

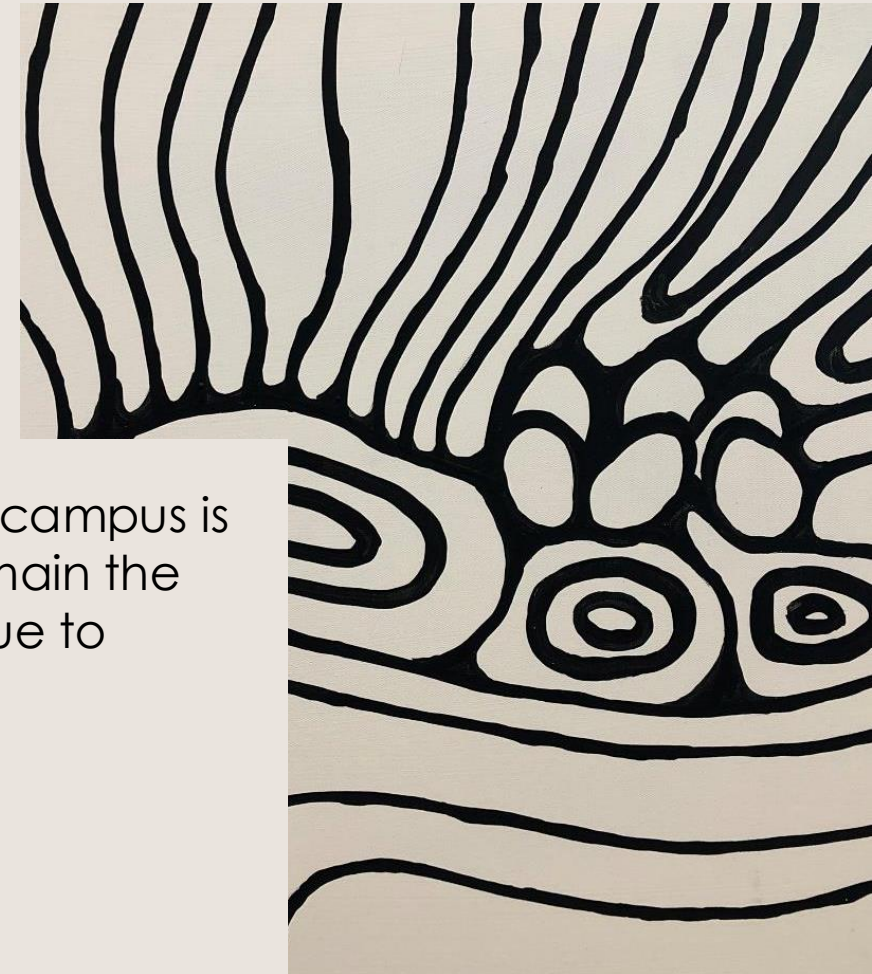


# Topic Eight: Programming and BI

## INMT5526: Business Intelligence

# Acknowledgement of country

The University of Western Australia acknowledges that its campus is situated on Noongar land, and that Noongar people remain the spiritual and cultural custodians of their land, and continue to practise their values, languages, beliefs and knowledge.



Artist: Dr Richard Barry Walley OAM

# Programming and BI

- So far, we have utilised 'built-in' (internal) visuals within Power BI Desktop and used its built-in data preparation, analysis and visualisation features.
  - What if we want to build something more complicated or more advanced?
- Power BI Desktop allows us to write our own visuals and analysis calculations, using programming language code (in either R or Python) to extend the above capabilities of the software to enable us to do more than what is built in.
  - This is achieved through the Power Query Editor or through specific Visualisations.

# R and Python

- Two different programming languages can be used for these purposes.
  - They are R and Python; you may be familiar with one, both or neither.
  - Both are frequently used for data analysis and visualisation;
  - Similar logic is used with the code in both languages, such as the use of matplotlib as option for visualisation and the use of data frames (or similar) to manipulate data.
- We will only be covering the very basics of R, so don't be too concerned.
  - In reality, you can mix and match which one that you use;
  - The code you will use will be supplied and explain – you can learn more in other units.

# Things we can do

- (More) advanced analytical calculations, compared to basic built-in statistics.
  - A wide range, but for example: correlation coefficients, conditionals, machine learning, artificial intelligence - although there is a limited bit of this built into Power BI already.
  - As we have control of the code, there is generally more customisability, but at the expense of usability (that is, you have to know what code to write and how to write it!)

# Things we can do (Part II)

- Data cleaning/wrangling/formatting/preparation, above Power Query Editor.
  - Next week, we will look at the complexity of building the dataset within Power BI - this could be a lot simpler and easier to understand using code (or, more difficult?)
  - Data sets generated using code in Power BI Desktop function the same way as ones created in the Power Query Editor – in fact, they are still manipulated with it!
  - Can do the other preparation tasks we can do within the Power BI software, but less limited in customisability – again, at the expense of usability.

# Things we can do (Part III)

- More options for visualisation than the built-in (or even third party) options.
  - Using the powerful matplotlib library, many different kinds of visuals can be created and customised to a very fine level – although they are not interactive.
  - Can also use the seaborn package for further visualisations - very 'pretty' and some unusual ones, plus very easy to use (compared to matplotlib) but still powerful.
  - Theoretically with these languages, we could do lots more – but we're limited to specific packages in Power BI as it needs to 'capture' the output of the visualisations.

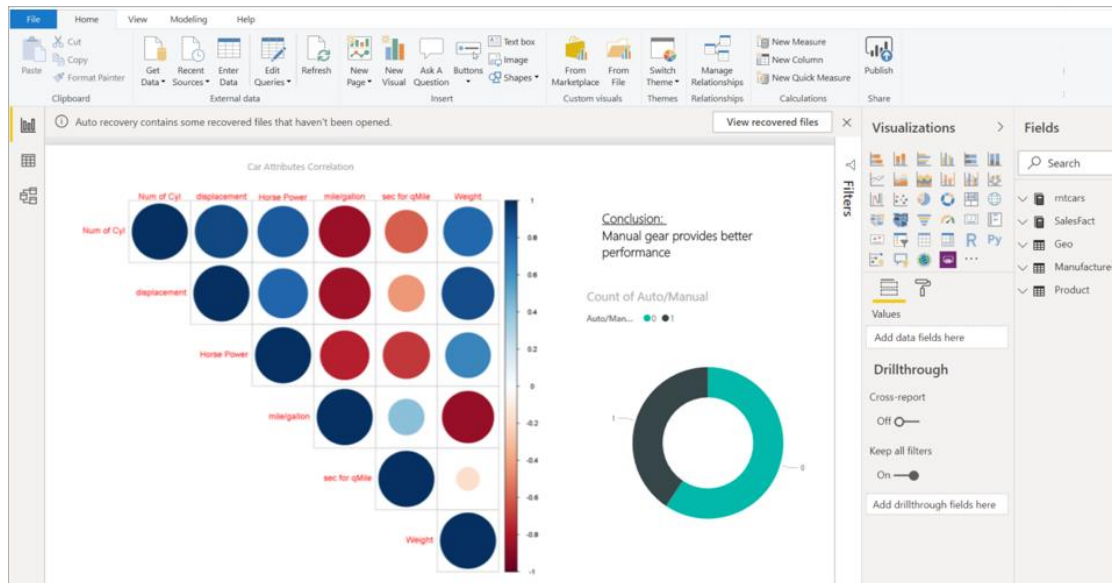
# Practical considerations

- The R and Python interpreters (the programs that turn the source code into outputs) as well as the required packages have to be installed to utilise them within Power BI.
  - This is more work for the end user – consider distribution of your dashboard;
  - Version incompatibilities – Python 2 vs Python 3 and smaller issues as well.
- This is easier said than done – talking from experience!
  - Ironically, it's easier with Power BI Online in this case.



# An example with R

- We will be working with R for this week's practical – although we could use Python.
- You'll be building something similar to what you see in the screenshot below.



Source: Microsoft

**Don't worry if you haven't used R before – we will be dealing with very simple and basic code.**

# How to do it?

- Like I just said, do not worry if you are new to programming – there is not that much of it, just a couple of lines of code that will be supplied in the practical!
  - A very basic primer on programming (for those new to it) will be provided.
  - It is a simple workflow in itself; it just involves setting a few different options.
  - We will be building off a single dataset that is regarding car performance.

# A Reminder about R

- In this week's activity, we will be exploring the creation, customisation and utilisation of script visualisations powered by R, that also involve some advanced analysis.
- If you are utilising Power BI Desktop using UniApps, then there is nothing else that you need to do – it has been tested and works with R having been included.
- Otherwise, if you have not installed it, this is a free download and an installer is available from the following URL: <https://cran.r-project.org/bin/windows/base/>.

# Data Cleaning – Dummy Vars

- We can use the 'Pivot Column' option under the Power Query editor to convert categorical variables into dummy variables, or vice-versa, if it is required.
  - This can sometimes assist in the analyses and visuals we create.
  - When creating dummy variables, a variable/attribute with X values is transformed into one less than X values of **True** / **False** – as all **False** can represent a particular value.
- Think about this when visualising or interpreting data – especially correlation!

# Correlation Options

- There are many options which we can pass via our R script code to the **corrplot** (correlation plot) to adjust its visual appearance – this is explained in the practical!
  - Each time you change the details / options, you must refresh!
- Replace the last line of code with the lines of code below (one and a time) and determine what each of the different options does. This is not an exhaustive list!
  - `corrplot(corr_matrix, method = "circle", type = "upper", tl.srt = 0)`
  - `corrplot(corr_matrix, method = "circle", order = "hclust", addrect = 2)`
  - `corrplot(corr_matrix, method = "color", tl.cex = 0.6, tl.srt = 45, tl.col = "black")`
  - `corrplot(corr_matrix, method = "ellipse", type = "upper", tl.cex = 0.6, tl.srt = 45, tl.col = "black")`

# A Note On Correlation

- It is also worthwhile to note that correlation with binary variables is not particularly statistically worthwhile due to the limited variability in the values.
  - In other words, it will be either **1** or **0** – not much variability!
- As such, it may be more appropriate to create a bar chart visual comparing some other values for the two groups within your report.



# Topic Eight: Connection Types

## INMT5526: Business Intelligence

# Imported vs Connected Data

- Imported data provides us a 'snapshot in time' with a normal file (residing on our system) that is only changed by 'us' – such as a spreadsheet or similar.
  - This is useful if we want to consider a snapshot in time – dashboard shows the same thing.
- Connected data comes from a remote system and can be updated as new events occur and the data source is updated.
  - Connected data allows us to refresh the dashboard and update it with the latest data.
  - Enables us to see how/if our conclusions change as the world changes around us.



# Uses for Connected Data

- Many sensor systems are connected to the internet these days, that observe the world or other systems around us, utilised within business and other contexts.
  - For example, 'internet of things' sensors to monitor weather, presence of something in a certain location (such as road volume or customers in premises).
- Other 'traditional' systems are either directly connected to the dashboard, or are first transformed and/or aggregated into another format.
  - For example, sales from cash registers, stock exchange data...
  - We could connect our MySQL databases we have been working on during semester!

# Why not always connected data?

- Generally, it pays to use connected data to make short term decisions – for example, think of same-day, real-time or something close to it.
  - Can see 'the state of play' in almost real-time – adjust and make decisions quickly.
- However, this comes with complexity and risk.
  - What if the data source is not working?
  - Need for transformation of the data source into the model format we are using.

# Practically, in Power BI

- We covered how to integrate 'traditional' spreadsheet formats using Power BI Desktop in last week's workshop.
  - Next week, we will attempt to connect to our MySQL server that we used before!
  - We will also connect to a web-based source – with very complex data!
- However, it also provides for a connection to many other different systems.
  - Generally, databases and other analytical tools or various data management systems.
  - We can query the data, integrate it with others or use their outputs in our visuals.
  - There are limitations with big data – but we can pre-process or limit query sizes!

# Things you can connect to

- Many different options and systems that can integrate dashboard/report generation into other business workflows – for example, Google Analytics.

Source: Microsoft

Excel Workbook	PostgreSQL database	Datamarts (preview)	Power BI dataflows (Legacy)	Data.World - Get Dataset (Beta)	Microsoft Exchange
Text/CSV	Sybase database	Warehouses	SharePoint Online List	GitHub (Beta)	Hadoop File (HDFS)
XML	Teradata database	Lakehouses	Microsoft Exchange Online	LinkedIn Sales Navigator (Beta)	Spark
JSON	SAP HANA database	KQL Databases	Dynamics 365 Online (Legacy)	Marketo (Beta)	Hive LLAP
Folder	Azure SQL database	Azure SQL database	Dynamics 365 (Dataverse)	Mixpanel (Beta)	R script
PDF	SAP Business Warehouse Application Server	Azure Synapse Analytics SQL	Dynamics NAV	Planview Portfolios	Python script
Parquet	SAP Business Warehouse Message Server	Azure Analysis Services database	Dynamics 365 Business Central	QuickBooks Online (Beta)	ODBC
SharePoint folder	Amazon Redshift	Azure Database for PostgreSQL	Dynamics 365 Business Central (on-premises)	SmartSheet	OLE DB
SQL Server database	Impala	Azure Blob Storage	Common Data Service (Legacy)	SparkPost (Beta)	Acterys : Model Automation & Planning (Beta)
Access database	Google BigQuery	Azure Table Storage	Azure DevOps (Boards only)	SweetIQ (Beta)	Actian (Beta)
SQL Server Analysis Services database	Google BigQuery (Microsoft Entra ID) (Beta)	Azure Cosmos DB v1	Azure DevOps Server (Boards only)	Planview Enterprise Architecture	Amazon Athena
Oracle database	Vertica	Azure Data Explorer (Kusto)	Salesforce Objects	Zendesk (Legacy) (Beta)	Amazon OpenSearch Service (Beta)
IBM Db2 database	Snowflake	Azure Data Lake Storage Gen2	Salesforce Reports	Web	Anaplan
IBM Informix database (Beta)	Esbase	Azure HDInsight (HDFS)	Google Analytics	SharePoint list	Aptix Insights (Beta)
IBM Netezza	AtScale Models	Azure HDInsight Spark	Adobe Analytics	OData Feed	Asana (Beta)
MySQL database	Power BI semantic models	HDInsight Interactive Query	appFigures (Beta)	Active Directory	Assemble Views
AtScale cubes	Dataflows	Equi5	InterSystems IRIS (Beta)	Planview ProjectPlace	SumTotal
Autodesk Construction Cloud	CloudBluePSA (Beta)	eWay-CRM	Janit Pro (Beta)	Dataflows	Supermetrics (Beta)
Automation Anywhere	Cognite Data Fusion	Exact Online Premium (Beta)	Jethro (Beta)	Product Insights (Beta)	SurveyMonkey
Automy Data Analytics (Beta)	Dataverse	Exasol	Kognitwin	Profilise	Azure Synapse Analytics workspace (Beta)
Azure Cost Management	Azure Cosmos DB v2 (Beta)	FactSet Analytics	Kyligence	QubolePresto (Beta)	TeamDesk (Beta)
Azure Resource Graph	Dynamics 365 Customer Insights (Beta)	FactSet RMS (Beta)	LEAP (Beta)	Quickbase	Microsoft Teams Personal Analytics (Beta)
Azure HDInsight on AKS Trino (Beta)	Azure Databricks	FHIR	Linker PICK Style / MultiValue Databases (Beta)	Roamler (Beta)	Tenforce (Smart)List
Solver	Data Virtuality LDW	Palantir Foundry	LinkedIn Learning (Beta)	Samsara (Beta)	TIBCO(R) Data Virtualization
BI Connector	Digital Construction Works Insights	Funnel	MariaDB	SIS-CC SDMX (Beta)	Usercube (Beta)
BitSight Security Ratings	Delta Sharing	Google Sheets	MarkLogic	Shortcuts Business Insights (Beta)	Vena
BQE CORE	Denodo	Hexagon PPM Smart® API	MicroStrategy for Power BI	SingleStore Direct Query Connector	Vessel Insight
Bloomberg Data and Analytics	Dremio Software	Indexima	MongoDB Atlas SQL	Siteimprove	Webtrends Analytics (Beta)
Wolters Kluwer CCH Tagetik (Beta)	Dremio Cloud	Industrial App Store	OneStream (Beta)	SoftOne BI (Beta)	Witvivo (Beta)
CDData Connect Cloud	Eduframe (Beta)	Information Grid (Beta)	OpenSearch Project (Beta)	SolarWinds Service Desk	Viva Insights
Celonis EMS	Emigo Data Source	Intune Data Warehouse (Beta)	Paxata	Planview IdeaPlace	Wrike (Beta)
Cherwell (Beta)	Entersoft Business Suite (Beta)	inwink (Beta)	Planview OKR (Beta)	Starburst Enterprise	Zendesk (Beta)



# Topic Eight: Spatial Data and BI

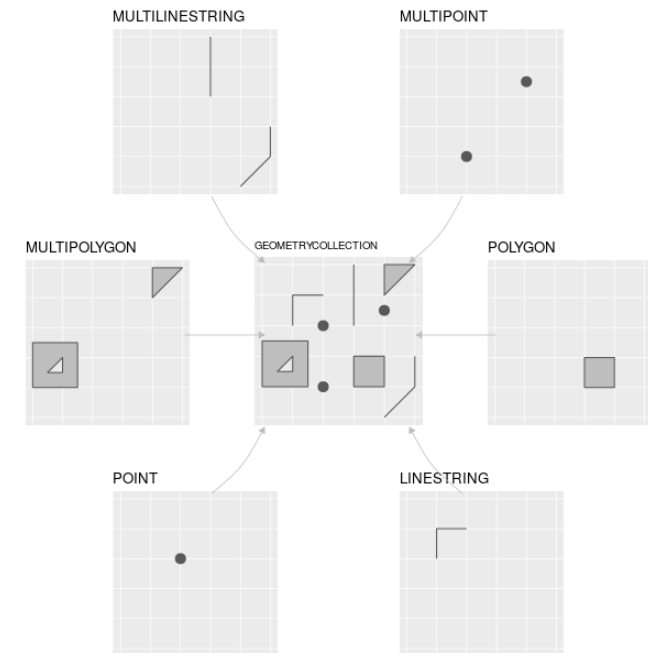
## INMT5526: Business Intelligence

# Why spatial?

- Think back to our discussions regarding data analysis.
  - Often data has some 'location' attached to it, as well as other attributes / properties.
  - This might be quantitative (such as latitude and longitude) or effectively categorical (at least text-based – such as country, state, city, campus or building).
  - Why is this of use to us? What benefit does it provide over other data in the same format?

# Revision: Spatial data types

- Primarily, spatial data falls into three categories:
  - Point:** a location with no area representing a single thing, e.g. the UWA Business School is located at 31.9859 S and 115.8207 E;
  - Polygon:** a location represented by a shape with area, made up of multiple lines, e.g. the coastline of Australia, the boundaries of UWA;
  - Line:** made up of multiple points, can be used to represent roads and similar. Least likely to run into.



Source: Geocomputation  
with R

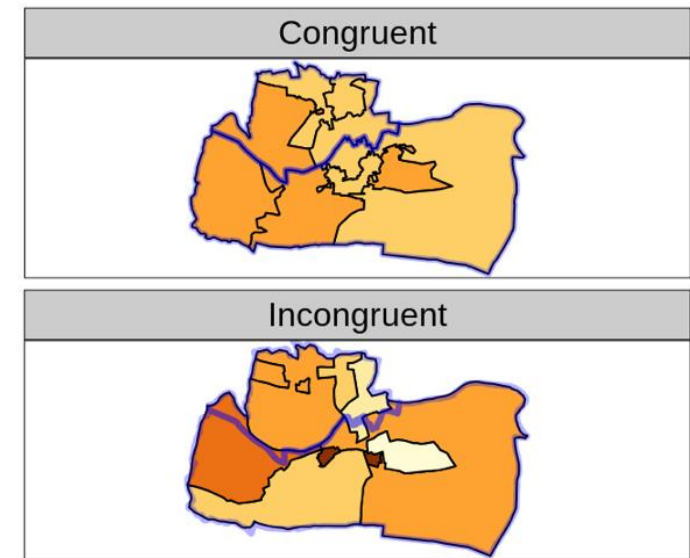
# Spatial operations

- Generally, the most useful spatial data to us is that recorded as polygons, since they enable the richest analysis to be undertaken as they are most descriptive.
- However, we can do spatial operations on “all” data types, such as:
  - *Topological*: do these two areas intersect and by how much? Also, are the areas/lines/points within each other, disjoint, touching, what is the distance from each?
  - *Joining*: which stores (points) fall within which country (polygon)? Which points are nearby and/or are possibly the same as each other?



# Spatial operations (Part II)

- There's (loads) more we can consider, but this is designed to be a quick primer/overview:
  - Raster data: think of obtaining patterns out of pictures;
  - More complex and advanced distance-based operations than what is described previously;
  - Aggregation of smaller areas into larger ones and incongruent areas ('re-aggregation') – issues!



Source: Geocomputation  
with R

# More spatial considerations in BI

- We can be much more advanced with our use of spatial data within Power BI, compared to the bubble-based plot that we did last week:
  - Point-based data, rather than categorical country-based;
  - Drilling up and down between regions in a country;
  - Choropleth (colour shaded) maps.
- Some of these require built-in and external third-party visuals, such as maps sourced from Mapbox and ESRI – or can be achieved using script visuals.
  - In the practical, we will look at what we can achieve with built-in options.

# Adjusting the Map

- In the practical, we will change the bubble map into a filled (choropleth) map.
  - This will make it easier for us to observe differences between countries.
  - Therefore, we must choose an appropriate colour scale that is 'noticeable'.
  - While aesthetic considerations still apply, we should ensure we are not misrepresenting things – much in the same way the bubble map did with the sizing of the bubbles!
  - The filled map does have its own issues, which will be discussed during the practical.
  - We should ensure that we consider the context of our map use – e.g. for colour blindness.

# The End: Thank You

Any Questions? Ask via email ([tristan.reed@uwa.edu.au](mailto:tristan.reed@uwa.edu.au))