
```
births_in_2050 = linn.predict([[2050]])
births_in_2100 = linn.predict([[2100]])

print(births_in_2050)
print(births_in_2100)
```

```
[370945.74399625]
[473754.24305816]
```

d. Instead of fitting the linear regression to all of the data, try fitting it to just the most recent data points (say from 2010 onwards). How is the fit? Which model would give better predictions of future population of Australia do you think and why?

[29]:
```
#A1-2d
from scipy.stats import linregress
plt.figure(figsize=(12, 8))

birth_sum_yearwise_2010=births[births.index>=2010].sum(axis=1)
slope, intercept, r_value, p_value, std_err = linregress(births[births.
 ↪index>=2010].index,birth_sum_yearwise_2010)
line = [slope*xi + intercept for xi in births[births.index>=2010].index]
plt.plot(births[births.index>=2010].index,line,'k-', linewidth=3)
plt.scatter(births[births.index>=2010].index,birth_sum_yearwise_2010,color='b')

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Number of births",labelpad=15,color='black',fontsize='xx-large')
plt.title("Total Number of Births in Australia from 2010␣
 ↪onwards",fontsize='xx-large',fontweight =600,pad=20)

plt.show()
```



About the Linear Regression Fit-
The above fit of Linear regression for the number of births in Australia over the time period of 2010 to 2016 is not a very good fit. The scatter points available to plot are few, and hence a linear regression is plotted using only few scatter points. This increases the chances of errors for future

predictions. The scatter points are also not very close to each other, and hence will not result in the best plotting of linear regression.

For predictions for the future we should choose a model which is formed of a good number of scatter points which are again of good value i.e. close to each other. Hence the model which would give better predictions of future population of Australia will be the first one i.e. the one where year ranges from 1977 to 2016. This model will hence include more data points which are close to each other and hence will give better predictions.

e. Challenge: Can you think of a better model than linear regression to fit to all of the data to capture the trend in the number of births.

i. Describe the model you suggested and explain why it is better suited for this task.

Yes. A better model can be plotted rather than the Linear Regression. We can plot a Polynomial Regression for better predictions in the number of births in Australia

The model that I am suggesting is a Polynomial Regression. A polynomial regression is a model which is just an extended version of a Linear Regression Model. In a Polynomial Regression we plot a line graph between the independent and dependent variables. Independent means the x value and dependent means the y value on the graph. The polynomial regression hence shows the relationship between the two and models it with an nth degree. A polynomial regression provides the best approximation for the relationship between the variables. This is better suited for this task as the error percentage observed in the polynomial regression is less when compared to the linear regression.

In the Linear Regression the RMSE error is 11928.82804501033 and the R2 error is 0.7983479933953936 In the Polynomial Regression the RMSE error is 8811.816095913402 and the R2 error is 0.7983479933953936 Hence Polynomial is better.

[30]:
```python
lp = pd.DataFrame(birth_sum_yearwise)
lp.reset_index(inplace = True)
lp.rename(columns = {0 : 'Birth Sum'} , inplace = True)
birth_sum_yearwise.head()
```

[30]:
```
Year
1977    226954
1978    226359
1979    223370
1980    223664
1981    230920
dtype: int64
```

[31]:
```python
lp.head()
```

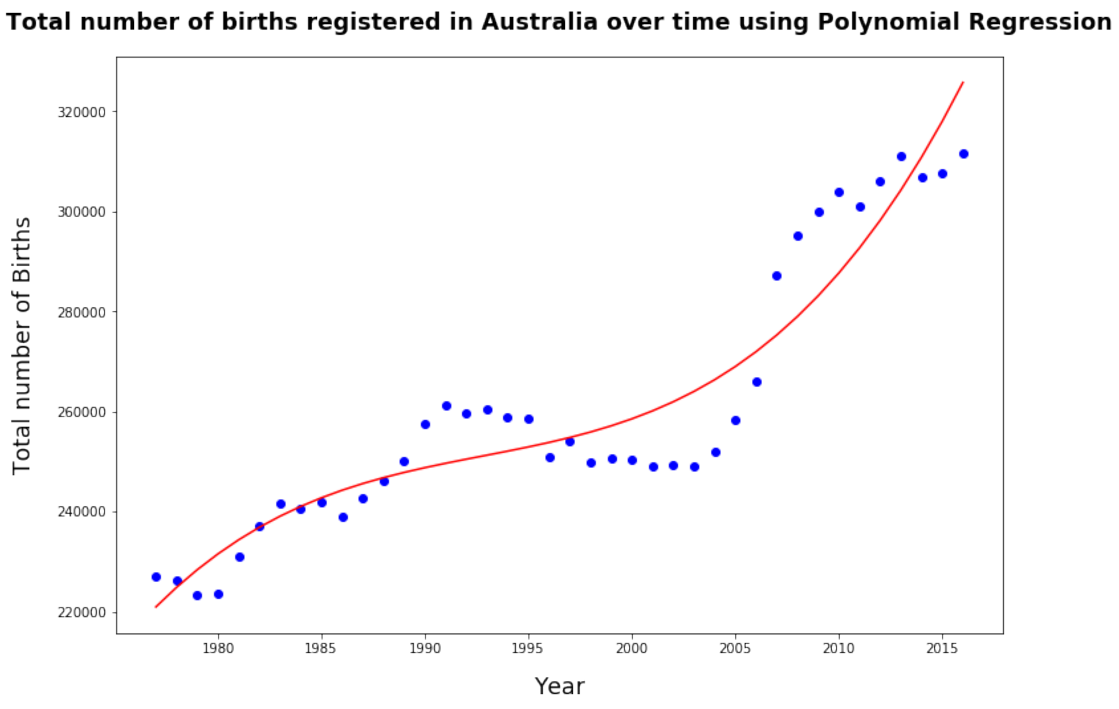
[31]:
```
   Year  Birth Sum
0  1977     226954
1  1978     226359
2  1979     223370
3  1980     223664
4  1981     230920
```

[32]:
```python
X = lp.iloc[:, 0:1].values
y = lp.iloc[:, 1].values
```

[33]:
```python
# Fitting Polynomial Regression to the dataset
import numpy as np
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures

poly = PolynomialFeatures(degree = 4)
X_poly = poly.fit_transform(X)

poly.fit(X_poly, y)
lin2 = LinearRegression()
lin2.fit(X_poly, y)
y_poly_pred = lin2.predict(X_poly)
rmse = np.sqrt(mean_squared_error(y,y_poly_pred))
print(rmse)
r2 = r2_score(y,y_poly_pred)
print(r2)
```

```
8811.816095913402
0.8899632281444942
```

[34]:
```python
# Visualising the Polynomial Regression results
plt.figure(figsize=(12, 8))

plt.scatter(X, y, color = 'blue')

plt.plot(X, lin2.predict(poly.fit_transform(X)), color = 'red')

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Total number of␣
 ↪Births",labelpad=15,color='black',fontsize='xx-large')
plt.title("Total number of births registered in Australia over time using␣
 ↪Polynomial Regression",fontsize='xx-large',fontweight =600,pad=20)

plt.show()
```



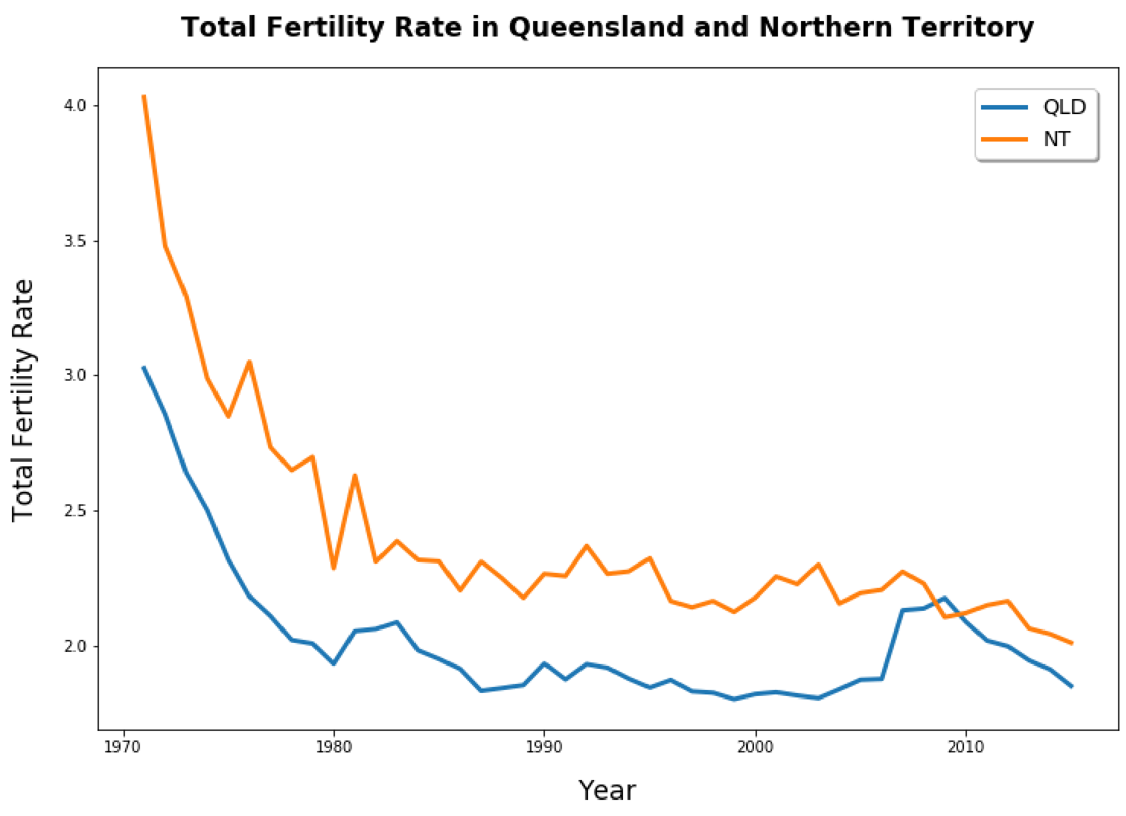
ii. Use your model to predict the total births for the years 2050 and 2100.

[35]:
```
# Predicting a new result with Polynomial Regression
birth_2050_poly = lin2.predict(poly.fit_transform([[2050]]))
birth_2100_poly = lin2.predict(poly.fit_transform([[2100]]))
print(float(birth_2050_poly))
print(float(birth_2100_poly))
```

```
1148774.1656742096
6049884.466032028
```

3. Inspect the data on Total Fertility Rate (TFR.csv) for Queensland and Northern Territory.

[36]:
```
#A1-3
tfr=pd.read_csv('tfr.csv')

plt.figure(figsize=(12, 8))

for i in tfr:
    if i=='Year':
        pass
    elif i == 'QLD' or i == 'NT':
        plt.plot(tfr['Year'],tfr[i],label=i,linewidth=3)

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Total Fertility Rate",labelpad=15,color='black',fontsize='xx-large')
plt.title("Total Fertility Rate in Queensland and Northern␣
↪Territory",fontsize='xx-large',fontweight =600,pad=20)

leg=plt.legend(loc=1, borderaxespad=1.,fontsize=14,shadow=True,fancybox=True)

plt.show()
```



a. What was the minimum value for TFR recorded in the dataset for Queensland and when did that occur? What was the corresponding TFR value for Northern Territory in the same year?

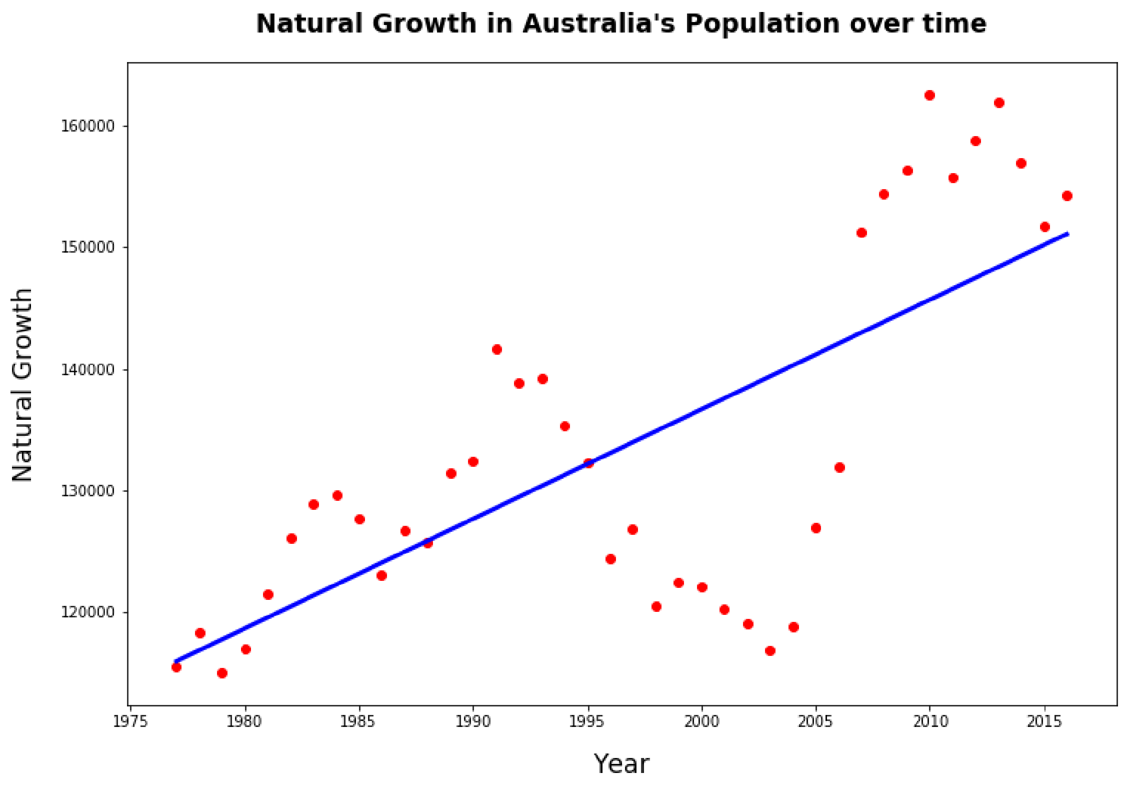
[37]:
```
#A1-3a
tfr[['Year','QLD','NT']][tfr['QLD']==tfr['QLD'].min()]
```

[37]:
```
    Year  QLD     NT
28  1999  1.8  2.123
```

4. Next, plot the natural growth in Australia's population over different years. For this, you will need to aggregate the total births and deaths by year. (HINT: Natural growth in a population is the difference between the total numbers of births and deaths in a population, for instance, Natural Growth of Australia's Population = Total Births in Australia - Total Deaths in Australia)

[38]:
```
#A1-4
plt.figure(figsize=(12, 8))
```

```
deaths=pd.read_csv("Deaths.csv")
deaths = deaths.set_index("Year")
natural_growth_yearwise=[]
death_sum_yearwise=deaths.sum(axis=1)

for i in birth_sum_yearwise.index:
    natural_growth_yearwise.append(birth_sum_yearwise[i]-death_sum_yearwise[i])

slope, intercept, r_value, p_value, std_err = linregress(births.
 ↪index,natural_growth_yearwise)
line = [slope*xi + intercept for xi in births.index]
plt.plot(births.index,line,'b-', linewidth=3)
plt.scatter(births.index,natural_growth_yearwise,color='r')

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Natural Growth",labelpad=15,color='black',fontsize='xx-large')
plt.title("Natural Growth in Australia's Population over␣
 ↪time",fontsize='xx-large',fontweight =600,pad=20)

plt.show()
```



a. Describe the trend in natural growth in Australian population over time using linear regression?

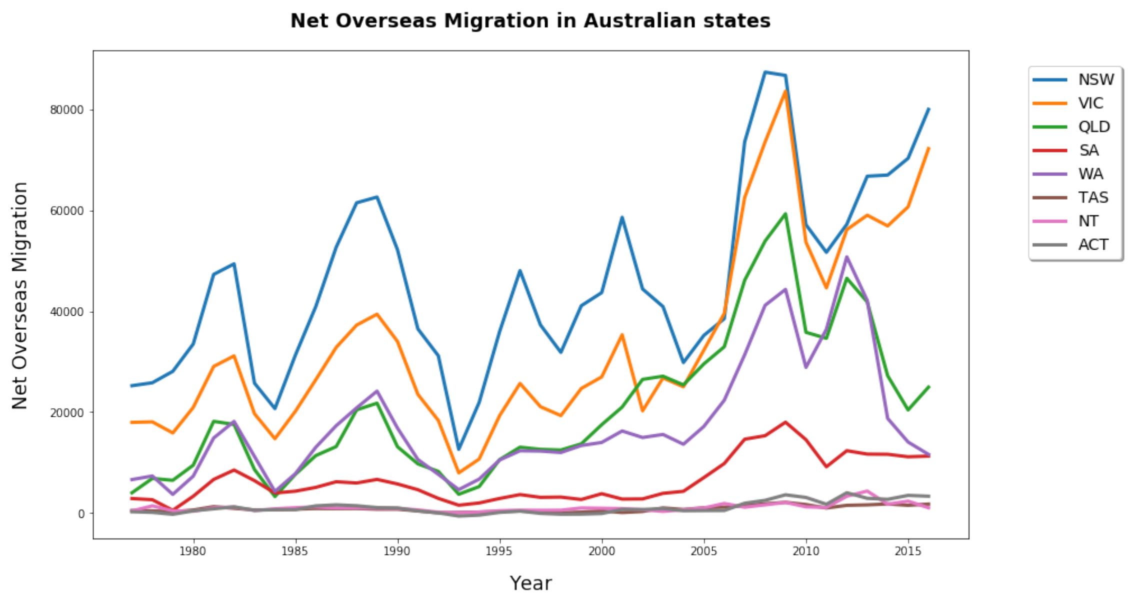
TREND -

The above Linear Regression shows an increasing trend in the value of Natural Growth in Australia over the time period of 1977 to 2016. The maximum value of the natural growth can be seen in the year 2010 with a value of approximately 160,000. The minimum value however can be seen in the year 1979 with a value of around 110,000. The linear regression can be used to predict future values in the natural growth. In the above graph the scatter points for the year between 1977 and 1997 are a good fit as they are close to each other and hence would give a good fit. From the year 1997 till 2005, the scatter points are plotted far away from each other which will result in a poor prediction. From the year 2006 onwards the scatter points range from approximately 130,000 to 160,000 as the value of natural growth. An unusual trend is observed from the years 1997 to 2005. This is visible from the plot clearly as the scatter points are far away from the linear regression and are also dispersed away from each other

A2. Investigating the Migration Data (NOM and NIM)

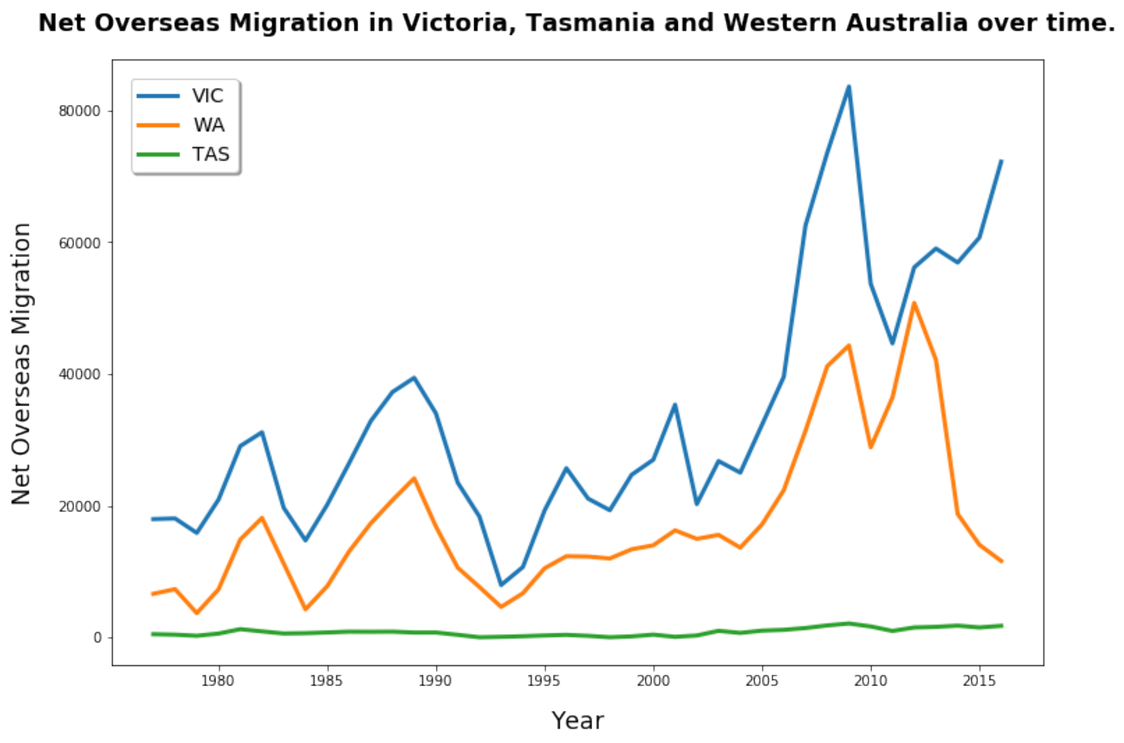
1. Let's look at the Net Overseas Migration (NOM) data in different states over time.

[39]:
```
#A2-1
plt.figure(figsize=(14, 8))

nom=pd.read_csv('NOM.csv')

for i in nom.columns:
    if i == 'Year':
        pass
    else:
        plt.plot(nom['Year'],nom[i],label=i,linewidth=3)

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Net Overseas␣
↪Migration",labelpad=15,color='black',fontsize='xx-large')
plt.title("Net Overseas Migration in Australian␣
↪states",fontsize='xx-large',fontweight =600,pad=20)

leg=plt.legend(bbox_to_anchor=(1.05, 1, 1,0.), loc=2,ncol=1, borderaxespad=1.
↪,fontsize=14,shadow=True,fancybox=True)

plt.show()
```



a. Use Python to plot the NOM to Victoria, Tasmania and Western Australia over time. Explain and compare the trend in all three states (VIC, TAS and WA).

[40]:
```
#A2-1a
plt.figure(figsize=(12, 8))

for i in nom:
    if ((i == "VIC") or (i=="TAS") or (i=="WA")):
        plt.plot(nom['Year'], nom[i],label=i,linewidth=3)


plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Net Overseas␣
 ↪Migration",labelpad=15,color='black',fontsize='xx-large')
plt.title("Net Overseas Migration in Victoria, Tasmania and Western Australia␣
 ↪over time.",fontsize='xx-large',fontweight =600,pad=20)

leg=plt.legend(loc=2,ncol=1, borderaxespad=1.
 ↪,fontsize=14,shadow=True,fancybox=True)

plt.show()
```



TREND -

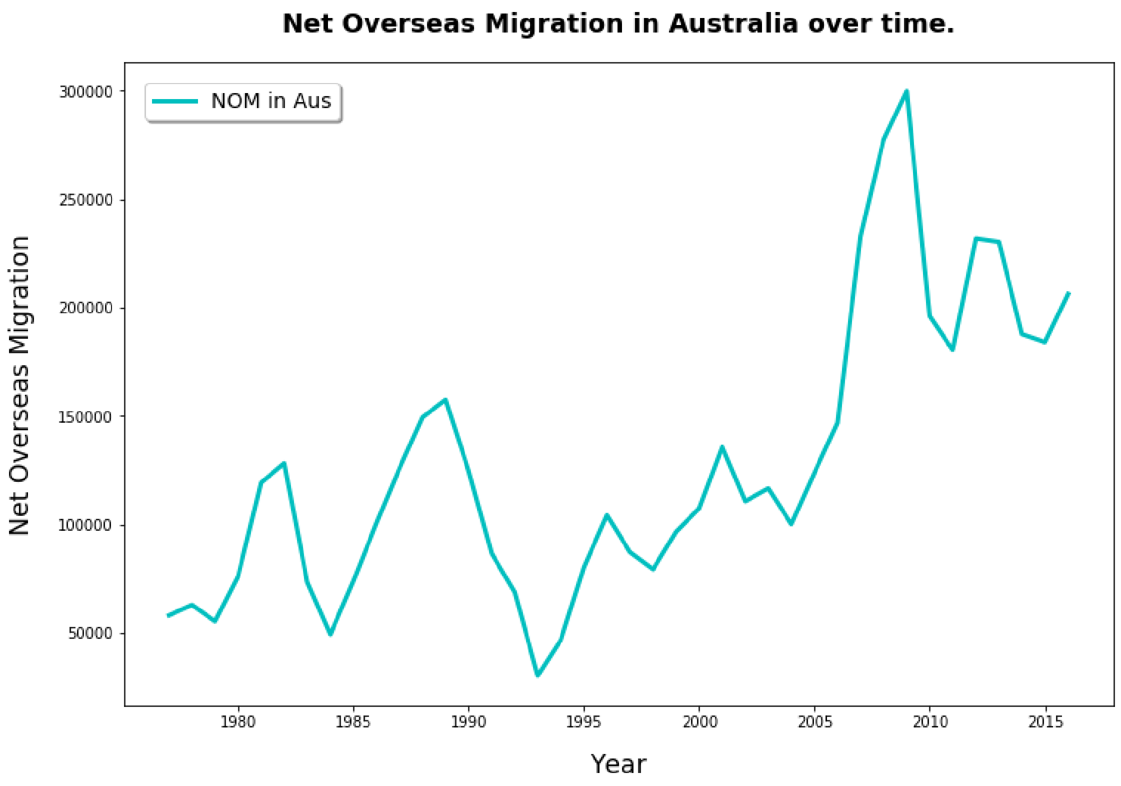
The above line graph depicts the relationship between Year and the Net Overseas Migration in the states of Victoria, Western Australia and Tasmania over the time period of 1977 till 2016. The line graph of Victoria sees many fluctuations over time with a steep increase near the year 2010. During this time the value of NOM reaches the maximum of around 80,000. The minimum value however seen for Victoria is during the year of approximately 1994 with 10,000 as the NOM value. The line graph of Western Australia shows fluctuations over the given time period but remians almost constant during 1994 till 2005 approximately. The maximum value is seen of about 45,000 in the year 2012 approximately and the minimum is seen of about 7,000 during the years 1979, 1984 and 1993 approximately. The line graph of Tasmania however depicts almost a straight line over the time period of 1977 till 2016. No major fluctuations are observed in this graph.

Comparing the three together we can see that the trend of NOM for Victoria and Western Australia is very similar during the initial years but varies after 2010 approximately. For Tasmania we can see that as compared to Victoria and Western Australia, the line is almost a straight line throughout the entire time period.

b. Plot the Net Overseas Migration (NOM) to Australia over time. Do you find the trend strange? Explain the reason to your answer (Hint: You might go online to find contributing factors to this trend).

[41]:
```
#A2-1b
plt.figure(figsize=(12, 8))

nom=nom.set_index('Year')
nom_sum_yearwise=nom.sum(axis=1)
```

```
plt.plot(nom_sum_yearwise.index,nom_sum_yearwise,label="NOM in Aus",␣
 ↪linewidth=3,color='c')

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Net Overseas␣
 ↪Migration",labelpad=15,color='black',fontsize='xx-large')
plt.title("Net Overseas Migration in Australia over time.
 ↪",fontsize='xx-large',fontweight =600,pad=20)

leg=plt.legend(loc=2,ncol=1, borderaxespad=1.
 ↪,fontsize=14,shadow=True,fancybox=True)

plt.show()
```



TREND -

The given trend is for Net Overseas Migration of time in Australia during the years 1977 till 2016. The strange observation about it is the sudden increase in the NOM value after the year of around 2005. From the year 2005 onwards, the value of NOM increases to reach a maximum of around 300,000 in the year 2010. Even after a sudden drop in the following year, the NOM value is still high at around 170,000.

The reason for this strange observation was due to the introduction of many new policies in Australia from 2004 onwards. New education policies in support of international students were also introduced which led more to the increase. Many new government policies were also introduced. Hence a huge increase in the NOM value is observed.

2. Now let's look at the relationship between Net Overseas Migration (NOM) and Net Interstate Migration (NIM).

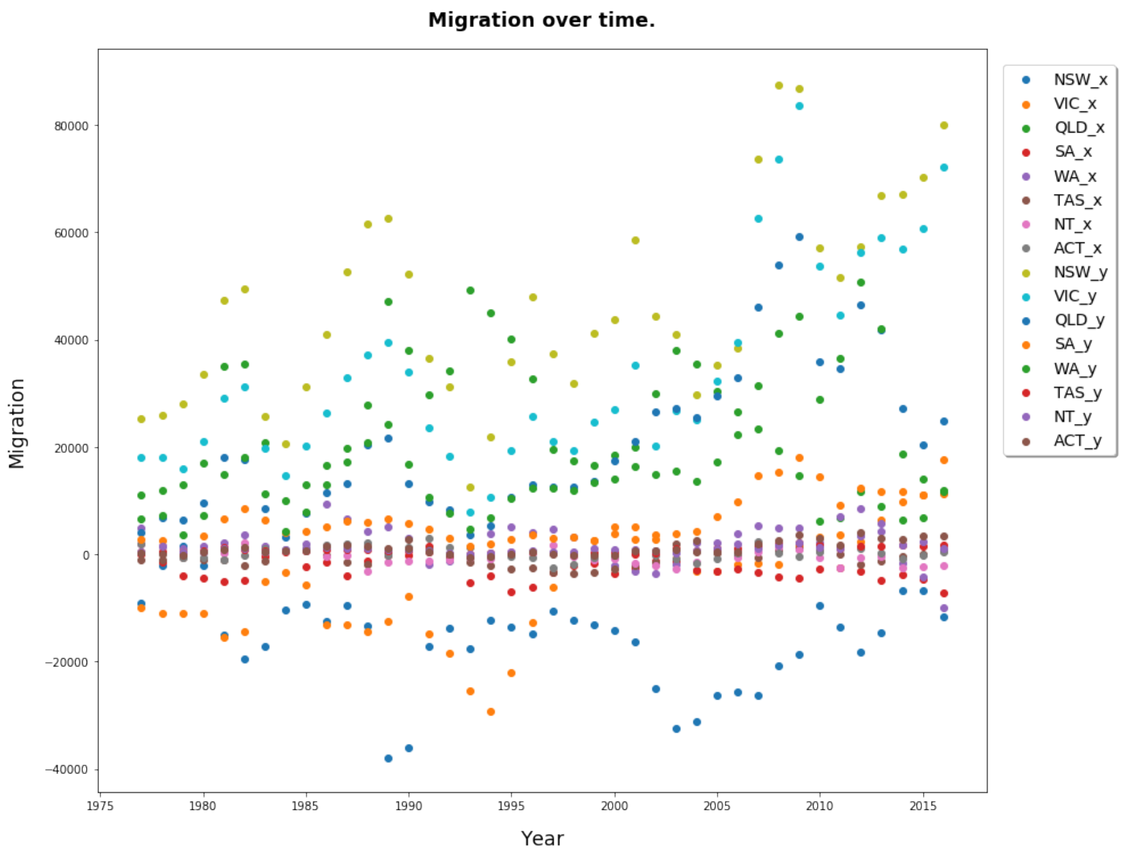
a. Use Python to combine the data from the different files into a single table. The resulting table should contain the NOM and NIM values for each of the states for a given year. What are the first year and last year for the combined data?

[42]:
```
#A2-2a
nim=pd.read_csv('NIM.csv')
nom=pd.read_csv('NOM.csv')
merged=pd.merge(nim,nom,on='Year')

first_year_merged = merged.head(1)['Year']
last_year_merged =  merged.tail(1)['Year']
print(first_year_merged)
print(last_year_merged)
```

```
0    1977
Name: Year, dtype: int64
39    2016
Name: Year, dtype: int64
```

b. Now that you have the data combined, we can see whether there is a relationship between NOM and NIM. Plot the values against each other using scatter plot. Can you see any relationship between NOM and NIM?

[43]:
```
#A2-2b
plt.figure(figsize=(14, 12))

merged = merged.set_index('Year')
for i in merged:
    plt.scatter(merged.index,merged[i],label=i)

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Migration",labelpad=15,color='black',fontsize='xx-large')
plt.title("Migration over time.",fontsize='xx-large',fontweight =600,pad=20)

leg=plt.legend(bbox_to_anchor=(1, 1, 1,0.),loc=2,ncol=1, borderaxespad=1.
 ↪,fontsize=14,shadow=True,fancybox=True)

plt.show()
```



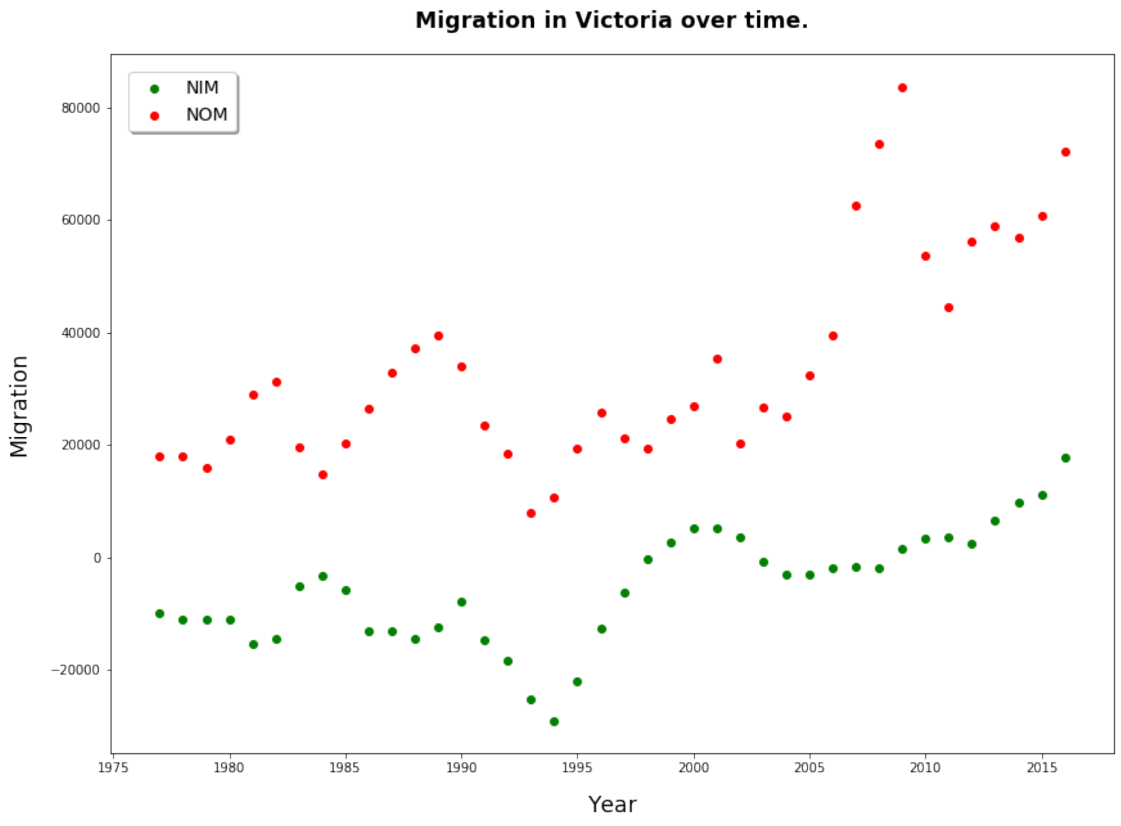
TREND -

The trend between NOM and NIM can be seen over the time period of 1977 till 2016. The value of NIM can be seen as constant line ranging from 1977 to 2016 as compared to NOM over the same time. The NOM values over the time period are seen fluctating. The minimum NOM value is observed in the year 1994 of around 20,000 and the maximum value is seen in the year of around 2010 with a value of approximately 300,000.

c. Try selecting and plotting the data for Victoria only using scatter plot. Can you see a relationship now? If so, explain the relationship.

[44]:
```
#A2-2c
plt.figure(figsize=(14, 10))

plt.scatter(nim['Year'],nim['VIC'],label='NIM',color='g')
plt.scatter(nom['Year'],nom['VIC'],label='NOM',color='r')

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Migration",labelpad=15,color='black',fontsize='xx-large')
plt.title("Migration in Victoria over time.",fontsize='xx-large',fontweight␣
 ↪=600,pad=20)

leg=plt.legend(loc=2,ncol=1, borderaxespad=1.
 ↪,fontsize=14,shadow=True,fancybox=True)
```

```
plt.show()
```



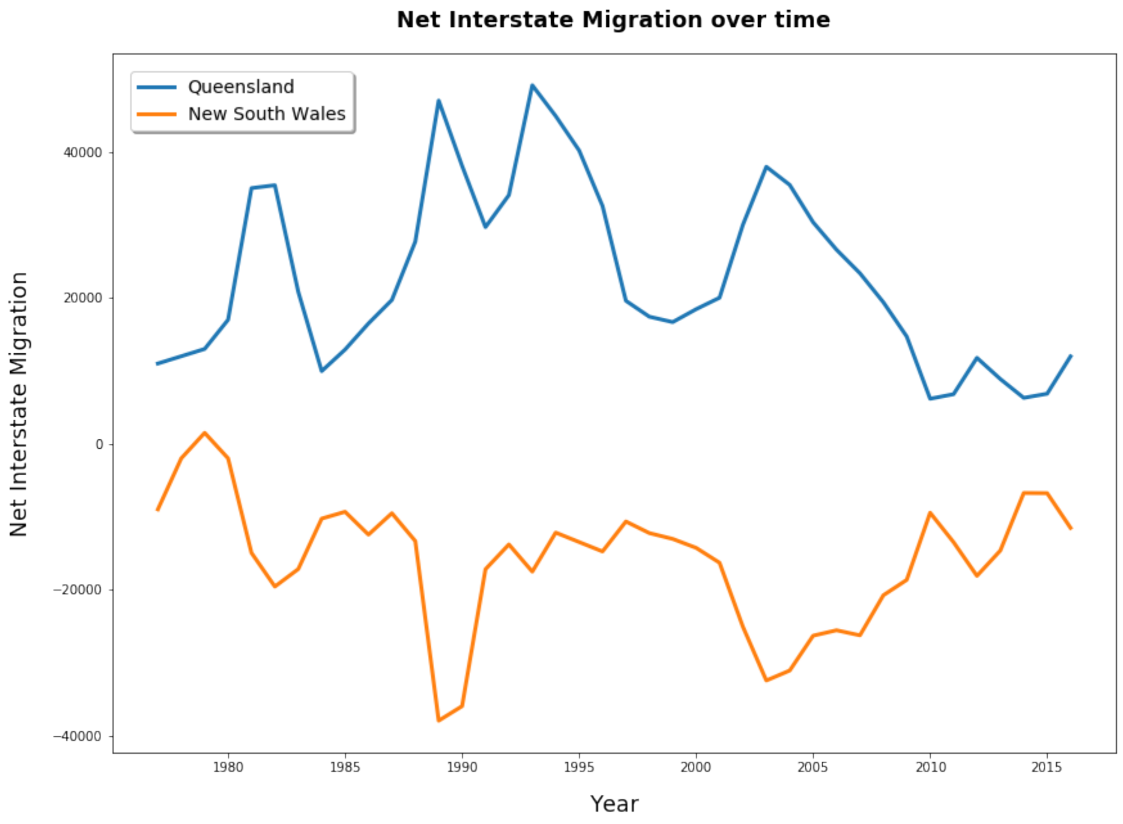
TREND -

The abover graph is a scatter plot depicting the NOM and NIM values in Victoria over the time period of 1977 to 2016. Both the plots are very similar to each other. Whenever a dip is observed in NIM a similar dip in time is observed in NOM. Similar trend is seen whenever an increase is observed during the time period. Both NIM and NOM values suffer a sudden drop in the year 1994 approximately of values -40000 and 50000 respectively. 1995 onwards an increasing trend can be seen for both NIM and NOM.

The NIM and NOM of Victoria is similar to the NIM and NOM os entire Australia. Hence we can say that Victoria contributed the most in terms of NIM and NOM values in the years 1977 to 2016.

d. Finally, plot the Net Interstate Migration (NIM) for Queensland and New South Wales over different years. Note graphs for both QLD and NSW should be on the same plot. Compare the plots for these two states. What can you infer from the trend you see for these two states?

[45]:
```
#A2-2d
plt.figure(figsize=(14, 10))

plt.plot(nim['Year'],nim['QLD'],label="Queensland",linewidth=3)
plt.plot(nim['Year'],nim['NSW'],label="New South Wales",linewidth=3)
plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Net Interstate␣
↪Migration",labelpad=15,color='black',fontsize='xx-large')
plt.title("Net Interstate Migration over time",fontsize='xx-large',fontweight␣
↪=600,pad=20)

leg=plt.legend(loc=2,ncol=1, borderaxespad=1.
↪,fontsize=14,shadow=True,fancybox=True)

plt.show()
```



TREND-

The above line graph depicts the NIM for Queensland and New South Wales for the years 1997 till 2016. From the above graph it is depicted that both the lines almost show the inverse relationship. So whenever value for NIM in Queensland increases the respective value for New South Wales decreses and vice versa. They are almost like a mirror image of each other. During the years 1980 , 1990 and 2004 we can see a sudden increase in Queensland value and a sudden decrease in the New SOuth Wales value of NIM. From 2010 onwards, both the state's NIM values become almost constant.

A3. Visualising the Relationship over Time Now let's look at the relationship between other variables impacting the population size and growth of Australian states/territories over

time. Ensure that you have combined all the data from the different files (Births.csv, Deaths.csv, TFR.csv, NOM.csv and NIM.csv) into a single table.

1. Use Python to build a Motion Chart, that compares the role migration (overseas and interstate) plays towards population growth in each Australia state/territory over time. The motion chart should show the Net Overseas Migration (NOM) on the x-axis, the Net Interstate Migration (NIM) on the y-axis, and the bubble size should show the Total Population Growth. (HINT: A Jupyter notebook containing a tutorial on building motion charts in Python is available here)

[46]:
```
deaths=pd.read_csv("Deaths.csv")
births=pd.read_csv("Births.csv")
tfr=pd.read_csv("TFR.csv")
nim=pd.read_csv("NIM.csv")
nom=pd.read_csv("NOM.csv")
```

[47]:
```
melt_births = pd.melt(births,␣
 ↪id_vars=['Year'],value_vars=['NSW','ACT','QLD','SA','WA','TAS','VIC','NT'],var_name='States
 ↪= 'BIRTHS')
melt_deaths = pd.melt(deaths,␣
 ↪id_vars=['Year'],value_vars=['NSW','ACT','QLD','SA','WA','TAS','VIC','NT'],var_name='States
 ↪= 'DEATHS')
melt_tfr = pd.melt(tfr,␣
 ↪id_vars=['Year'],value_vars=['NSW','ACT','QLD','SA','WA','TAS','VIC','NT'],var_name='States
 ↪= 'TFR')
melt_nim = pd.melt(nim,␣
 ↪id_vars=['Year'],value_vars=['NSW','ACT','QLD','SA','WA','TAS','VIC','NT'],var_name='States
 ↪= 'NIM')
melt_nom = pd.melt(nom,␣
 ↪id_vars=['Year'],value_vars=['NSW','ACT','QLD','SA','WA','TAS','VIC','NT'],var_name='States
 ↪= 'NOM')
```

[48]:
```
melt_nom['NIM'] = melt_nim['NIM']
melt_nom['DEATHS'] = melt_deaths['DEATHS']
melt_nom['BIRTHS'] = melt_births['BIRTHS']
melt_nom['TFR'] = melt_tfr['TFR']
final = melt_nom
```

[49]:
```
final.head()
```

[49]:
```
   Year States    NOM    NIM  DEATHS  BIRTHS    TFR
0  1977    NSW  25236  -9000   42075   78173  2.806
1  1978    NSW  25825  -2000   40121   78190  2.653
2  1979    NSW  28086   1500   39975   77669  2.385
3  1980    NSW  33499  -2000   39799   78859  2.303
4  1981    NSW  47291 -14963   39979   80980  2.125
```

[50]:
```
final['NATURAL GROWTH'] = final['BIRTHS'] - final['DEATHS'] + final['NIM'] +␣
 ↪final['NOM']
```

[51]:
```
final.head()
```

[51]:
```
   Year States    NOM    NIM  DEATHS  BIRTHS    TFR  NATURAL GROWTH
0  1977    NSW  25236  -9000   42075   78173  2.806           52334
1  1978    NSW  25825  -2000   40121   78190  2.653           61894
2  1979    NSW  28086   1500   39975   77669  2.385           67280
3  1980    NSW  33499  -2000   39799   78859  2.303           70559
4  1981    NSW  47291 -14963   39979   80980  2.125           73329
```

[52]:
```
from motionchart.motionchart import MotionChart
import pandas as pd
```

[53]:
```
%%html
<style>
.output_wrapper, .output {
    height:auto !important;
    max-height:1000px;  /* your desired max-height here */
}
.output_scroll {
    box-shadow:none !important;
    webkit-box-shadow:none !important;
}
</style>
```

```
<IPython.core.display.HTML object>
```

[54]:
```
mChart = MotionChart(df = final, key='Year', x='NOM', y='NIM', xscale='linear',␣
 ↪yscale='linear',
                      size='NATURAL GROWTH',category='States')

mChart.to_notebook()
```

```
<IPython.lib.display.IFrame at 0x683d21fb00>
```

2. Run the visualisation from start to end. (Hint: In Python, to speed up the animation, set timer bar next to the play/pause button to the minimum value.) And then answer the following questions:

a. Comment generally on the trend you see in Net Overseas Migration (NOM) and Net Interstate Migration (NIM) overtime. Is there any relationship between the two variables?

TREND -

The trend between NOM and NIM can be seen over the time period of 1977 till 2016. The value of NIM can be seen as constant line ranging from 1977 to 2016 as compared to NOM over the same time. The NOM values over the time period are seen fluctating. The minimum NOM value is observed in the year 1994 of around 20,000 and the maximum value is seen in the year of around 2010 with a value of approximately 300,000. For NIM the maximum and minimum values remain almost similar when compared to NOM.

No there is no relationship between the two variables as the NIM remains almost constant with no fluctuations however the NOM shows many fluctuations over the entire time period.

b. Select VIC and NSW for this question: In which year(s) does VIC have a higher Net Overseas Migration (NOM) than NSW. Please support your answer with a relevant python code and motion chart screenshot.

In the Year - 2006 the NOM for VIC is greater than that of NSW

[55]:
```
mChart = MotionChart(df = final, key='Year', x='States', y='NOM',␣
↪xscale='linear', yscale='linear',
                     size='NATURAL GROWTH',category='States')

mChart.to_notebook()
```

```
<IPython.lib.display.IFrame at 0x683e9ec080>
```

c. Which state has the highest Net Interstate Migration most of the years (for the period 1977 to 2016)?

[56]:
```
final1 = final.pivot(index = 'Year', columns = "States", values = "NIM").
↪reset_index()
max_state = final1.idxmax(axis = 1).value_counts()
max_state1 = max_state.reset_index()
max_state1
```

[56]:
```
  index   0
0   QLD  36
1   VIC   3
2    WA   1
```

Hence we can say that QLD i.e. Queensland has a maximum NIM over all the years

0.2 Task B: Exploratory Analysis of Data

B1. Daily number of crimes

1. For each suburb, calculate the number of days that at least 15 crimes have occurred per day. (Hint: your answer should contain all suburbs in the dataset together with a value showing the number of days that at least 15 crimes have happened)

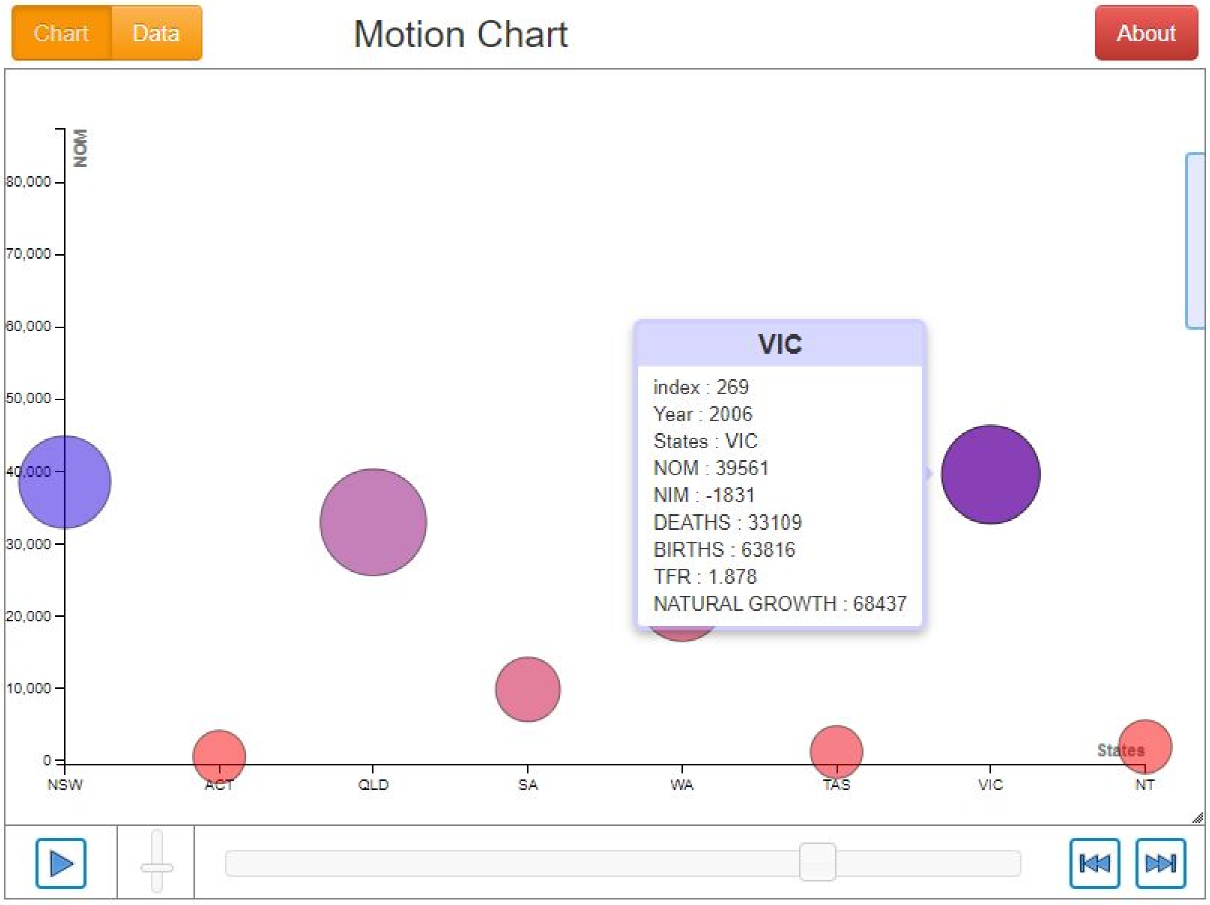
[57]:
```
import pandas as pd
crimes=pd.read_csv("Crime_Statistics_SA_2014_2019.csv")
```

[58]:
```
filt={'Offence Count' : {'count of offence' : lambda x : sum(e>=15 for e in x)}}
groupbySD=crimes.groupby(['Suburb - Incident' , 'Reported Date']).sum()
```

[59]:
```
count_days=groupbySD.groupby('Suburb - Incident',sort=False).agg(filt)
count_days.columns = count_days.columns.droplevel(1)
count_days.rename(columns = {'Offence Count': 'Count of days with crimes␣
↪atleast 15' }, inplace = True)
count_days
```

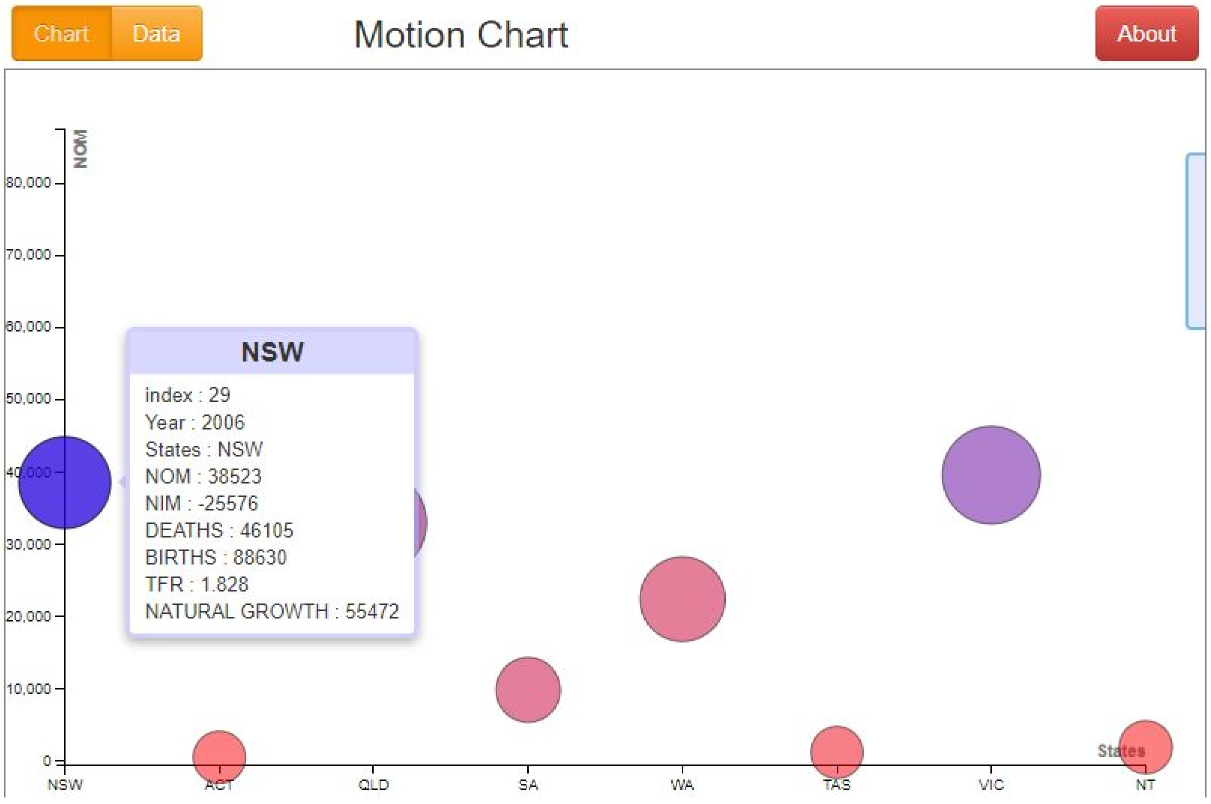
```
C:\Users\dishi\Anaconda3\lib\site-packages\pandas\core\groupby\generic.py:1315:
FutureWarning: using a dict with renaming is deprecated and will be removed in a
```

```
In [1103]: mChart = MotionChart(df = final, key='Year', x='States', y='NOM', xscale='linear', yscale='linear',
                                 size='NATURAL GROWTH',category='States')

            mChart.to_notebook()
```



VIC
index : 269
Year : 2006
States : VIC
NOM : 39561
NIM : -1831
DEATHS : 33109
BIRTHS : 63816
TFR : 1.878
NATURAL GROWTH : 68437

```
In [1103]: mChart = MotionChart(df = final, key='Year', x='States', y='NOM', xscale='linear', yscale='linear',
                                 size='NATURAL GROWTH',category='States')

            mChart.to_notebook()
```



NSW
index : 29
Year : 2006
States : NSW
NOM : 38523
NIM : -25576
DEATHS : 46105
BIRTHS : 88630
TFR : 1.828
NATURAL GROWTH : 55472

```
future version
  return super(DataFrameGroupBy, self).aggregate(arg, *args, **kwargs)
```

[59]:

```
                                    Count of days with crimes atleast 15
Suburb - Incident
ABERFOYLE PARK                                                       0.0
ADDRESS UNKNOWN                                                      0.0
ADELAIDE                                                           877.0
ADELAIDE AIRPORT                                                     0.0
AGERY                                                                0.0
ALAWOONA                                                             0.0
ALBANY                                                               0.0
ALBERT PARK                                                          0.0
ALBERTON                                                             0.0
ALDGATE                                                              0.0
ALDINGA                                                              0.0
ALDINGA BEACH                                                        0.0
ALFORD                                                               0.0
ALLENBY GARDENS                                                      0.0
ALLENDALE EAST                                                       0.0
ALLENDALE NORTH                                                      0.0
ALMA                                                                 0.0
ALTONA                                                               0.0
AMATA                                                                0.0
AMERICAN RIVER                                                       0.0
ANAMA                                                                0.0
ANANGU PITJANTJATJARA YANKUNYTJATJARA                                0.0
ANDAMOOKA                                                            0.0
ANDREWS FARM                                                         0.0
ANGAS PLAINS                                                         0.0
ANGAS VALLEY                                                         0.0
ANGASTON                                                             0.0
ANGLE PARK                                                           0.0
ANGLE VALE                                                           0.0
ANNADALE                                                             0.0
...                                                                  ...
WYNN VALE                                                            0.0
WYOMI                                                                0.0
YACKA                                                                0.0
YAHL                                                                 0.0
YALATA                                                               0.0
YALLUNDA FLAT                                                        0.0
YAMBA                                                                0.0
YANINEE                                                              0.0
YANKALILLA                                                           0.0
YANKANINNA                                                           0.0
YANYARRIE                                                            0.0
```

```
YARANYACKA                                                          0.0
YARDEA                                                              0.0
YATALA VALE                                                         0.0
YATINA                                                              0.0
YATTALUNGA                                                          0.0
YEELANNA                                                            0.0
YELTA                                                               0.0
YINKANIE                                                            0.0
YONGALA                                                             0.0
YONGOLA                                                             0.0
YORKE VALLEY                                                        0.0
YORKETOWN                                                           0.0
YOUNG HUSBAND                                                       0.0
YOUNGHUSBAND                                                        0.0
YOUNGHUSBAND HOLDINGS                                               0.0
YUMALI                                                              0.0
YUNDI                                                               0.0
YUNTA                                                               0.0
ZADOWS LANDING                                                      0.0

[1627 rows x 1 columns]
```

2. Now which suburbs do have at least one day where the daily number of crimes are more than 15. Plot the number of days that at least 15 crimes have occurred for the suburbs you found in this step (step 2) using a bar graph.

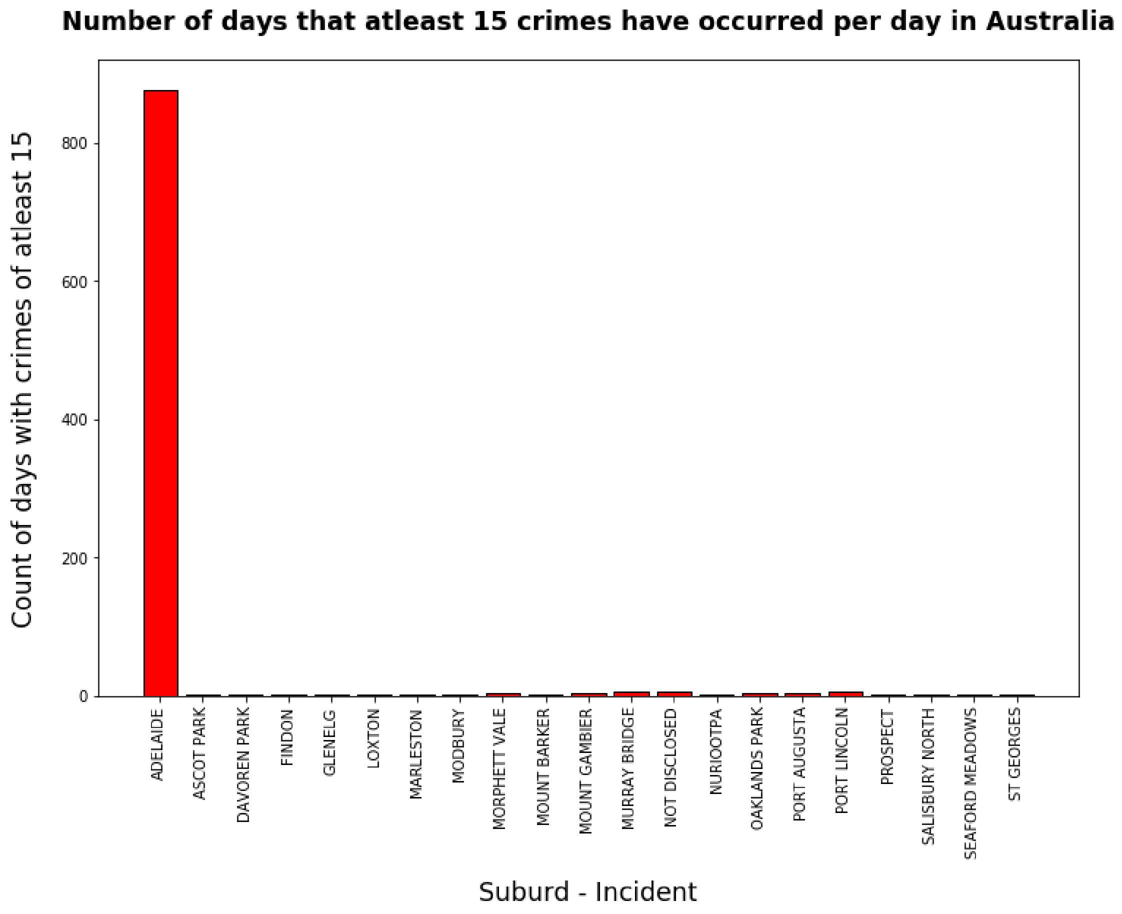
[60]:
```
final_data = count_days[count_days['Count of days with crimes atleast 15']>=1]
final_data.reset_index(inplace = True)
```

[61]:
```
final_data
```

[61]:
```
   Suburb - Incident  Count of days with crimes atleast 15
0           ADELAIDE                                 877.0
1         ASCOT PARK                                   1.0
2       DAVOREN PARK                                   1.0
3             FINDON                                   1.0
4            GLENELG                                   1.0
5             LOXTON                                   1.0
6          MARLESTON                                   1.0
7            MODBURY                                   1.0
8      MORPHETT VALE                                   3.0
9       MOUNT BARKER                                   1.0
10     MOUNT GAMBIER                                   3.0
11     MURRAY BRIDGE                                   5.0
12     NOT DISCLOSED                                   5.0
13         NURIOOTPA                                   1.0
14      OAKLANDS PARK                                  3.0
15      PORT AUGUSTA                                   4.0
16      PORT LINCOLN                                   5.0
```

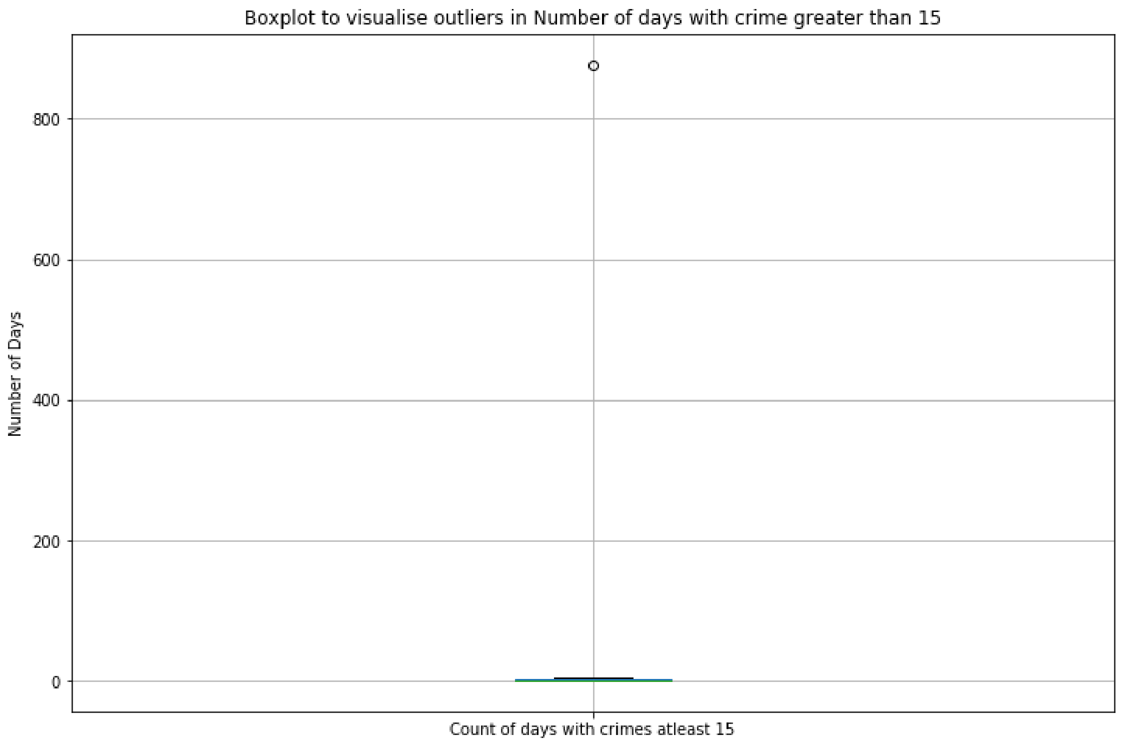
```
17          PROSPECT                                  2.0
18    SALISBURY NORTH                                 1.0
19    SEAFORD MEADOWS                                 1.0
20         ST GEORGES                                 1.0
```

[62]:
```
plt.figure(figsize=(12, 8))

plt.bar(final_data['Suburb - Incident'],final_data['Count of days with crimes␣
 ↪atleast 15'], width=0.8,align='center',color='r',edgecolor='k',linewidth=1)
plt.xlabel("Suburd - Incident",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Count of days with crimes of atleast␣
 ↪15",labelpad=15,color='black',fontsize='xx-large')
plt.title("Number of days that atleast 15 crimes have occurred per day in␣
 ↪Australia",fontsize='xx-large',
          fontweight =600,pad=20)
plt.xticks(rotation=90)
plt.show()
```



3. Use an appropriate graph to visualize and detect outliers (extreme values) on the data from step 2 and remove them. Then, plot the data again using a bar graph.

[63]:
```
plt.figure(figsize=(12,8))
final_data.boxplot(column = 'Count of days with crimes atleast 15')

plt.title('Boxplot to visualise outliers in Number of days with crime greater␣
 ↪than 15')
plt.ylabel('Number of Days')
plt.show()
```



[64]:
```
#Removing outliers

Q1 =  final_data['Count of days with crimes atleast 15'].quantile(0.25)
Q3 = final_data['Count of days with crimes atleast 15'].quantile(0.75)
IQR = Q3 - Q1

outlier_values=final_data[(final_data['Count of days with crimes atleast 15'] <␣
 ↪Q1-1.5*IQR ) |
      (final_data['Count of days with crimes atleast 15'] > Q3+1.5*IQR)]['Count␣
 ↪of days with crimes atleast 15']
```

[65]:
```
outlier_values
```

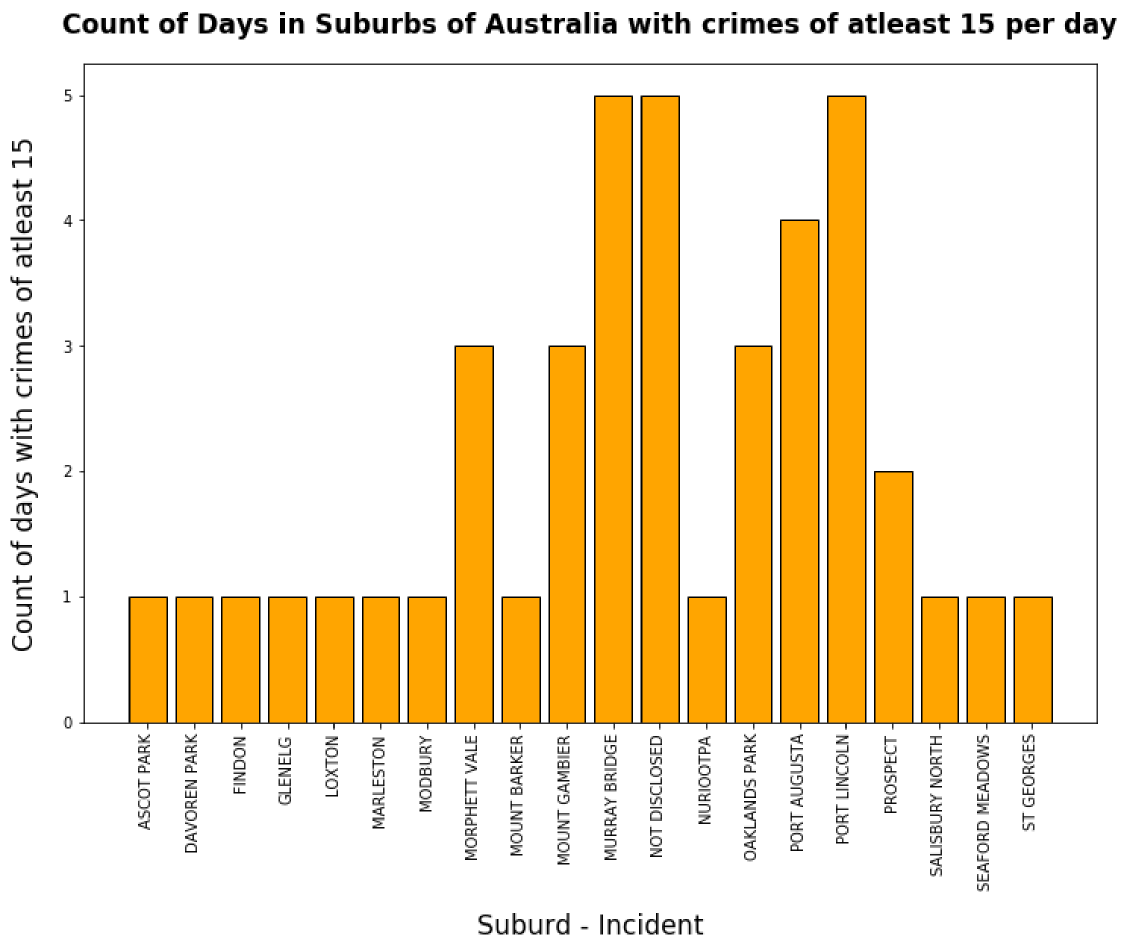
[65]:
```
0    877.0
Name: Count of days with crimes atleast 15, dtype: float64
```

[66]:
```
data_without_outliers = final_data[~final_data['Count of days with crimes␣
↪atleast 15'].isin(outlier_values)]
data_without_outliers.reset_index(drop=True,inplace=True)
```

[67]:
```
data_without_outliers
```

[67]:
```
   Suburb - Incident  Count of days with crimes atleast 15
0         ASCOT PARK                                   1.0
1       DAVOREN PARK                                   1.0
2             FINDON                                   1.0
3            GLENELG                                   1.0
4             LOXTON                                   1.0
5          MARLESTON                                   1.0
6            MODBURY                                   1.0
7      MORPHETT VALE                                   3.0
8       MOUNT BARKER                                   1.0
9      MOUNT GAMBIER                                   3.0
10     MURRAY BRIDGE                                   5.0
11     NOT DISCLOSED                                   5.0
12         NURIOOTPA                                   1.0
13     OAKLANDS PARK                                   3.0
14      PORT AUGUSTA                                   4.0
15      PORT LINCOLN                                   5.0
16          PROSPECT                                   2.0
17   SALISBURY NORTH                                   1.0
18   SEAFORD MEADOWS                                   1.0
19        ST GEORGES                                   1.0
```

[68]:
```
#Plotting data without outliers
from matplotlib.pyplot import colormaps

plt.figure(figsize=(12, 8))

plt.bar(data_without_outliers['Suburb - Incident'],data_without_outliers['Count␣
↪of days with crimes atleast 15'],
        width=0.8,align='center',color='orange',edgecolor='k',linewidth=1)
plt.xlabel("Suburd - Incident",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Count of days with crimes of atleast␣
↪15",labelpad=15,color='black',fontsize='xx-large')
plt.title("Count of Days in Suburbs of Australia with crimes of atleast 15 per␣
↪day",fontsize='xx-large',
          fontweight =600,pad=20)
plt.xticks(rotation=90)
plt.show()
```



4. Compare the bar graphs in step 2 and 3. Which bar graph is easier to interpret? Why?

The two bar graphs plotted of Part 2 and Part3 have different variations. The Part 2 bar graph has outlier values plotted in it. Due to this, the visualisation becomes difficult and hence it is not a good model. We should plot a bar graph without the outliers so that a better image can be seen and we can visualise and compare data easily. The bar chart in Part 3 is plotted after removing al the outliers. Hence it is visually better and easy to understand. We can see the suburbs easily and can visualise the count of days with crimes of atleast 15.

The bar graph in Part 3 is easier to interpret as all the suburbs and day counts are easily visible there. We can visually see it and compare the values in different suburbs.

B2. Challenge: Identify mistakes in data entry

There are some errors in the data entry in one of the columns.

1. Identify the data entry errors and provide possible solutions

There are 3 errors present in the column 'Suburb - Incident'

Error 1 :

There are Null Values present in the dataset in the columns 'Suburb - Incident' and 'Postcode - Incident'. Null values should not be present in the dataset as it would cause problems while manipulating the data in the future. The error was identified using the given python code-

[69]:
```
crimes.isnull().sum()
```

[69]:
```
Reported Date                   0
Suburb - Incident             159
Postcode - Incident           403
Offence Level 1 Description     0
Offence Level 2 Description     0
Offence Level 3 Description     0
Offence Count                   0
dtype: int64
```

[70]:
```python
error1 = crimes[crimes['Suburb - Incident'].isnull() | crimes['Postcode -␣
↪Incident'].isnull()]
error1.head()
```

[70]:
```
      Reported Date        Suburb - Incident Postcode - Incident  \
3176     2014-01-13  NO FIXED PLACE OF ABODE                 NaN
7409     2014-01-31          ADDRESS UNKNOWN                 NaN
10189    2014-11-02          ADDRESS UNKNOWN                 NaN
10190    2014-11-02          ADDRESS UNKNOWN                 NaN
20564    2014-03-25          ADDRESS UNKNOWN                 NaN

        Offence Level 1 Description          Offence Level 2 Description  \
3176      OFFENCES AGAINST PROPERTY  FRAUD DECEPTION AND RELATED OFFENCES
7409      OFFENCES AGAINST PROPERTY  FRAUD DECEPTION AND RELATED OFFENCES
10189  OFFENCES AGAINST THE PERSON          ACTS INTENDED TO CAUSE INJURY
10190  OFFENCES AGAINST THE PERSON            ROBBERY AND RELATED OFFENCES
20564     OFFENCES AGAINST PROPERTY  FRAUD DECEPTION AND RELATED OFFENCES

                   Offence Level 3 Description  Offence Count
3176                Obtain benefit by deception            1.0
7409                Obtain benefit by deception            1.0
10189  Serious Assault not resulting in injury            1.0
10190                   Blackmail and extortion            1.0
20564               Obtain benefit by deception            1.0
```

Possible Solution:

For this data entry error the possible solution is to fill the Null values with a relevant data. With the help of the above code we can see that “Address Unknown” is filled in some cells of the column ‘Suburb - Incident’. Hence for the Null values in the column ‘Suburb - Incident’ we can write ‘Address Unknown’

Error 2 :

With the help of the error1 output we could identify that there are some entries in the dataset where ‘Suburb - Incident’ is given a value but the respective ‘Postcode - Incident’ is Null. This can be checked using the below python code

There are cases where ‘Suburb - Incident’ has a value of ‘Address Unknown/No Fixed Place Of Abode/Not Disclosed’ for which the respective ‘Postcode - Incident’ has a Null Value. But there are also some cases where the ‘Suburb - Incident’ has a name of a Suburb but the respective ‘Postcode - Incident’ has a Null value. This should be corrected.

[71]:
```
error2 = crimes[crimes['Suburb - Incident'].notna() & crimes['Postcode -␣
↪Incident'].isnull()]
error2.head()
```

[71]:
```
      Reported Date        Suburb - Incident Postcode - Incident  \
3176     2014-01-13  NO FIXED PLACE OF ABODE                 NaN
7409     2014-01-31          ADDRESS UNKNOWN                 NaN
10189    2014-11-02          ADDRESS UNKNOWN                 NaN
10190    2014-11-02          ADDRESS UNKNOWN                 NaN
20564    2014-03-25          ADDRESS UNKNOWN                 NaN

       Offence Level 1 Description           Offence Level 2 Description  \
3176     OFFENCES AGAINST PROPERTY  FRAUD DECEPTION AND RELATED OFFENCES
7409     OFFENCES AGAINST PROPERTY  FRAUD DECEPTION AND RELATED OFFENCES
10189  OFFENCES AGAINST THE PERSON          ACTS INTENDED TO CAUSE INJURY
10190  OFFENCES AGAINST THE PERSON           ROBBERY AND RELATED OFFENCES
20564    OFFENCES AGAINST PROPERTY  FRAUD DECEPTION AND RELATED OFFENCES

                   Offence Level 3 Description  Offence Count
3176               Obtain benefit by deception            1.0
7409               Obtain benefit by deception            1.0
10189  Serious Assault not resulting in injury            1.0
10190                  Blackmail and extortion            1.0
20564              Obtain benefit by deception            1.0
```

Possible Solution :

For cases where 'Suburb - Incident' has a value of 'Address Unknown/No Fixed Place Of Abode/Not Disclosed', the respective 'Postcode - Incident' can be filled with 'Unknown' as a value. Hence this can be fixed.

However, with the cases where 'Suburb - Incident' has a suburb name and still the respective 'Postcode - Incident' has a Null value, here we should add the postcode of the suburb. However as we do not have adequate data to fill the postcodes for a suburb name, hence the respective postcodes are also filled with 'Unknown' as a value.

Error 3 -

There are certain Null values in the column 'Suburb - Incident' which have a respective 'Postcode - Incident'. This should not be the case as if the postcode is known of a crime location ,the repective suburb should also be filled. As with just the postcodes, the suburb name cannot be filled, hence for this case we can just fill 'Unknown' in place of Null values in the column 'Suburb - Incident'

[72]:
```
crimes[crimes['Suburb - Incident'].isnull() & crimes['Postcode - Incident'].
↪notna()]
```

[72]:
```
       Reported Date Suburb - Incident Postcode - Incident  \
349965    2018-11-16               NaN              5330.0
355663    2018-08-12               NaN              5540.0
357477    2018-12-15               NaN              5037.0
366653    2019-01-20               NaN              5540.0
376388    2019-02-25               NaN              5723.0
```

```
        Offence Level 1 Description    Offence Level 2 Description  \
349965    OFFENCES AGAINST PROPERTY      THEFT AND RELATED OFFENCES
355663    OFFENCES AGAINST PROPERTY      THEFT AND RELATED OFFENCES
357477    OFFENCES AGAINST PROPERTY       SERIOUS CRIMINAL TRESPASS
366653  OFFENCES AGAINST THE PERSON  ACTS INTENDED TO CAUSE INJURY
376388  OFFENCES AGAINST THE PERSON  ACTS INTENDED TO CAUSE INJURY

       Offence Level 3 Description  Offence Count
349965                Other theft            1.0
355663    Theft from motor vehicle            1.0
357477         SCT - Non Residence            1.0
366653              Assault police            1.0
376388              Common Assault            1.0
```

Possible Solution :

The best solution for this case is to fill in the suburb names. However we do not have adequate data to fill in the suburb name based on just the postcode.

For eg- the postcode 5330 has multiple suburb name as shown below.

Hence the best possible solution is to fill 'Unknown' in the Null Values for the column 'Suburb - Incident'

[73]:
```
crimes[['Suburb - Incident', 'Postcode - Incident']][crimes['Postcode -␣
↪Incident']=='5330.0'].tail(7)
```

[73]:
```
       Suburb - Incident Postcode - Incident
374985          WAIKERIE              5330.0
375132            HOLDER              5330.0
375133            HOLDER              5330.0
375831          WAIKERIE              5330.0
379895          WAIKERIE              5330.0
382912          WAIKERIE              5330.0
384512          WAIKERIE              5330.0
```

2. Use Python to fix the errors.

Fixing Error 1

The Error 1 will be fixed automatically as soon as we put in the correct values for Error 2 and Error 3. Hence Error 1 is a confirmation of fixing all the errors in the dataset. Before fixing the errors ,the null values are -

[74]:
```
crimes.isnull().sum()
```

[74]:
```
Reported Date                    0
Suburb - Incident              159
Postcode - Incident            403
Offence Level 1 Description      0
Offence Level 2 Description      0
Offence Level 3 Description      0
Offence Count                    0
dtype: int64
```

Fixing Error 2

By filling in the value as 'Unknown' for the Null values in the column 'Postcode - Incident', the error will be fixed.

Python code for the same is given below-

[75]:
```
crimes['Postcode - Incident'].fillna('Unknown' , inplace = True)
```

Now we can check whether the error has been fixed or not by using the below python code-

[76]:
```
crimes.isnull().sum()
```

[76]:
```
Reported Date                  0
Suburb - Incident            159
Postcode - Incident            0
Offence Level 1 Description    0
Offence Level 2 Description    0
Offence Level 3 Description    0
Offence Count                  0
dtype: int64
```

Fixing Error 3

By filling in the value as 'Address Unknown' for the Null values in the column 'Suburb - Incident', the error will be fixed. We have chosen to fill 'Address Unknown' out of "Address Unknown/No Fixed Place Of Abode/Not Disclosed" as that makes the most sense.

Python code for the same is given below-

[77]:
```
crimes['Suburb - Incident'].fillna('Address Unknown' , inplace = True)
```

Now we can check whether the error has been fixed or not by using the below python code-

[78]:
```
crimes.isnull().sum()
```

[78]:
```
Reported Date                  0
Suburb - Incident              0
Postcode - Incident            0
Offence Level 1 Description    0
Offence Level 2 Description    0
Offence Level 3 Description    0
Offence Count                  0
dtype: int64
```

3. Argue how your answers to part B1 might be changed after fixing the errors.

After fixing the errors, the count of days where crime occured for the 'Suburb - Incident' value as 'Address Unknown' has increased. Before and after count are attached below-

[79]:
```
#Before Fixing the Errors
crimes=pd.read_csv("Crime_Statistics_SA_2014_2019.csv")
groupby1=crimes.groupby('Suburb - Incident' , sort=True).sum()
```

[80]:
```
groupby1.head()
```

[80]:
```
                     Offence Count
Suburb - Incident
ABERFOYLE PARK              1280.0
ADDRESS UNKNOWN               84.0
ADELAIDE                   24598.0
ADELAIDE AIRPORT           665.0
AGERY                        5.0
```

[81]:
```
#After Fixing the Errors
crimes=pd.read_csv("Crime_Statistics_SA_2014_2019.csv")
crimes['Postcode - Incident'].fillna('Unknown' , inplace = True)
crimes['Suburb - Incident'].fillna('ADDRESS UNKNOWN' , inplace = True)
groupby2=crimes.groupby('Suburb - Incident' , sort=True).sum()
groupby2.head()
```

[81]:
```
                      Offence Count
Suburb - Incident
ABERFOYLE PARK               1280.0
ADDRESS UNKNOWN               252.0
ADELAIDE                    24598.0
ADELAIDE AIRPORT              665.0
AGERY                           5.0
```

However answers to part B1 did not show any changes after fixing the errors. We have just added ‘Address Unknown’ in missing places.

This is because still the count of days with crimes atleast 15 did not increase for the Suburb ‘Address Unknown’

0.3 Task C: Exploratory Analysis on Other Data

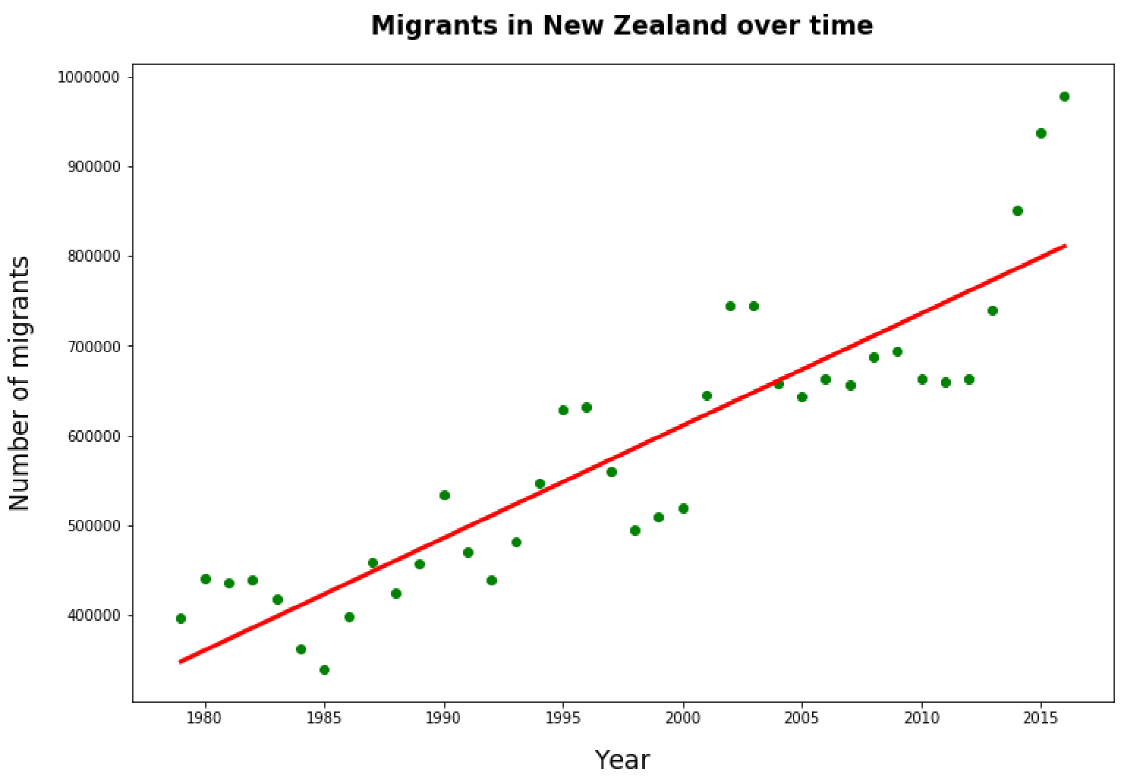
LINK FOR FILE - https://drive.google.com/open?id=19qLlZ5-YC97Xv2ZnC-eVk0izD64HR0IJ

[82]:
```
migration=pd.read_csv("migration_nz.csv")
groupbyyear = migration.groupby("Year").sum()
groupbyyear.reset_index(inplace = True)
```

[83]:
```
groupbyyear.head()
```

[83]:
```
   Year     Value
0  1979  396132.0
1  1980  440970.0
2  1981  435476.0
3  1982  439674.0
4  1983  417516.0
```

[84]:
```
#Linear Regression to predict number of migrants
plt.figure(figsize=(12, 8))

from scipy.stats import linregress
slope, intercept, r_value, p_value, std_err =␣
 ↪linregress(groupbyyear['Year'],groupbyyear['Value'])
line = [slope*xi + intercept for xi in groupbyyear['Year']]
plt.plot(groupbyyear['Year'],line,'r-',linewidth=3)
plt.scatter(groupbyyear['Year'],groupbyyear['Value'],color='g')

plt.xlabel("Year",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Number of migrants",labelpad=15,color='black',fontsize='xx-large')
plt.title("Migrants in New Zealand over time",fontsize='xx-large',fontweight␣
 ↪=600,pad=20)

plt.show()
```



Predictions Task

[85]:
```
#Predictions
migrants_in_2020 = slope*2020+intercept
print(migrants_in_2020)
migrants_in_2050 = slope*2050+intercept
print(migrants_in_2050)
```

```
860609.1957544573
1235555.4918481223
```

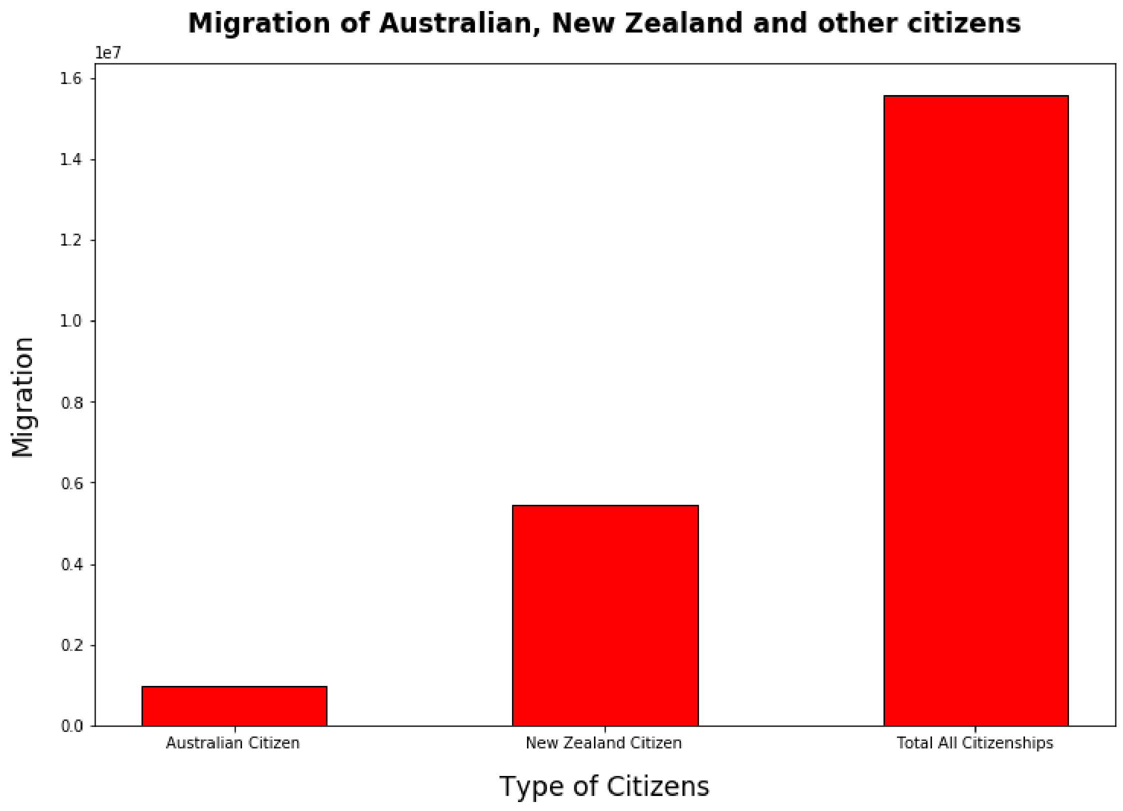
[86]:
```
groupbycitizen = migration.groupby("Citizenship").sum()
groupbycitizen.reset_index(inplace=True)
```

[87]:
```
groupbycitizen.head()
```

[87]:
```
              Citizenship      Year      Value
0      Australian Citizen  57611895   973604.0
```

```
1     New Zealand Citizen  57611895   5452148.0
2  Total All Citizenships  57611895  15591428.0
```

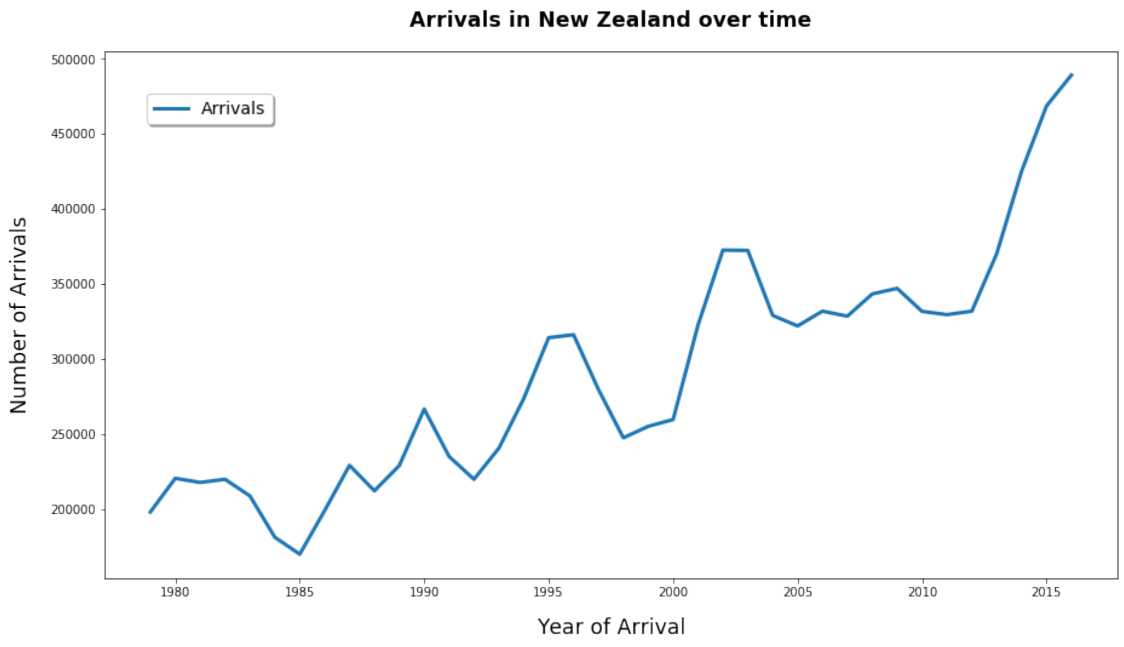
[88]:
```
plt.figure(figsize=(12, 8))

plt.bar(groupbycitizen['Citizenship'],groupbycitizen['Value'],width=0.
 ↪5,align='center',color='red',edgecolor='k',linewidth=1)

plt.xlabel("Type of Citizens",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Migration",labelpad=15,color='black',fontsize='xx-large')
plt.title("Migration of Australian, New Zealand and other␣
 ↪citizens",fontsize='xx-large',fontweight =600,pad=20)

plt.show()
```



[89]:
```
measure_arrivals = migration[migration['Measure']=='Arrivals']
measure_arrivals_year = measure_arrivals.groupby('Year').sum()
measure_arrivals_year.reset_index(inplace = True)
```

[90]:
```
plt.figure(figsize=(15, 8))

plt.plot(measure_arrivals_year['Year'],␣
 ↪measure_arrivals_year['Value'],label='Arrivals',linewidth=3)
```

```
plt.xlabel("Year of Arrival",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Number of Arrivals",labelpad=15,color='black',fontsize='xx-large')
plt.title("Arrivals in New Zealand over time",fontsize='xx-large',fontweight␣
 ↪=600,pad=20)

plt.legend(loc=2,ncol=4, fontsize=14,borderaxespad=2.5,shadow=True)

plt.show()
```



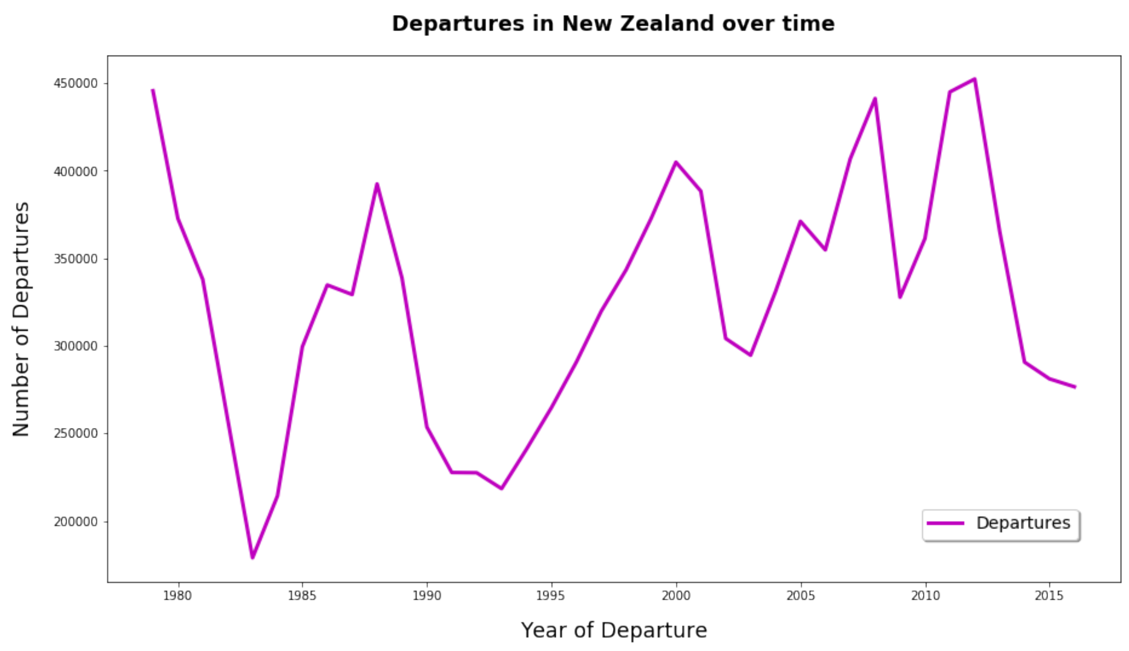
TRENDS -

The above three line graphs plotted are of Arrivals, Departures and Net Migrations in New Zealand over time from the year 1979 to 2016. The Arrivals in New Zealand is seen to face a fluctuation over the time period but also gradually increases. The minimum value of arrivals is seen in the year of around 1985 with a value of 150,000 and the maximum value is seen in the year 2016 with a value of about 500,000. The line graph sees a sudden drop from the year 1998 onwards till the year 2000. The number of arrivals remain constant from the year 2005 till 2014

[91]:
```
measure_departures = migration[migration['Measure']=='Departures']
measure_departures_year = measure_departures.groupby('Year').sum()
measure_departures_year.reset_index(inplace = True)
```

[92]:
```
plt.figure(figsize=(15, 8))

plt.plot(measure_departures_year['Year'],␣
 ↪measure_departures_year['Value'],label='Departures',linewidth=3, color = 'm')
```

```
plt.xlabel("Year of Departure",labelpad=15,color='black',fontsize='xx-large')
plt.ylabel("Number of Departures",labelpad=15,color='black',fontsize='xx-large')
plt.title("Departures in New Zealand over time",fontsize='xx-large',fontweight␣
 ↪=600,pad=20)

plt.legend(loc=4,ncol=4, fontsize=14,borderaxespad=2.5,shadow=True)

plt.show()
```



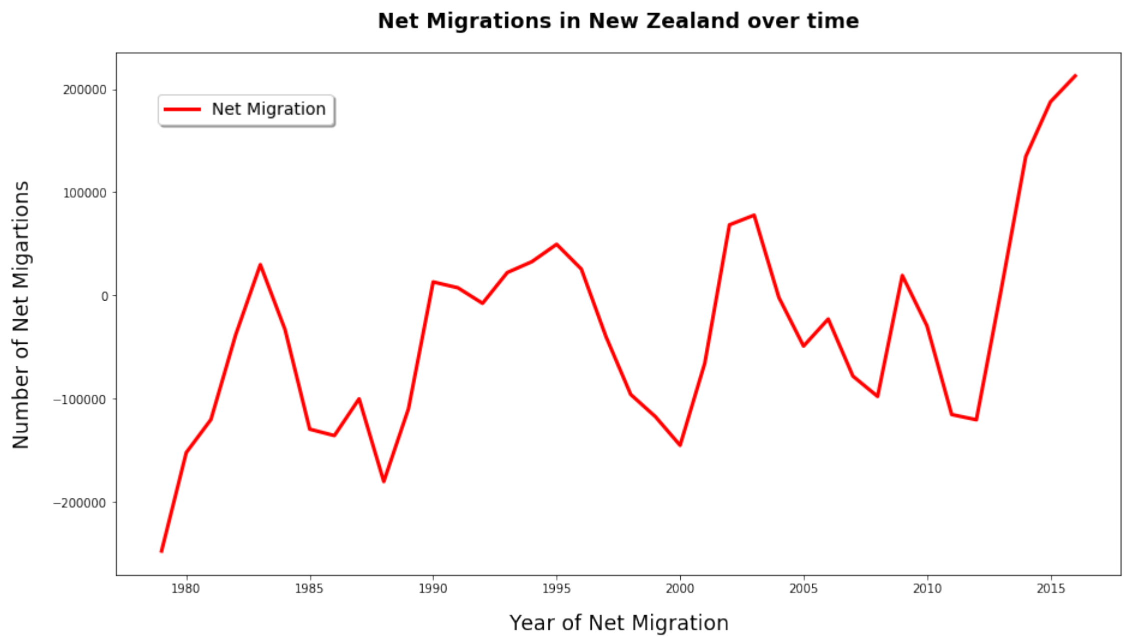
TRENDS -

The graph of Departures in New Zealand over the time period depicts a very fluctuating trend. From 450,000 in the year 1980 it drops suddenly to 150,000 in the year 1983. The trend then suddenly increases to 400,000 in the year 1988 and drops eventually. A fluctuation is seen in the year of 2005 onwards.

[93]:
```
measure_net = migration[migration['Measure']=='Net']
measure_net_year = measure_net.groupby('Year').sum()
measure_net_year.reset_index(inplace = True)
```

[94]:
```
plt.figure(figsize=(15, 8))

plt.plot(measure_net_year['Year'], measure_net_year['Value'],label='Net␣
 ↪Migration',linewidth=3, color = 'r')

plt.xlabel("Year of Net␣
 ↪Migration",labelpad=15,color='black',fontsize='xx-large')
```

```
plt.ylabel("Number of Net␣
 ↪Migartions",labelpad=15,color='black',fontsize='xx-large')
plt.title("Net Migrations in New Zealand over␣
 ↪time",fontsize='xx-large',fontweight =600,pad=20)

plt.legend(loc=2,ncol=4, fontsize=14,borderaxespad=2.5,shadow=True)


plt.show()
```



TRENDS -

The Net Migration in New Zealand for the given time period shows a fluctuating pattern. It increases at 1984,1995, 2014 and 2016 and shows sudden srops in the years 1989,2000 and 2012. The maximum value of Migrations is shown in the year 2016 with a value of 200,000. The minimum value is shown in the year 1979 with a value of -250,000

[ ]:

[ ]: