

I.T Journal Semester 2

Week 1:

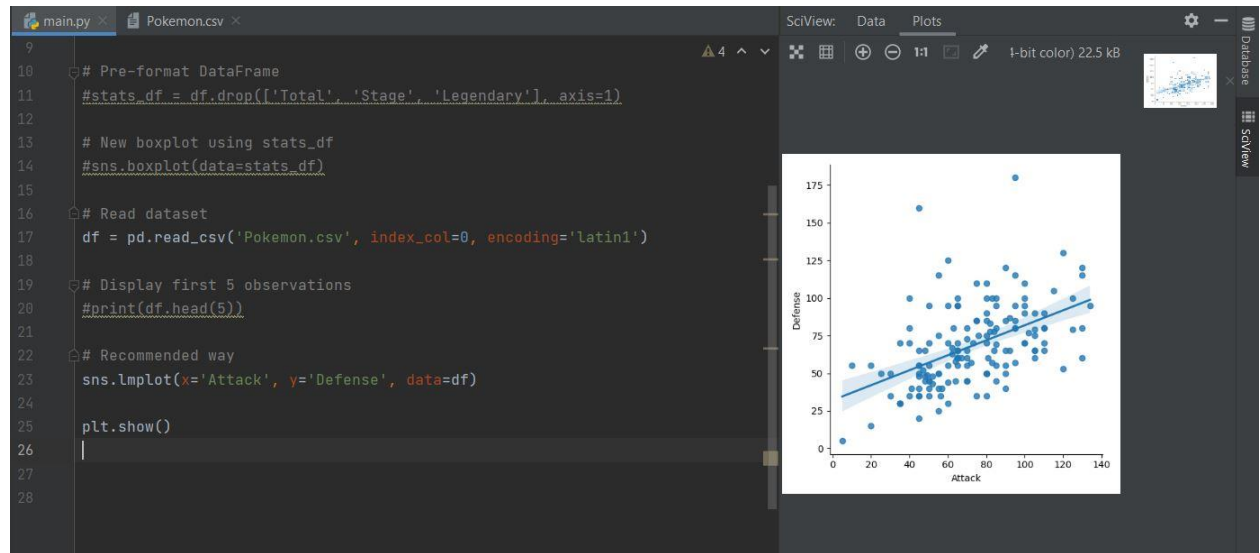
Overview: I was in Isolation.

I.T Journal Semester 2

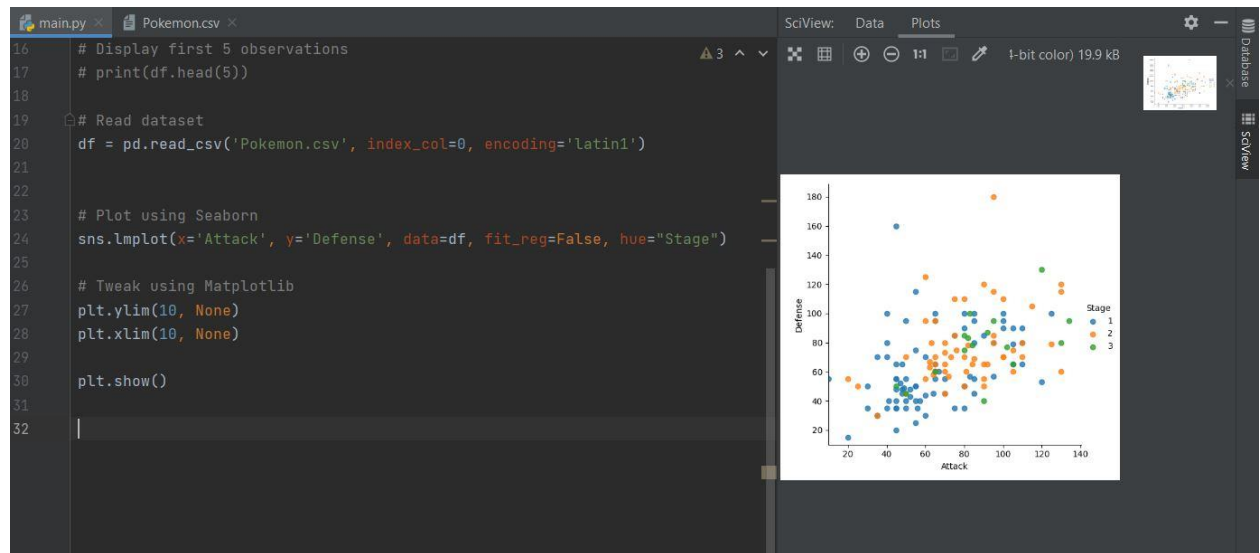
Week 2:

Overview: I was introduced to seaborn and started to learn how to plot data on a scatter plot using CVS files. I started off with a CVS dataset file of all the first generation Pokémon and learnt how to plot the data on a scatter plot using seaborn. I learnt about the different functions that are used in seaborn like **fit_regression** to get rid of the regression line or **hue** to add colour by evolution stage.

Regular Scatter Plot:



Customized Scatter Plot:



Reflection: This tutorial taught me a lot about the library seaborn and the different functions that can be used to visualize data. I learned that seaborn is a data visualization library similar to Matplotlib library and it provides a high level interface for drawing a variety of informative statistical graphs. The reason data visualization is important in data science is because it quickly identifies data trends and patterns, data scientists don't necessarily need to visualise data, but data visualization makes it so much easier to communicate data findings or crucial information from them.

Questions:

I.T Journal Semester 2

1. What did you learn this week?

This week I learned about the basics of seaborn and how to make different statistical graphs, I also learned about the importance of data visualization and how it is used in data science to make communicating data a lot simpler.

2. What made you curious?

While I was learning how to use seaborn, I was curious on how the seaborn library plots the data that I give it and I was also curious on how I can use seaborn for more practical uses like predicting crypto prices with scatter plots made using the in-built seaborn functions.

3. What is the purpose of this?

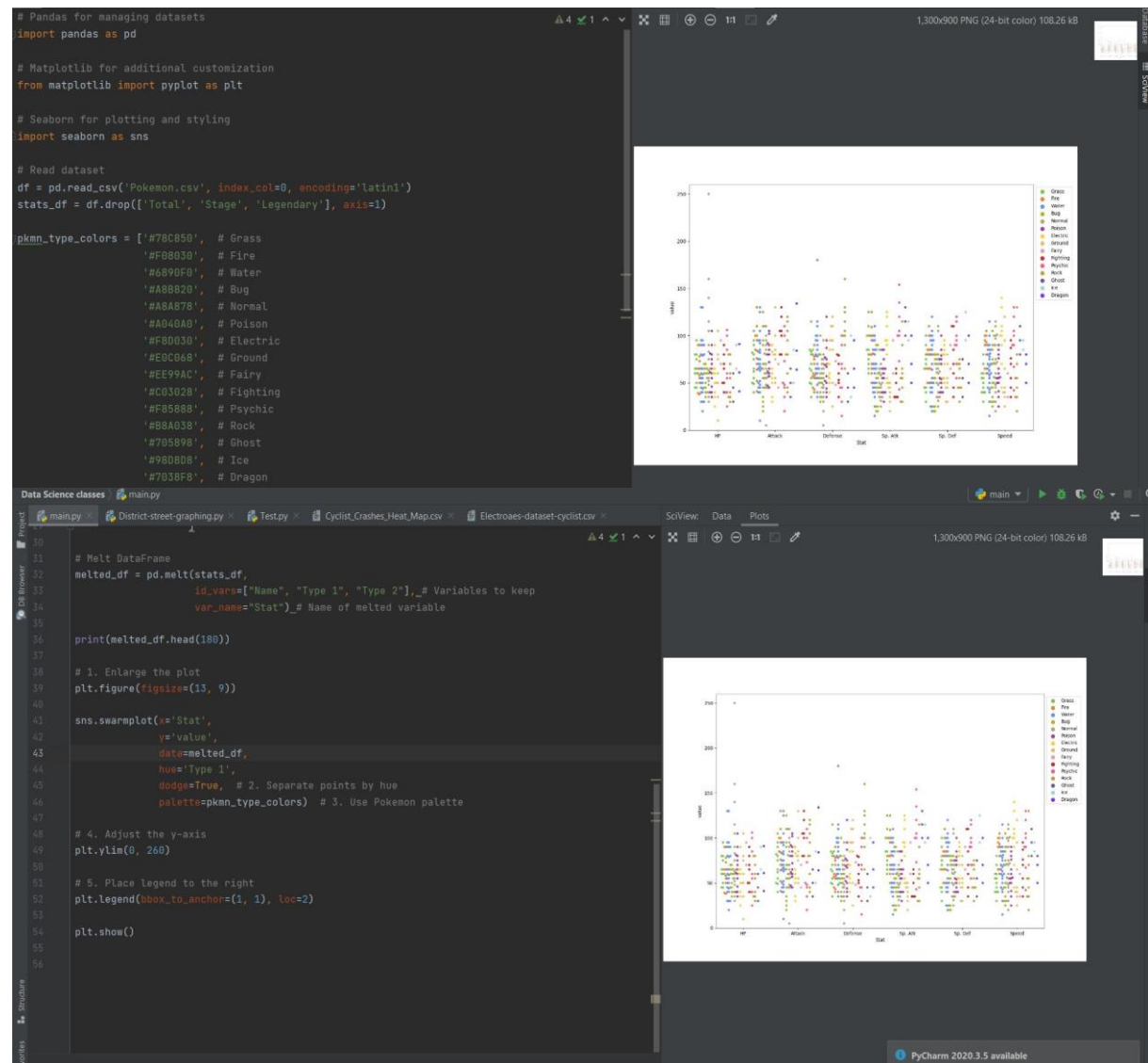
It teaches me on how to analyse trends in a dataset and easily visualise all the sorted data, and analysing trends is very important in data science as analysing trends is a strategy used to make future predictions based on historical data.

I.T Journal Semester 2

Week 3:

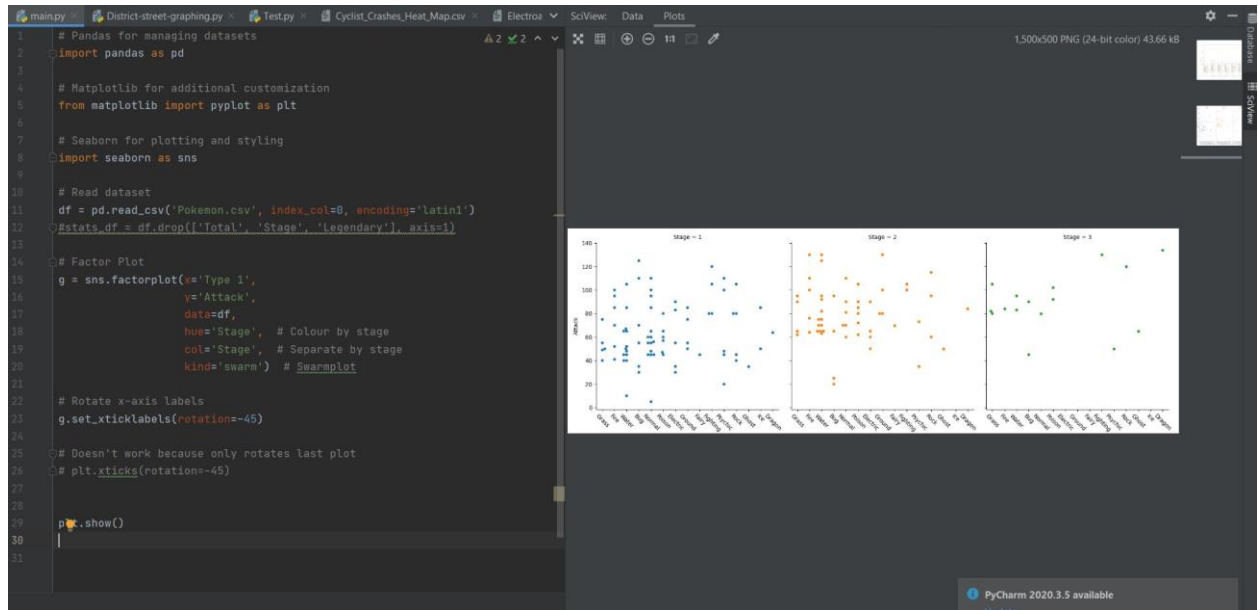
Overview: This week I continued with the seaborn tutorial and learned how to use panda with seaborn and learned about all the different graph's I could make with seaborn like **boxplot**, **violin plot**, **swarm plot**, **histogram**, **heatmap**, **factor plot** etc... It also taught me most of the functions that will allow me to customize my graph to my liking like putting each value in different colours or like putting multiple plots combined into one i.e., **swarm plot** and **violin plot**.

Swarm plot with seaborn and pandas:



Factor plot:

I.T Journal Semester 2



Reflection: Finishing the seaborn tutorial made me learn a great deal about how to use and customize my graphs, I had no idea you could make so many graphs with seaborn, I only thought you could make linear regression graphs, never knew you could make density plots, swarm plots and half of the graphs I've never heard of like violin plots and joint distribution plots. Learning about seaborn will be an essential tool for learning about data science as the purpose of data science is **to find patterns**, and seaborn makes it easy to not only plot all the data that you have collected but also visualize all the patterns in the data that you've collected.

Question:

1. Why do you believe we're studying this?

I believe that I am studying seaborn to help me improve on my data science skills and I have states in my reflection, data science is all about finding patterns and seaborn basically makes it easier to not only plot the data you give it, but also allows you to visualise it and it makes finding patterns in the data much easier.

2. What made you curious?

The thing that made me curious were the inner working of seaborn like how the library actually plots all the data in a dataset and actually give you a visual representation of all the data, as I am not good at creating responsive GUI, so I was interested how the creators of seaborn actually created the seaborn library and how they made it incredibly customizable.

3. How can I verify or test this?

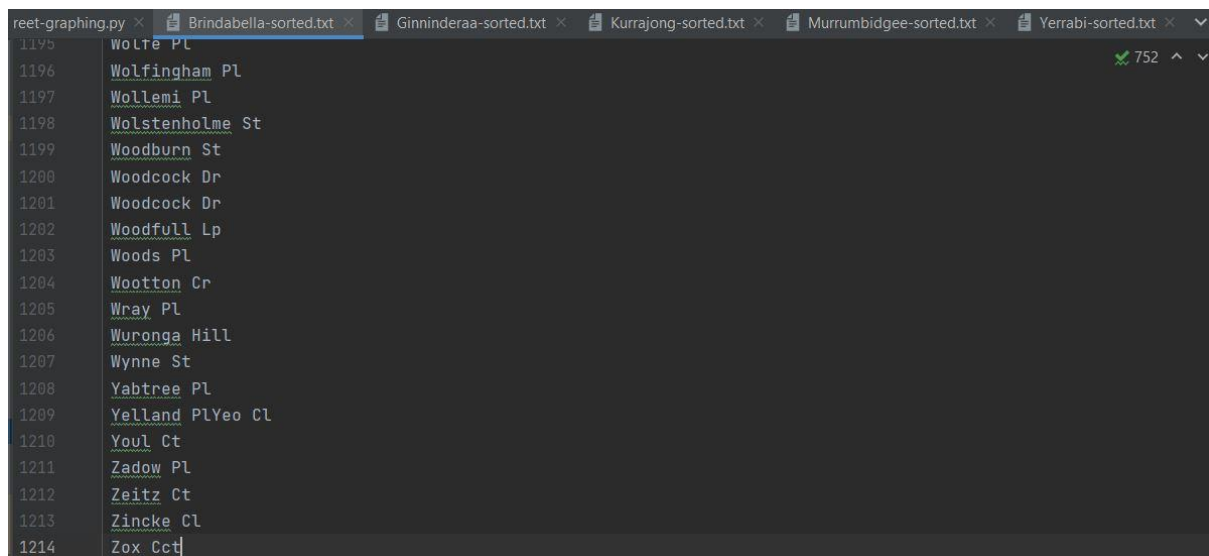
I can verify/test the seaborn library if it is giving me accurate reading, by making my own dataset and crunching the numbers to figure out different patterns in the dataset without the visual representation and then put the dataset into seaborn and if the visual representation is accurate to what I predicted then I will know that seaborn is accurate.

I.T Journal Semester 2

Week 4:

Overview: This week, I was given the task of making my own visualization using seaborn using datasets from [ACT Open Data Portal](#), I decided to use the **Cyclist_Crashes_Heat_Map.csv**, as the dataset I was going to visualize. I looked at the dataset and realized that most of the values in the **Cyclist_Crashes_Heat_Map.csv**, I couldn't use in a graph, but then Mr. Griffin forgot to give me another google doc which basically teaches me about certain useful functions which will allow me to actually use the data in the dataset and put it into a graph but since I didn't have the google doc, I decided to do my own thing and realized that I could graph the amount of cyclist casualties according to where the casualty happened, but there was a problem in the dataset there were way too many locations that I could put in a graph. So, I came up with an idea where I could minimize all the locations in the dataset to only 5 values, which were **Brindabella**, **Ginninderra**, **Kurrajong**, **Murrumbidgee** and **Yerrabi**. These are called electorates, electorates are basically political districts and since the **Cyclist_Crashes_Heat_Map.csv**, only showed which street the casualty happened. I had to figure out which street is in which electorate, for this I used two useful tool, which is this helpful website which labels all the suburbs in each electorate and another website called **where-is**, which allows you to put in a suburb in their site and it comes back with every single street in that specific suburb, after that I spent my entire week copying every single street and sorting them into their specific electorate files which I named, **Brindabella-sorted.txt**, **Ginninderaa-sorted.txt**, **Kurrajong-sorted.txt**, **Murrumbidgee-sorted.txt**, **Yerrabi-sorted.txt**.

Brindabella-sorted.txt:



```
reet-graphing.py x Brindabella-sorted.txt x Ginninderaa-sorted.txt x Kurrajong-sorted.txt x Murrumbidgee-sorted.txt x Yerrabi-sorted.txt x
1195 Wolfe Pl
1196 Wolfingham Pl
1197 Wollemi Pl
1198 Wolstenholme St
1199 Woodburn St
1200 Woodcock Dr
1201 Woodcock Dr
1202 Woodfull Lp
1203 Woods Pl
1204 Wootton Cr
1205 Wray Pl
1206 Wurunga Hill
1207 Wynne St
1208 Yabtree Pl
1209 Yelland Pl Yeo Cl
1210 Youl Ct
1211 Zadow Pl
1212 Zeitz Ct
1213 Zincke Cl
1214 Zox Cct
```

Ginninderaa-sorted.txt:

I.T Journal Semester 2

```
reet-graphing.py x Brindabella-sorted.txt x Ginninderaa-sorted.txt x Kurrajong-sorted.txt x Murrumbidgee-sorted.txt x Yerrabi-sorted.txt x SciV
1497 Wollin Cr
1498 Wolgal Pl
1499 Woodger Pl
1500 Woodhouse Pl
1501 Woolner Cct
1502 Woolnough St
1503 Worral St
1504 Worrell Pl
1505 Wragge Pl
1506 Wrenfordsley Pl
1507 Wrixon St
1508 Wunderly Cct
1509 Wybalena Gr
1510 Wylde Pl
1511 Wyles Pl
1512 Wynn Pl
1513 Wynn St
1514 Yabsley Pl
1515 Yuille Pl
1516 Zali Steggall Wk
1517 Zelman Pl
1518 Zwar Pl
```

Kurrajong-sorted.txt:

```
reet-graphing.py x Brindabella-sorted.txt x Ginninderaa-sorted.txt x Kurrajong-sorted.txt x Murrumbidgee-sorted.txt x Yerrabi-sorted.txt x SciV
1431 Wilton Rd
1432 Wiluna St
1433 Windeyer St
1434 Winnecke St
1435 Wise St
1436 Wollongong St
1437 Wonga St
1438 Wongoola Cl
1439 Wonna St
1440 Woods La
1441 Woolcock St
1442 Woolley St
1443 Woolls St
1444 Workshop Rd
1445 Wormald St
1446 Wylie St
1447 Yallourn St
1448 Yamba PlYapunya St
1449 Yarra Gln
1450 Yarrow Pl
1451 Yha Access Ft
1452 Young St
1453 Zeehan St
1454 Zelling St
```

Murrumbidgee-sorted.txt:

I.T Journal Semester 2

```
reet-graphing.py x Brindabella-sorted.txt x Ginninderaa-sorted.txt x Kurrajong-sorted.txt x Murrumbidgee-sorted.txt x Yerrabi-sorted.txt x
1551 Wyllly PL
1552 Wyndham Av
1553 Wyndham Av
1554 Wynter PL
1555 Wyseleskie Cct
1556 Xenica St
1557 Yaldwyn PL
1558 Yamba Dr
1559 Yamba Dr
1560 Yamba Dr
1561 Yamba Dr
1562 Yamba Dr
1563 Yamba Dr
1564 Yamba Dr
1565 Yambina Cr
1566 Yampi PL
1567 Yanco PL
1568 Yanda St
1569 Yarra Gln
1570 Yarra Gln
1571 Yate GdnsYiman St
1572 Yolla PL
1573 Yorston St
1574 Zeal PL
```

Yerrabi-sorted.txt:

```
reet-graphing.py x Brindabella-sorted.txt x Ginninderaa-sorted.txt x Kurrajong-sorted.txt x Murrumbidgee-sorted.txt x Yerrabi-sorted.txt x
1690 Yamma Wy
1691 Yandell Wy
1692 Yantara St
1693 Yarra St
1694 Yarrawonga St
1695 Yarri St
1696 Yeend Av
1697 Yellowfin StYerra Ct
1698 Yerrabi PL
1699 Yerradhang St
1700 Yidaki Wy
1701 Yinnar St
1702 Yirawala St
1703 Yule St
1704 Yumba Av
1705 Yuranigh Ct
1706 Yuyu St
1707 Zakharov Av
1708 Zamia PL
1709 Zanci St
1710 Zeidler St
1711 Zelling St
1712 Ziggel Wy
1713 Zorzi St
```

Reflection: After I finished copying and pasting all the data, I realised that in every single file there are usually duplicate streets like **Yamba Drive** for example and also some streets like **Athlon drive** are in multiple files like there is an **Athlon drive** in both the Brindabella file and the Ginninderra file, so next week when I actually create the program, I have to consider these issues as duplicates can give me false readings. I also have to figure out a plan on how I am going to organize this data and actually use the data that I have collected and compare them to all the locations in **Cyclist_Crashes_Heat_Map.csv** file. I also need to think of what kind of searching algorithm I am

I.T Journal Semester 2

going to use, I could either go with a linear search which is simpler but much slower, or I could do a binary search which is way more complicated but much faster.

Questions:

1. Did you give your best effort on this?

I spent 2hrs just copying and pasting every single Canberra Street into a text file, and I spent an entire week on different strategies on how I want my code to be built like if I wanted it to be designed with a linear or binary search algorithm and how I was going to overcome the problem of assigning all the streets in the **Cyclist_Crashes_Heat_Map.csv** file to their allocated electorates.

2. What were you most proud of?

That I was able to complete the task of naming all the streets in Canberra and sorting them into their allocated electorates, it wasn't impossible, that I knew but I thought it would take at least a month to create a dataset of this degree when in reality it only took a few hours.

3. Which issues / problems do you see / find?

The one issue I found was how I was going to get all the streets in Canberra and allocate them into their specific electorates and I knew if I just searched up all the streets one by one, it was going to take ages as when I made my dataset, there were over 10,000 values. So, I had to find a way to search all the streets in Canberra more efficient. Then I found 2 tools on google which would help me in achieving this, one was a pdf of all the suburbs in Canberra and also showed the suburbs in their allocated Electorates and the second tool I used was a search engine called **whereis**, it basically allows you to search up a suburb and it will show all the streets in that suburb and I basically copy pasted all the streets into their allocated Electorates, which still took 2hrs but it's much better than spending a month or two searching each street individually.

I.T Journal Semester 2

Week 5:

Overview: This week I focused on trying to graph the data collected in the last week. I first started with addressing the problem of duplicate values in my text files, so I sorted all the street names in each file into lists, **Brindabella_list**, **Ginninderra_list**, **Kurrajong_list**, **Murrumbidgee_list**, **Yerrabi_list**. After sorting all the street names into their allocated lists, I made sure there were no duplicates in the list afterwards. After I solved the duplicate value issue, I started doing data cleaning in the **Cyclist_Crashes_Heat_Map.csv** file as there was a lot of unnecessary values like the intersection values, when I only need the street names, so I did a bit of data cleaning to fix the values by using the split function, to split the street names from the names of the intersections.

Proof of Work:

I.T Journal Semester 2

```
14         else:
15             None
16
17     with open('Sorted-streetnames/Ginninderra-sorted.txt', 'r') as file2:
18         for i in file2:
19             first, *second = i.split()
20             first = first.lower()
21             if first not in Brindabella_list or first not in Ginninderra_list:
22                 Ginninderra_list.append(first)
23         else:
24             None
25
26     with open('Sorted-streetnames/Murrumbidgee-sorted.txt', 'r') as file3:
27         for i in file3:
28             first, *second = i.split()
29             first = first.lower()
30             if first not in Brindabella_list or first not in Ginninderra_list or first not in Murrumbidgee_list:
31                 Murrumbidgee_list.append(first)
32         else:
33             None
34
35     with open('Sorted-streetnames/Murrumbidgee-sorted.txt', 'r') as file4:
36         for i in file4:
37             first, *second = i.split()
38             first = first.lower()
39             if first not in Brindabella_list or first not in Ginninderra_list or first not in Murrumbidgee_list:
40                 Murrumbidgee_list.append(first)
41         else:
42             None
43
44 # Pandas for managing datasets
45 import pandas as pd
46
47 # Matplotlib for additional customization
48 from matplotlib import pyplot as plt
49
50 # Seaborn for plotting and styling
51 import seaborn as sns
52
53 # CSV for reading csv file
54 import csv
55
56 end = 0
57 Brindabella_list = []
58 Ginninderra_list = []
59 Murrumbidgee_list = []
60 Yerrabi_list = []
61
62 -----
63 Brindabella = 0
64 Ginninderra = 0
65 Murrumbidgee = 0
66 Yerrabi = 0
67
68 while end == 0:
69     with open('Sorted-streetnames/Brindabella-sorted.txt', 'r') as file1:
70         for i in file1:
71             first, *second = i.split()
72             first = first.lower()
73             if first not in Brindabella_list:
74                 Brindabella_list.append(first)
```

I.T Journal Semester 2

```
with open('sorted-ginninderra/Verrabi-sorted.txt', 'r') as file5:
    for i in file5:
        first, *second = i.split()
        first = first.lower()
        if first not in Brindabella_list or first not in Ginninderra_list or first not in Kurrajong_list or first not in Murrumbidgee_list or first not in Verrabi_list:
            Verrabi_list.append(first)
        else:
            None
    end += 1
end = 0
```

Reflection: Doing this task made me realise a lot of new skills that I had developed without me even knowing, like data cleaning for example, I had no clue what data cleaning until now and realised that data cleaning is just cleaning up the data and reforming or rearranging the raw data to get a desired value. This task also taught me a lot about bugs, like the bug in my program where there were duplicates values in the files which throws my count off. This task taught me a lot about handling bugs and gave me experiences on how to handle these bugs.

Questions:

1. What are some things you did really well?

I did a really good job in identifying an unresolved problem in my code and figuring out solutions to solve that problem, and once I found a solution to the problem, I implemented it into my code and fixed it.

2. What lessons were learned from failure?

When I first made my code, I tried to do data cleaning for the first time and I tried splitting the slashes and the brackets out of the value and it worked but some of the values still had brackets and slashes in them and I was confused why. Later after a bit of trial and error, I realised that some values had an extra slash or bracket and I made a second condition that will remove all the extra slashes and brackets and it worked as intended.

3. What's the purpose of this?

To learn about how to do data cleaning in a dataset and preparing the data to be sorted so I can use it in a graph.

I.T Journal Semester 2

Week 6:

Overview: This week in IT, I used the data that I cleaned in the last week and started sorting them in their specific electorate lists. After sorting all the values into their specific electorate list and checking how many values there were as there were supposed to be 818 values in the dataset as there were 818 crash sites. Then I made 5 variables called, **Brindabella**, **Kurrajong**, **Ginninderra**, **Murrumbidgee**, **Yerrabi**. After making the variables, I simply did a linear search using the cleaned up values in the **Cyclist_Crashes_Heat_Map.csv** and started trying to match them seeing if that value is in any of the electorate lists which were, **Brindabella_list**, **Ginninderra_list**, **Kurrajong_list**, **Murrumbidgee_list**, **Yerrabi_list**. If the value matched up to any of the values that were in the electorate list, then that electorate got 1 point added to any of these variables, **Brindabella**, **Kurrajong**, **Ginninderra**, **Murrumbidgee**, **Yerrabi**. After all the values have been matched up, I used matplotlib to plot all the variables onto a simple bar graph and in the end, Kurrajong had the most crashes.

Code:

```
# Matplotlib for additional customization
from matplotlib import pyplot as plt

# Seaborn for plotting and styling
import seaborn as sns

#CSV for reading cvs file
import csv

end = 0
Brindabella_list = []
Ginninderra_list = []
Kurrajong_list = []
Murrumbidgee_list = []
Yerrabi_list = []
#-----
-----

Brindabella = 0
Ginninderra = 0
Kurrajong = 0
Murrumbidgee = 0
Yerrabi = 0

while end == 0:
    with open('Sorted-streetnames/Brindabella-sorted.txt', 'r') as file1:
```

I.T Journal Semester 2

```
for i in file1:
    first, *second = i.split()
    first = first.lower()
    if first not in Brindabella_list:
        Brindabella_list.append(first)
    else:
        None

with open('Sorted-streetnames/Ginninderaa-sorted.txt', 'r') as file2:
    for i in file2:
        first, *second = i.split()
        first = first.lower()
        if first not in Brindabella_list or first not in
Ginninderra_list:
            Ginninderra_list.append(first)
        else:
            None

with open('Sorted-streetnames/Kurrajong-sorted.txt', 'r') as file3:
    for i in file3:
        first, *second = i.split()
        first = first.lower()
        if first not in Brindabella_list or first not in
Ginninderra_list or first not in Kurrajong_list:
            Kurrajong_list.append(first)
        else:
            None

with open('Sorted-streetnames/Murrumbidgee-sorted.txt', 'r') as file4:
    for i in file4:
        first, *second = i.split()
        first = first.lower()
        if first not in Brindabella_list or first not in
Ginninderra_list or first not in Kurrajong_list or first not in
Murrumbidgee_list:
            Murrumbidgee_list.append(first)
        else:
            None

with open('Sorted-streetnames/Yerrabi-sorted.txt', 'r') as file5:
    for i in file5:
        first, *second = i.split()
        first = first.lower()
        if first not in Brindabella_list or first not in
Ginninderra_list or first not in Kurrajong_list or first not in
Murrumbidgee_list or first not in Yerrabi_list:
            Yerrabi_list.append(first)
        else:
            None

end += 1

end = 0
with open('Electroaes-dataset-cyclist.csv', 'w', newline='') as f:
    field_names = ['Electorates', 'Casualties']
    thewriter = csv.DictWriter(f, fieldnames=field_names)

    thewriter.writeheader()
    while end == 0:
        with open('Cyclist_Crashes_Heat_Map.csv', 'r') as cvs_file:
            cvs_reader = csv.reader(cvs_file)
```

[illegible]

I.T Journal Semester 2

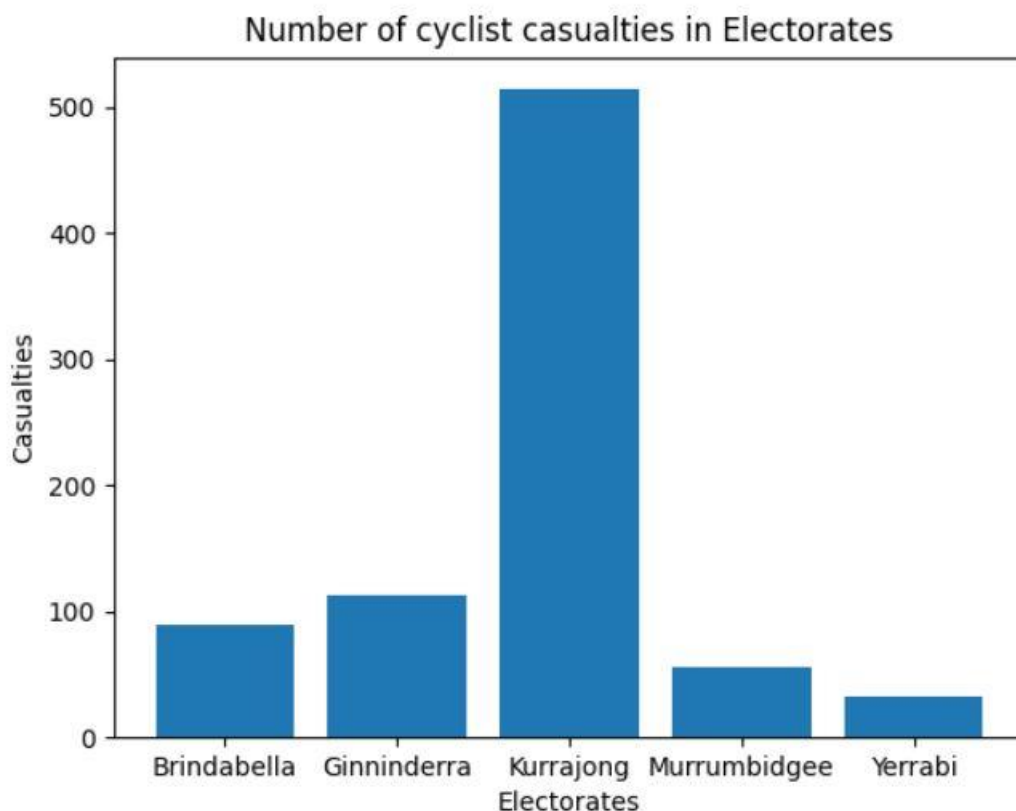
```
       thewriter.writerow({'Electorates':
'Ginninderra', 'Casualties': int(line[6])})
    else:
        Brindabella += int(line[6])
        thewriter.writerow({'Electorates': 'Brindabella',
'Casualties': int(line[6])})
    end += 1

#-----
-----

x = ["Brindabella", "Ginninderra", "Kurrajong", "Murrumbidgee", "Yerrabi"]
y = [90, 113, 514, 56, 33]
plt.bar(x,y)
plt.xlabel("Electorates")
plt.ylabel("Casualties")
plt.title("Number of cyclist casualties in Electorates")
plt.show()

print(f"Casualties in Brindabella: {Brindabella}")
print(f"Casualties in Ginninderra: {Ginninderra}")
print(f"Casualties in Kurrajong: {Kurrajong}")
print(f"Casualties in Murrumbidgee: {Murrumbidgee}")
print(f"Casualties in Yerrabi: {Yerrabi}")
```

Output:



I.T Journal Semester 2

Reflection: I do admit my code is not the best but it did teach me something, and that is that I was able to understand long complicated strings of code and I knew why each line of code was important in my program to achieve the desired outcome. Doing this task made me realise that I have gained the rudimentary knowledge to continue to even more difficult projects that I can understand and go through it with ease. This task also taught me a lot about CVS and how to use it, as I have never used it before and now, I am confident in using CVS and confident in applying it to any project that requires CVS.

Questions:

1. If you could do this week over, what would you do differently?

I would first do the searching differently like before I search all the streets separately which did increase the speed of the program but it gave me inaccurate values like duplicate values. So instead, I just searched the streets all together which slowed down the speed of my program but gave me a more accurate reading. I wish I started with just searching it entirely in that way, I wouldn't have needed to waste my time on searching everything separately.

2. What about your thinking, learning or work brought you the most satisfaction?

My work brought me the most satisfaction as I spent a long time optimizing my code and when it finally worked, I was so happy. It also gave me satisfaction that I was able to find a problem and find multiple solutions and implement one of the solutions to my program, which shows that I have improved on my problem solving abilities.

3. Which issues / problems do you see / find?

I found a problem where I was getting inaccurate values and somehow got 1000 crash sites even though there was only 808 crash sites and this was because I search all of the streets separately. SO I decided to search it all in one bulk and it fixed the values and got 808 crash sites.

I.T Journal Semester 2

Week 7:

Overview: This week I was tasked to plot all the locations of bicycle crashes in the **Cyclist_Crashes_Heat_Map.csv** on a map. I started by first doing data cleaning and started cleaning up the data as there were a lot of unnecessary values like the brackets and I need to separate the longitude and latitude values and, in the file, there were combined in one line so I had to separate them first so I could use them individually. I first started cleaning the data on my python test file, I started by first exporting the location of the crash site into a list called location and each value was a simple string that had the longitude and latitude of each crash site. I then in a separate for loop, I start separating the longitude and latitude from the location values and also got rid of unnecessary symbols in the string like brackets. After cleaning the data and separating the longitude and latitude values, I stored the longitude and latitude values in a CVS file called **Longitude_Latitude_Locations.csvs**, so I can use those values later to plot them on my map.

Code:

```
import csv

LOCATION = []
end = 0
while end == 0:
    with open('Cyclist_Crashes_Heat_Map.csv', 'r') as cvs_file1:
        cvs_reader = list(csv.reader(cvs_file1))
        skip = 0
        end2 = 0

        while end2 == 0:
            for line1 in cvs_reader:
                if skip == 0:
                    skip += 1
                else:
                    LOCATION.append(line1[10])
            end2 += 1

        end += 1

with open('Longitude_Latitude_Locations.csvs', 'w', newline='') as f:
    field_names = ['Latitude', 'Longitude']
    thewriter = csv.DictWriter(f, fieldnames=field_names)
    thewriter.writeheader()

    for i in LOCATION:
```

I.T Journal Semester 2

```
latitude, longitude = i.split(",")
latitude = latitude.replace("(", "")
longitude = longitude.replace(")", "")
thewriter.writerow({'Latitude': float(latitude), 'Longitude':
float(longitude)})
```

Reflection: This task gave me even more experience on how to data clean and its good practise for harder projects in the future. This task also gave me more experience using CVS and this is the first time I'm using CVS in a project and to not only to create a dataset but also use that dataset to get a desired outcome for example, here I cleaned the data and created a CVS file to store that cleaned data. The I'm going to use the values in the newly created CVS file and plot them on my map.

Questions:

1. What resources did you find / do you have that can help you learn?

I found that learning documentation was extremely helpful as I was going to use a library called plotly. To plot all the points on my map and I didn't know how to use plotly until I read the documentation and learned what each command did.

2. Which ideas make the most sense?

The idea of first cleaning the data of any unnecessary values like brackets so that it doesn't mess up your program when you run it, like for me I want to map the values and I just need the numbers, if I left the data raw then I would get a error because there are unnecessary values in the data like brackets.

3. Which issues / problems do you see / find?

I found that the location data for both the latitude and longitude were together and I couldn't plot them in a graph because I need to use both values individually and then values were in closed in brackets, which I can't use as a bracket is not a number that can be plotted.

I.T Journal Semester 2

Week 8:

Overview: This week I worked on plotting all the data that I cleaned up in the last week and started using a library called Plotly to plot all the data on a 2d map of Canberra. I first started extracting all the data from the **Longitude_Latitude_Locations.csv**, and after that I simply looked up the documentation for the library Plotly and used different functions to plot the data from the **Longitude_Latitude_Locations.csv** into a 2d map of Canberra.

Code:

```
import pandas as pd
import plotly.express as px

df = pd.read_csv('Longitude_Latitude_Locations.csv')

fig = px.scatter_mapbox(df,
                        lon= df['Longitude'],
                        lat= df['Latitude'],
                        zoom= 3,
                        width= 1200,
                        height= 900,
                        title= "Cyclist Crashes In Canberra"
                        )

fig.update_layout(mapbox_style='open-street-map')
fig.update_layout(margin={"r":0, "t":50, "l":0, "b": 10})
fig.show()
```

Reflection: This week I learned a lot about documentations, this was because I had to find a library that could easily and efficiently plot thousands of values onto a 2d map, and Plotly was the library I chose to plot my data. I first had to learn the documentation from Plotly and this let me develop and polish my skills in documentation reading, which is a very important skill in coding as when you go into larger projects you will likely use these libraries and you need to learn how to read documentation so you can use the library for whatever project you are doing.

Questions:

1. Did you come to class prepared to learn, in both your attitude and with all your supplies?

I.T Journal Semester 2

Yes, I came prepared to class with a plan on what I was going to do this lesson and with my attitude I was very excited as I was interested in the subject and wanted to learn more about it.

2. How did you agree or disagree with yourself / others?

I agreed that it's going to be difficult to make my own 2d map and it would take a lot of time if I used a library like matplotlib. So, I opted for another library which in this case was Plotly to make the job of making the map and plotting the data a lot easier.

3. How can I verify or test this?

I can verify that my program works by comparing the ACT open data portal heatmap and wherever there is a lot of heat on the map, I can check that specific area on my map to see if there is a lot of values or plot points in that specific area.

I.T Journal Semester 2

Week 9:

No work was given

I.T Journal Semester 2

Week 10:

No work was given

I.T Journal Semester 2

Week 11:

Overview: No Work Was Given.

I.T Journal Semester 2

Week 12:

Overview: I carried on finishing my uncomplete journals

I.T Journal Semester 2

Week 13:

Overview: I carried on finishing my incomplete journals

I.T Journal Semester 2

Week 14:

Overview: I carried on finishing my incomplete journals

I.T Journal Semester 2

Week 15:

Overview: I carried on completing my unfinished journals

I.T Journal Semester 2

Week 16:

Overview: I carried in completing my unfinished journals.

I.T Journal Semester 2

Semester Review

This semester for I.T, I feel like I did do a lot of good work in the first term but not a lot in the second term. In the first term I did progress a lot in my data science course and learned a lot from it, and in the second term I was very behind in journaling and needed to improve on my organizational skills.