

# Week 7: Visualising Spatial Data

Visual Data Analytics

University of Sydney



# Outline

- Plotting maps
  - Understanding projections
- Adding data to maps
  - Choropleth
- Heatmap
  - Heatmaps are not choropleths!

# Motivation

- Many business decisions depend on geospatial data
  - Store locations
  - Investment in real estate
  - Profile customers in different parts of a city
- Geospatial data used in many other applications.

# Packages

- There is an entire field known as Geographical Information Systems (GIS) that we could cover over an entire course.
- It involves complicated problems including how coordinates are obtained, how they can be projected in different ways, etc.
- The `geopandas` and `geoplot` packages can be used to handle these problems for us.
- This package has many dependencies and can be fiddly to install

# World map



# What is a projection?

- The earth is round.
- There are different ways to project a sphere onto a flat screen.
- All create distortion
- The Mercator projection (previous slide) makes areas near the poles look bigger.

# Robinson Projection



# Does this matter?

- For looking at small countries or cities... no
- For large countries near the North pole... yes
- For the North America it is common to use an Albers projection.
- For Europe it is common to use an Lambert azimuthal projection.



# North America



# Europe



# What about Australia?

- To see we will actually download data for Australia.
- First we need to download a *Shapefile*
- A Shape file is actually multiple files that store all the information about borders.
- We will use the SA4 areas of Australia.
- The shapefiles can be downloaded from the [Australian Bureau of Statistics](#).
- Shapefiles are a standard format used globally.

# Read in shapefile

```
aus = gpd.read_file('../data/SA4_2021_AUST_SHP_GDA2020')
aus
```

```
##      SA4_CODE21  ...                                     geometry
## 0           101  ...  MULTIPOLYGON (((150.05261 -37.26253, 150.05251...
## 1           102  ...  MULTIPOLYGON (((151.31497 -33.55578, 151.31496...
## 2           103  ...  POLYGON ((150.14236 -32.34153, 150.14255 -32.3...
## 3           104  ...  MULTIPOLYGON (((153.07639 -30.42982, 153.07645...
## 4           105  ...  POLYGON ((148.67619 -29.50976, 148.67662 -29.5...
## ..           ...  ...                                     ...
## 103          899  ...                                     None
## 104          901  ...  MULTIPOLYGON (((167.94747 -29.12757, 167.94748...
## 105          997  ...                                     None
## 106          999  ...                                     None
## 107          ZZZ  ...                                     None
##
## [108 rows x 13 columns]
```

# Simplify series

These shapefiles are very detailed making plotting slow. So we can simplify them.

```
aus.geometry = aus.geometry.simplify(0.001)
```

# Australia

```
geoplot.polyplot(aus, geoplot.crs.Mercator())
```



# Australia

```
geoplot.polyplot(aus, geoplot.crs.Robinson())
```



# Summary

- The Mercator projection looks OK for Australia
- However we don't simply want to plot maps
- We want to add *data* to these maps.
- This can be done using a choropleth
- We will use data from the Australian Bureau of Statistics on mortgage repayments in Sydney.
- We will merge this with the geopandas data frame



# Choropleth

# Data

```
import pandas as pd
mortgage = pd.read_csv('../data/Mortgage.csv')
mortgage
```

```
##                                SA4  ...  Total
## 0                        Central Coast  ...  152870
## 1  Sydney - Baulkham Hills and Hawkesbury  ...  89466
## 2                        Sydney - Blacktown  ...  135540
## 3      Sydney - City and Inner South  ...  178493
## 4      Sydney - Eastern Suburbs  ...  122105
## 5      Sydney - Inner South West  ...  227887
## 6      Sydney - Inner West  ...  133899
## 7  Sydney - North Sydney and Hornsby  ...  180531
## 8      Sydney - Northern Beaches  ...  105115
## 9  Sydney - Outer West and Blue Mountains  ...  130809
## 10      Sydney - Outer South West  ...  104420
## 11      Sydney - Parramatta  ...  188066
## 12      Sydney - Ryde  ...  83400
## 13      Sydney - South West  ...  155905
## 14      Sydney - Sutherland  ...  90775
##
```

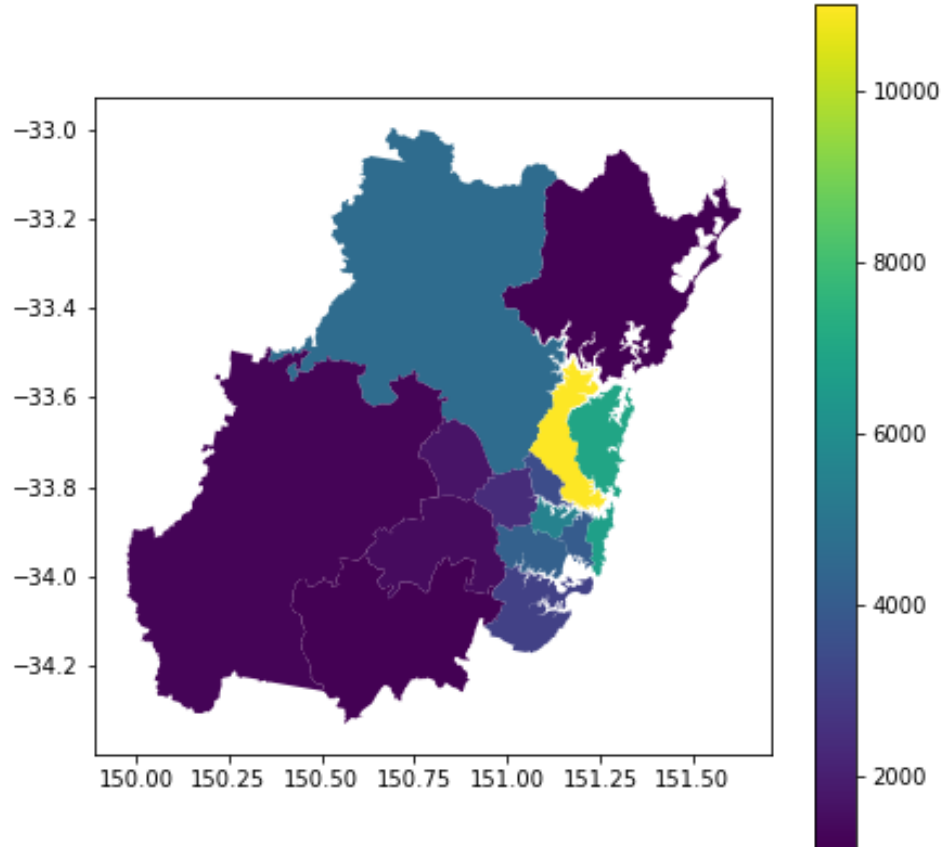
# Merge

```
import pandas as pd
merged = aus.merge(mortgage, how='right', left_on = 'SA4_NAME21', right_
merged
```

##	SA4_CODE21	SA4_NAME21	...	Not applicable
## 0	102	Central Coast	...	107844
## 1	115	Sydney - Baulkham Hills and Hawkesbury	...	51873
## 2	116	Sydney - Blacktown	...	82831
## 3	117	Sydney - City and Inner South	...	146248
## 4	118	Sydney - Eastern Suburbs	...	97426
## 5	119	Sydney - Inner South West	...	167199
## 6	120	Sydney - Inner West	...	100829
## 7	121	Sydney - North Sydney and Hornsby	...	130327
## 8	122	Sydney - Northern Beaches	...	72014
## 9	124	Sydney - Outer West and Blue Mountains	...	84896
## 10	123	Sydney - Outer South West	...	62085
## 11	125	Sydney - Parramatta	...	139866
## 12	126	Sydney - Ryde	...	60698
## 13	127	Sydney - South West	...	105896 <sup>19</sup>

# Choropleth

```
merged.plot(column="$5,000 and over", legend=True)
```

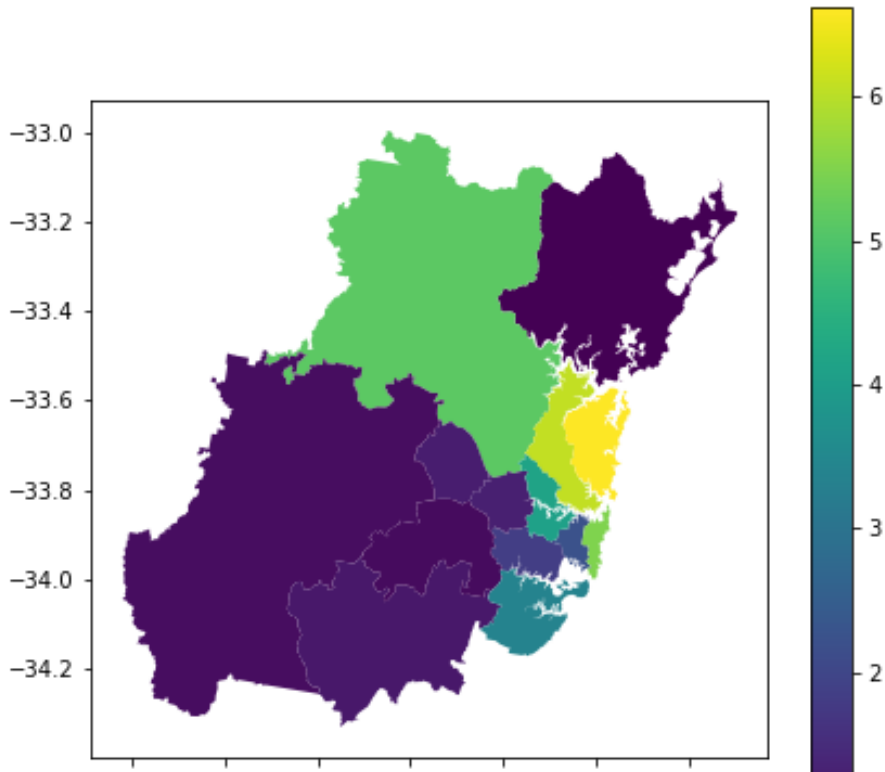


# A word of caution

- The choropleth on the previous slide uses *spatially extensive* data.
- By this we mean they are raw counts of people that do not take the population of each area into account.
- It is better to use *spatially intensive* data in a choropleth

# Proportion

```
merged["$5,000+ Percentage"] = 100*merged["$5,000 and over"]/merged["Total"]  
merged.plot(column="$5,000+ Percentage", legend=True)
```



# Making things interactive

# Why interactivity?

- Unless you know the location well, it may be hard to tell which regions are which.
- Also urban areas tend to be smaller therefore it helps to be able to zoom in and move around.
- We can also augment the choropleth with data from individual locations.



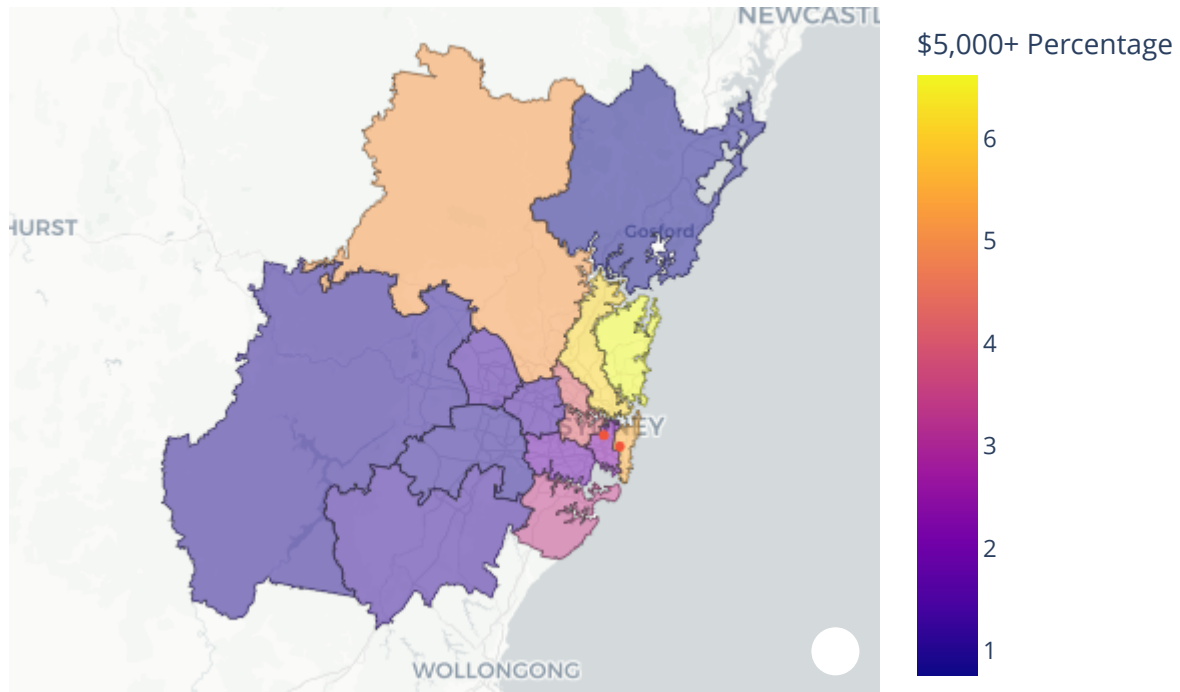
# Plotly Code

```
import plotly.express as px
merged.index = merged['SA4']
fig = px.choropleth_mapbox(merged,
    geojson=merged.geometry,
    locations = merged.index,
    color = "$5,000+ Percentage",
    opacity=0.5, mapbox_style="carto-positron",
    zoom = 7, #Zoom in scale
    center = {"lat": -33.7, "lon": 150.75})
fig.add_scattermapbox(lat = [-33.89083755569, -33.91648618776671],
    lon = [151.18711290919222, 151.23079491201923],
    text = ["The University of Sydney", "UNSW"],
    marker_size=6)
```

```
<div>                                <script type="text/javascript">window.Plot
    <script src="https://cdn.plot.ly/plotly-2.16.1.min.js"></script>
```

```
fig.write_html('plotly_choropleth.html')
```

# Interactive



# Summary

- There are a rich range of visualisations that you can create with geographic data.
- However you can generally only look at one variable at one moment of time.
- You can construct multiple choropleths and display them together but this can be difficult to interpret.

# Misleading choropleths

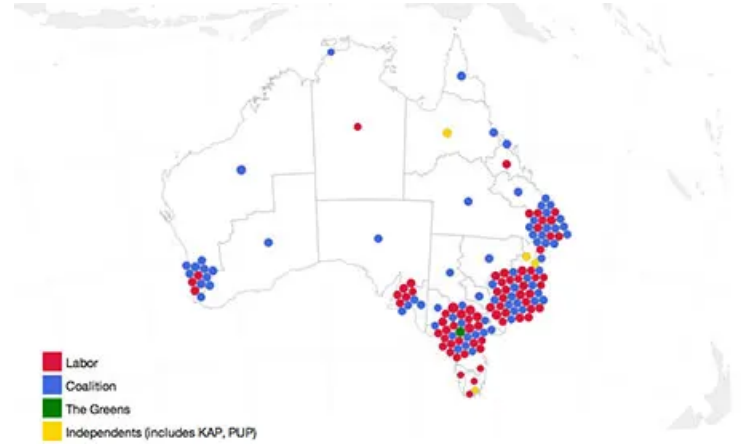
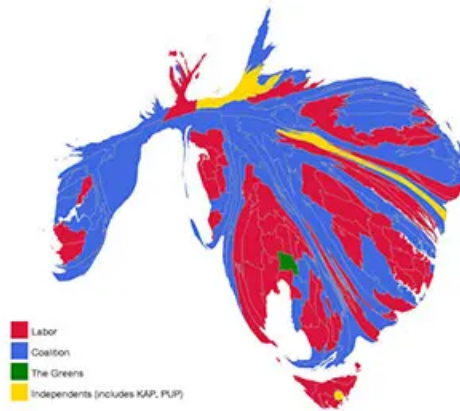
- In many countries, the population density is not uniform
- This includes Australia, the US and China where most of the population lives on the coast and well as India and France as other examples with less densely populated regions.
- This commonly leads to misinterpretation of data.

# Australian 2010 election

- The 2010 election was very close.
- However the map on the right suggests the Coalition (blue) was dominant.
- This map shows land but not voters.



# Two alternatives



# Discussion

- The alternative on the left is called a *cartogram*.
  - It can be visually jarring
- The alternative on the right loses precise spatial information.
- All plots and further discussion are from [this article](#)
- This issue is not restricted to politics, an example on brand choice may suffer from similar issues.

# A word on heatmaps



# Heatmaps

- A choropleth is often called a *heatmap*.
- A heatmap is a similar idea with areas shown using different colors.
- However a heatmap is used to visualise numbers in a matrix.
  - A data matrix
  - A correlation matrix

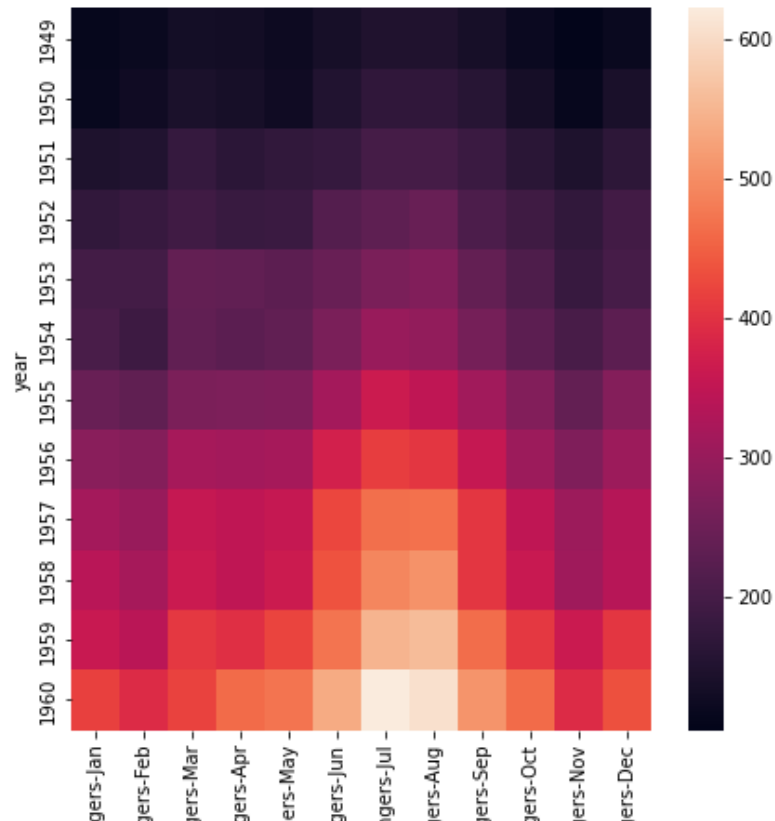
# Data Preparation

```
import seaborn as sns
fl = sns.load_dataset('flights')
fltab = fl.pivot(index = 'year', columns = 'month')
fltab
```

##	passengers												
##	month	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
##	year												
##	1949	112	118	132	129	121	135	148	148	136	119	104	118
##	1950	115	126	141	135	125	149	170	170	158	133	114	140
##	1951	145	150	178	163	172	178	199	199	184	162	146	166
##	1952	171	180	193	181	183	218	230	242	209	191	172	194
##	1953	196	196	236	235	229	243	264	272	237	211	180	201
##	1954	204	188	235	227	234	264	302	293	259	229	203	229
##	1955	242	233	267	269	270	315	364	347	312	274	237	278
##	1956	284	277	317	313	318	374	413	405	355	306	271	306
##	1957	315	301	356	348	355	422	465	467	404	347	305	336
##	1958	340	318	362	348	363	435	491	505	404	359	310	337
##	1959	360	342	406	396	420	472	548	559	463	407	362	405
##	1960	417	391	419	461	472	535	622	606	508	461	390	432

# Heatmap

```
import seaborn as sns
sns.heatmap(fltab)
```



# Correlation heatmap

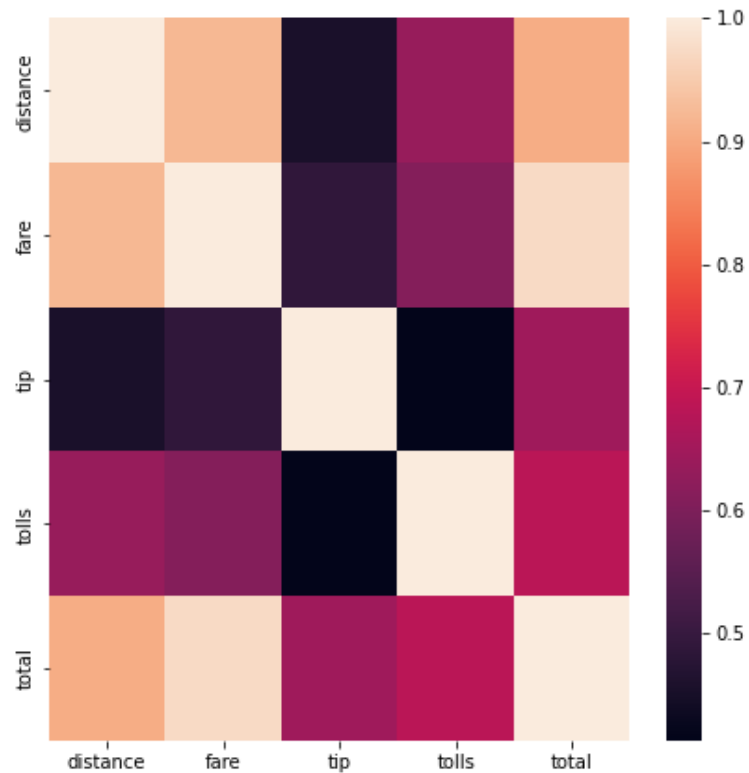
- With a moderate number of numerical variables we can compute the correlation of all pairs of variables.
- These can be put into a matrix known as the correlation matrix.
- This can also be shown in a heatmap.
- We will do this some numerical variables from the taxi data

# Data preparation

```
taxisdat = sns.load_dataset('taxi')  
taxisnum = taxisdat[['distance', 'fare', 'tip', 'tolls', 'total']]  
taxiscor = taxisnum.corr()
```

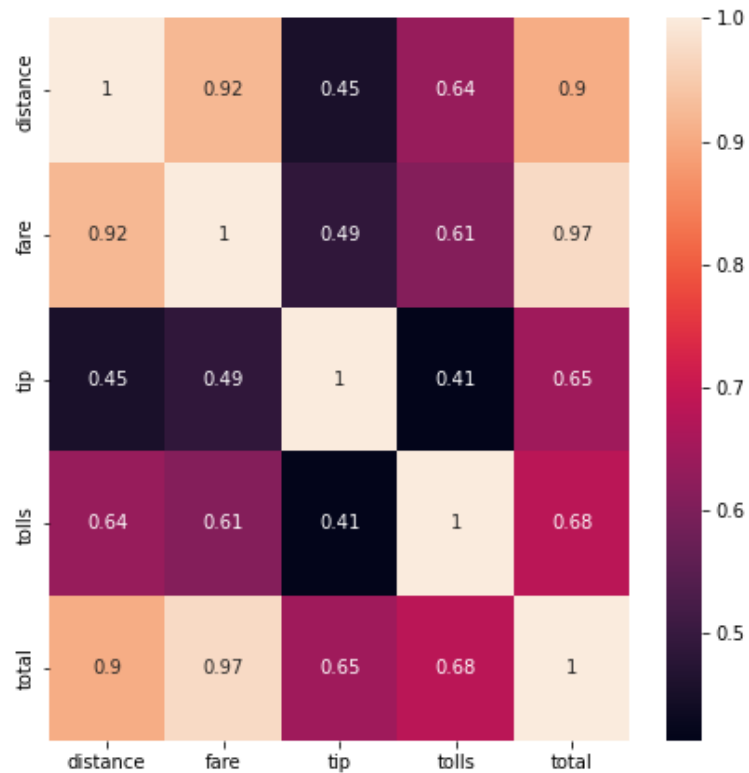
# Correlation heatmap

```
sns.heatmap(taxiscor)
```



# With annotation

```
sns.heatmap(taxiscor, annot = True)
```



# Heatmaps

- Heatmaps work best when the adjacent rows and columns are related.
- This is why it works well for the flights example.
- For a correlation heatmap, the row and column order is arbitrary.
- It is better than looking at a correlation matrix.
- However, something like a pairs plot shows more information about the relationships between variables.



# Wrap-up

# Conclusions

- Spatial data can be used to create valuable visualisations. However
  - Think carefully about how population is distributed in geographic regions
  - Do not neglect other visualisation methods
- Heatmaps are different to choropleths even if these two words are often confused.
  - Be aware of the strengths and weaknesses of heatmaps.

# Questions