Table of Contents

Terms of Reference

An assignment report submitted in fulfilment of the requirements for Course B8IT108 (Data and Web Mining), Higher Diploma in Data Analytics, Dublin Business School. The report will be read by Dublin Business School staff. The report was written to develop the student's understanding of open data, data mining techniques, algorithms, CRISP-DM analysis techniques and the use of scripting tools and RapidMiner software. Lecture notes, text books and online resources were used in the development of this report.

Summary

Data mining is the process of sorting through large data sets to discover patterns and establish relationships to solve problems through data analysis. It is an interdisciplinary subfield of computer science and involves methods from the fields of machine learning, statistics, and database systems. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Data mining tools allow enterprises to predict future trends.

A public bicycle system, or bike-share scheme, is a service in which bicycles are made available for shared use to individuals on a very short-term basis. They offer a sustainable mode of transport that can reduce car use, congestion and emissions.

Data analysts across the globe are interested in the data created from bike-share schemes. The data is invaluable and provides insight on urban dweller behaviour and mobility in cities.

Dublinbikes is a public bicycle rental scheme which has operated in the city of Dublin since 2009. At its launch, the scheme, which is sponsored by JCDecaux, used 450 bicycles with 40 stations. Today, there are circa 1,580 bikes and 102 stations. Dublin was the 17th city to implement a bike-share scheme. The benefits of bike sharing schemes include transport flexibility, reductions to vehicle emissions, health benefits, reduced congestion and fuel consumption, and financial savings for individuals.

In this assignment, the following features of the Dublin bike-share scheme is analysed:

- Use patterns
- The impact of weather
- Characteristics of bike stations

This information would be invaluable to decision-makers (elected members, urban planners, transport modellers) to underpin policy and guide strategic development.

1.0 Introduction

This report consists of three sections – selecting the data, determining the business questions and applying the CRISP-DM methodology to analyse the data.

The *'Dublinbikes'* data was the open data source of choice for this assignment. The dataset consists of static data and dynamic data. Static The data is accessible through the 'data.gov.ie' portal[1] and the data sharing platform 'Dublinked'[2]. The dynamic data is available through the JC Deceaux API and the CityBikes API.

The Dublinbikes dataset was selected because of my interest in urban transport planning & mobility, spatial analytics, the openness of the data, the extent of academic work and media articles on the subject and the visibility of Dublinbikes on the streetscapes of Dublin city centre.

I developed my 4 business questions based on some research on the subject, business questions used in similar studies and my curiosity on the subject.

The *'Cross-industry standard process for data mining'*, (CRISP-DM) methodology was used to analyse the data. CRISP-DM is a comprehensive methodology and process model that provides a complete blueprint for conducting a data mining project. CRISP-DM breaks down the life cycle of a data mining project into six phases; – business understanding, data understanding, data preparation, modelling, evaluation and deployment. The framework allows for projects to be replicated, aids project planning and management, is tool neutral, focuses on business issues and technical analysis.

2.0 Selecting the Data

I chose the Dublinbikes data for this assignment because I have an interest in smart cities, spatial analytics, transport planning and mobility. I also chose the data because of its openness and the creative commons licence which has already enabled the creation of many useful apps, data exploration and the deployment of the analysis on multiple platforms.

There are many bike share schemes across the globe. The data generated by bike share systems is attractive for researchers because the duration of travel, departure location, arrival location, and time elapsed is openly recorded[3]. Bike sharing systems therefore function as a sensor network, which can be used for studying mobility in a city.

Weather data was required to help answer some of the business questions on the Dublin bike share scheme. Met Éireann[4] and GitHub[5] were selected as the sources of weather data.

[1] https://data.gov.ie/dataset/dublinbikes (data is harvested from the Dublinked platform)

[2] https://data.dublinked.ie/dataset/dublinbikes

[3] The level and variety of data available to the public varies amongst the schemes

[4] http://www.met.ie/climate-request/

[5] https://github.com/jameslawlor/dublin-bikes-timeseries-analysis

2.1 Background / Context of Data Subject

2.1.1 Bike share schemes

A bike-share scheme is a public transportation program, that facilitates short term bicycle rental at well placed, unattended stations. Bike sharing is ideal for short distance point-to-point trips providing users the ability to pick up a bicycle at any self-serve bike-station and return it to any other bike station located within the system's service area. Bike sharing takes many forms.

In cities where such schemes are particularly successful, it seems that a new paradigm in short distance transport is emerging. These schemes are a realistic alternative to the bus, tram, taxi or private car, and have infinite benefits to the individual and the environment. The benefits of bike sharing schemes include transport flexibility, reductions to vehicle emissions, health benefits, reduced congestion and fuel consumption, and financial savings for individuals. Bike-share schemes can also act as a catalyst for increased bicycle use, by normalizing cycling as a means of getting around a city.

2.1.2 Dublinbikes

Dublinbikes is a public bike-share system, where the local authority, Dublin City Council, is engaged in the funding, managing, administering and/or permitting the bike-share program. The scheme is in operation since 2009 and was launched with 450 bicycles and 40 stations. Today, there are circa 1,580 bikes and 102 stations. JCDecaux, the outdoor advertising company, has been operating the Dublinbikes contract on behalf of Dublin City Council since its launch. The scheme is funded through a mixture of public funding, private funding and advertising space.

Some of the biggest challenges facing Dublin and other cities across the globe include mobility, environment, energy, as well as better data and communication. Dublin Bikes is connected to each of these challenges. Thus, there is a motivating focus to analyse such data.

2.1.3 Cycling in Dublin

People chose to cycle bikes in cities for many different reasons. Some of the primary reasons include finance, health, environmentalism and time efficiency. Boosting fitness, enhancing health, reducing stress, saving money, saving the environment and reducing commute times are all examples of benefits.

Cleaner modes of transport (cycling and walking) have become popular in many cities across the globe. Finding new ways to support and facilitate more cycling and walking in Dublin is outlined in many strategic reports produced by Dublin City Council, the National Transport Authority and the Department of Transport. Tax saver schemes and bike share schemes are just some of the policies used to encourage cycling. The bike-share scheme that is established in Dublin is based on a scheme with dedicate docking stations. Stationless / dock-less bike share schemes are currently being considered by Dublin City Council. The adjacent local authority Dun-Laoghaire Rathdown County Council are trialling a stationless bike-share scheme[6]. To grow the bike-share schemes, more insight about current use patterns and possible future use patterns is required. This need for more insight is the basis of my study.

[6] http://ecouncil.dlrcoco.ie:9071/documents/s56127/Stationless%20Bikes%20Sept%202017.pdf

2.1.4 Dublin Climate

The climate of Dublin is like much of the rest of north-western Europe. Dublin experiences a maritime climate with cool summers, mild winters, and a lack of temperature extremes. The average maximum January temperature is 8.8 °C, while the average maximum July temperature is 20.2 °C. On average, the sunniest months are May and June, while the wettest month is October with 76 mm of rain, and the driest month is February with 46 mm. Rainfall is evenly distributed throughout the year. Dublin's sheltered location on the east coast makes it the driest place in Ireland, receiving only about half the rainfall of the west coast. The city experiences long summer days and short winter days. Strong Atlantic winds are most common in autumn. These winds can affect Dublin, but due to its easterly location it is least affected compared to other parts of the country. However, in winter, easterly winds render the city colder and more prone to snow showers.

2.1.5 Cycling and weather

Adverse weather is often cited as a reason why people do not cycle. The effect of both (short-term) weather conditions and (long-term) seasonal variation patterns on bicycle commuting patterns is of interest to policy makers and decision makers. It is often assumed that certain conditions are perceived by cyclists to make commuter cycling non-viable, and thus lead to a significant drop in numbers on days or periods when these conditions persist.

2.2 The Data

2.2.1 Dublinbikes Data

The Dublin Bikes data is comprised of real time and static data. The static data is available on the Dublinked data portal. Both types of data are available on the JC Deceaux API platform[7]. The static data consists of stable information like station position, number of bike stands and payment terminal availability. The dynamic data consists of station state, number of available bikes and number of free bike stands. The dynamic data are refreshed every minute.

2.2.2 Weather Data

The weather data was sourced from Met Éireann[8] and GitHub[9]. Hourly, daily and monthly resolution data is available on the Met Éireann website. There are 15 weather stations in Dublin. However, only 5 of the stations are within proximity of the spatial extent of the Dublinbikes scheme. (Glasnevin, Merrion Road, Ringsend, Simmonscourt and Phoenix Park). The information recorded at the 5 station varies. The Phoenix Park Weather Station is an automatic station and contains the most comprehensive weather information and was therefore selected as the weather station of choice for this study (Although, the full available data was not required, this station was selected in case future analysis might require some of the extended weather data). The data available at this weather station includes the following:

- Precipitation (mm)
- Mean air temperature (C)

[7] https://developer.jcdecaux.com/#/login
https://api.citybik.es/v2/

[8] http://www.met.ie/climate-request/

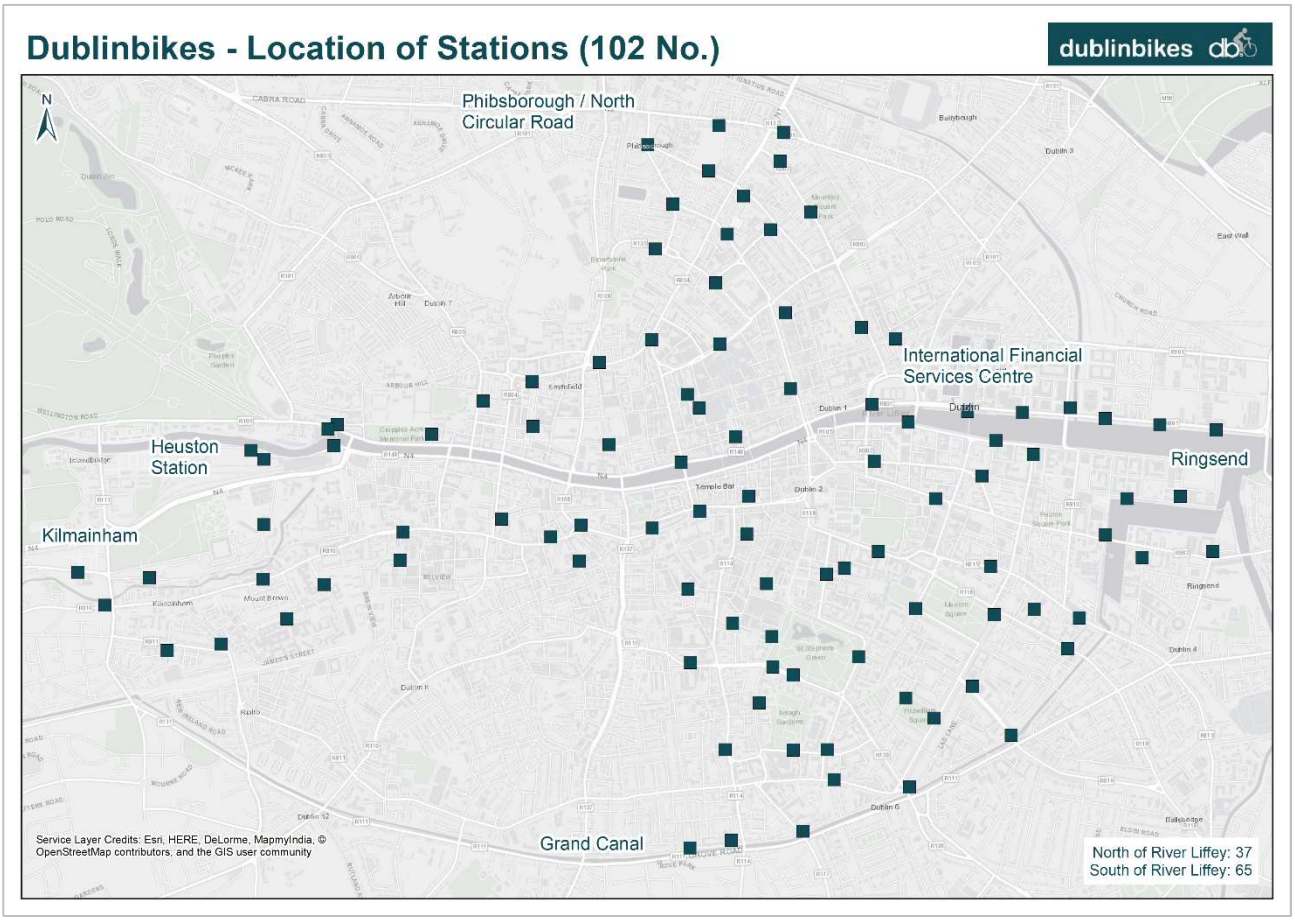
[9] https://github.com/jameslawlor/dublin-bikes-timeseries-analysis

- Maximum air temperature (C)
- Minimum air temperature (C)
- Mean max temperature (C)
- Mean minimum temperature (C)
- Grass minimum temperature (C)
- Mean wind speed (knot)
- Highest gust (knot)
- Sunshine duration (hours)

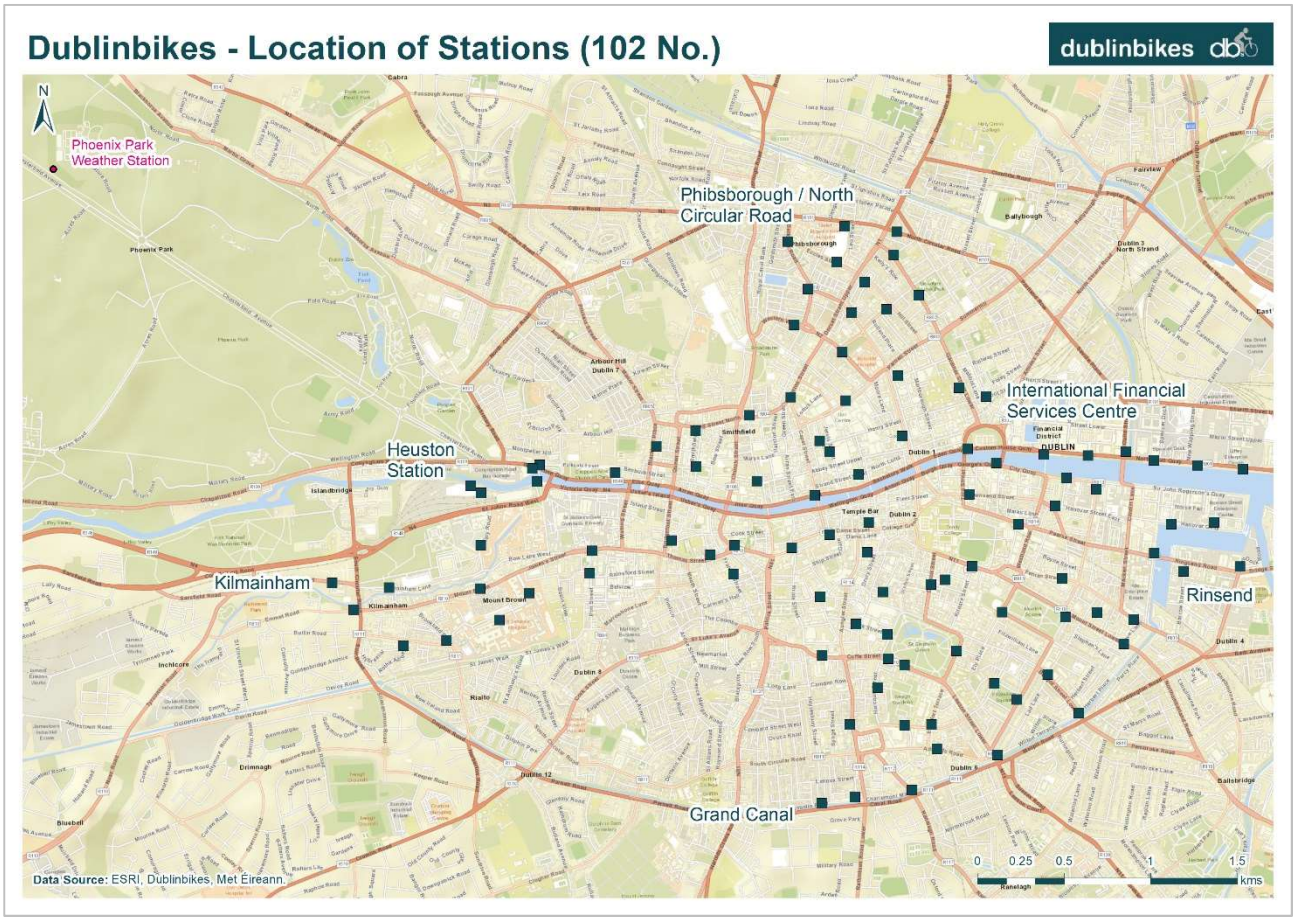
The GitHub data contains categorical weather data, quantitative (and continuous) temperature and wind speed data.

2.3 Spatial Context of Data

The existing Dublinbike stations are located within the administrative area of Dublin city council, more specifically Dublin city centre. The stations extend to Phibsborough in the north, Ringsend in the east, the Grand canal in the south and Kilmainham in the west.



The Phoenix Park weather station is located north-west of Dublin city centre (please see map below). The 2 maps were generated in ESRI ArcMap. The static data was added to the GIS software using the 'add XY data' tool.

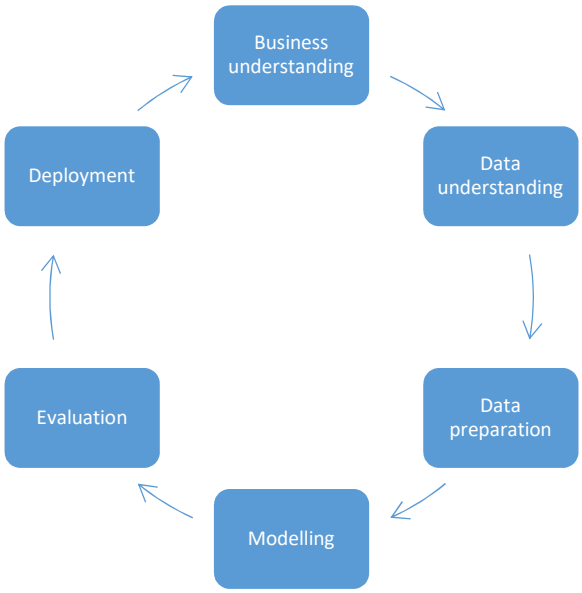

Dublinbikes - Location of Stations (102 No.)
dublinbikes
N
Phoenix Park
Weather Station
Phibsborough / North
Circular Road
International Financial
Services Centre
Heuston
Station
Kilmainham
Rinsend
Grand Canal
Data Source: ESRI, Dublinbikes, Met Éireann
0 0.25 0.5 1 1.5
kms

3.0 CRISP-DM

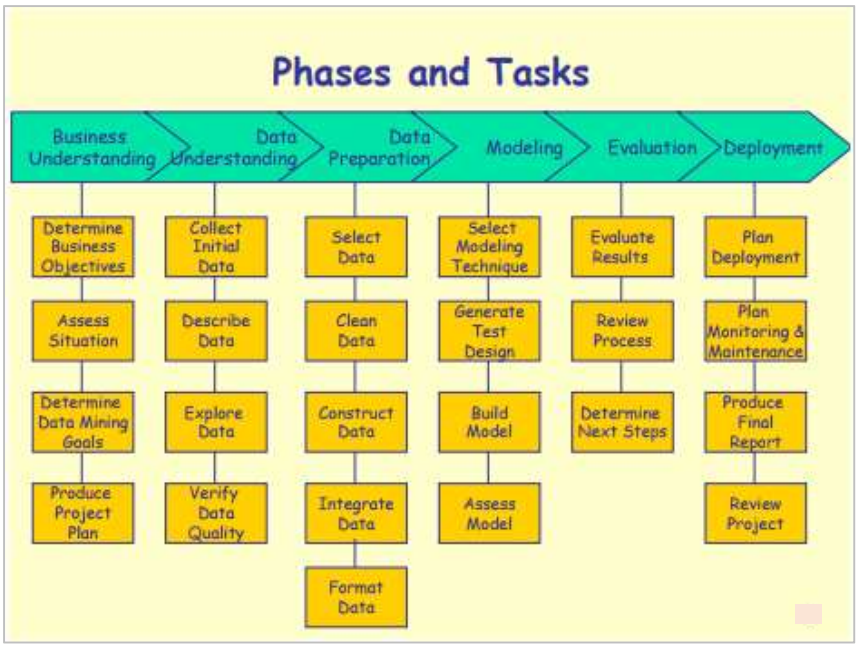
CRISP-DM stands for *'cross-industry process for data mining'*. The CRISP-DM methodology provides a structured approach to planning a data mining project. It is a robust and well-proven methodology.

This model is an idealised sequence of events. In practice many of the tasks can be performed in a different order and it will often be necessary to backtrack to previous tasks and repeat certain actions. The model does not try to capture all possible routes through the data mining process.

CRISP-DM breaks the process of data mining into six major phases / stages: business understanding; data understanding; data preparation; modelling; evaluation and deployment.



The image below details the breakdown of the phases / steps into tasks. As this is an academic assignment, all the steps outlined in the diagram below might not be apt for this task and were therefore omitted.



3.1 Stage 1 (CRISP-DM) – Business Understanding

3.1.1 Determine Business Objectives

The first stage of the CRISP-DM process is to understand what needs be to accomplish from a business perspective. The goal of this stage of the process is to uncover important factors that could influence the outcome of the project[10].

This study is being carried out to gain insight into the use of the Dublinbikes scheme to develop a business case for further investment. This study will also provide some ancillary information on mobility in the city. The business objectives developed for this study include the following:

1) To identify if there is a different pattern of use on different days (weekdays V weekends)
2) To identify if there is a different pattern of use at different times throughout the day
3) To identify if there are similar characteristics amongst the stations (i.e. busy and quiet stations)
4) To identify if bike use patterns can be accurately predicted using weather values

Some business objectives were eliminated at this stage, as initial review of the data revealed that the data did not contain answers to the questions and some of the objectives were beyond the scope of this project

- To identify the age and gender profile of users (This data is not available in the Dublinbikes scheme. Data protection might be the reason why the Dublinbikes dataset has limited fields of data compared to other schemes)
- To identify the difference in use patterns between casual and registered users (this data is not available to the public)
- To identify if term time effects use (initial data prep and model testing revealed that the data processing required a lot of computing power, therefore the dataset was stripped down and there was no scope to analyse term time data in this project)
- To identify some of the contributing reasons behind the success of the scheme (outside scope of data mining project - role of decision makers – external issues need to be considered)
- To identify if there is a possibility of co-location of facilities with other transport providers (outside scope of data mining project - role of decision makers – external issues)
- To identify if topography has an impact on patterns of use (outside the scope of this project)

3.1.2 Assess the situation

The problem to be addressed is limited insight on the Dublin bike share scheme. The key stakeholders of this information include the local authority and adjacent local authorities, public transport authorities and the sponsors of the scheme (Dublin City Council, Dun-Laoghaire Rathdown, South Dublin County Council, the National Transport Authority, JC Decaux and Just Eat[11]). The problem is suitable for an analytics solution because there is a vast amount of numerical data on the subject. While there is anecdotal evidence on how people use the scheme, an analytics solution can verify the evidence and therefore underpin strategic planning in the city and provide a more robust case for additional financial investment. The redistribution of bikes is a problem with many bike-share schemes. Inferring the potential destinations and arriving time of each individual trip beforehand can effectively help the service providers schedule manual bike re-dispatch in advance. Analysing the current data could help with this problem.

This study is being carried out by 1 individual as part of an assignment to learn about data mining and the CRISP-DM process. Lack of expert knowledge and time are constraints. Risks include the inability to apply the

[10] *Neglecting this step can mean that a great deal of effort is put into producing the right answers to the wrong questions.*

[11] Just Eat joined as a commercial partner on a 3-year agreement, on 20th July 2017, replacing Coca-Cola Zero

methodology, issues with software, issues with hardware, resource availability (working on other tasks) and access to the right data.

The raw data used in this project, is freely available and therefore any work carried out is bound by creative common licence. All data sources will be acknowledged.

3.1.3 Determine data mining goals

A business goal states an objective in business terminology. A data mining goal states a project objective in technical terms. Often when business and data analysts work together, to define a technical solution to the business problem, there can be miscommunications. It is important to define and document how the business problem is translated into a data mining goal. Below, I have documented the business objective and the corresponding data mining goals.

1) The business objective *'To identify if there is a different pattern of use on different days'* can be translated into the data mining goals:
    - Carry out exploratory data analysis and examine histograms and statistics at the data exploration stage and at the model building stage.
      Using charts to visualize the findings

2) The business objective *'To identify if there is a different pattern of use at different times throughout the day'* can be translated into the data mining goals:
    - Carry out exploratory data analysis and examine histograms and statistics at the data exploration stage and at the model building stage.
    - Using charts to visualize the findings

3) The business objective *'To identify if there are busy and quiet stations'* cane be translated into the mining goals:
    - Building a cluster model using K-means clustering to identify stations with similar use pattern
    - Verifying the performance of the cluster model using the Davies Bouldin index and the average within centroid distance and examining the visual output

4) The business objective *'To identify if bike use patterns can be accurately predicted'* can be translated into the following data mining goals:
    - Build a model using linear regression and neural network operators to obtain prediction data and test it against data that was not put into the model
    - Building a linear regression model using available weather and bike data to identify if there is a relationship / correlation between weather and bike use patterns
    - Verifying the performance of the model by examining the performance indicators and developing models with different algorithms to identify the best performing model

3.2 STAGE TWO – DATA UNDERSTANDING

The second stage of the CRISP-DM process requires project the acquisition of the data listed in the project resources. This initial collection includes data loading, if this is necessary for data understanding. For example, if you intend to use a specific tool for data understanding, it is good practice to load the data into this tool. If the project requires multiple data sources, then it is necessary to consider how and when to integrate these.

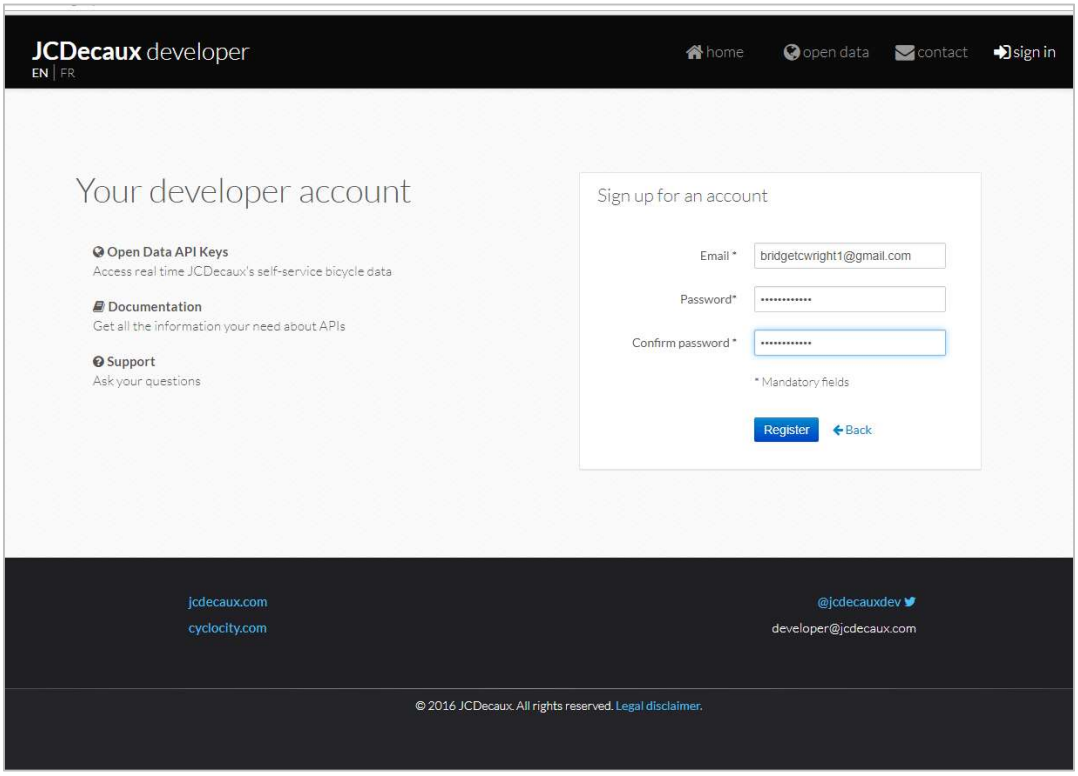
List the data sources acquired together with their locations, the methods used to acquire them, and any problems encountered. Record problems encountered, and any resolutions achieved. This will help both with future replication of this project and with the execution of similar future projects.

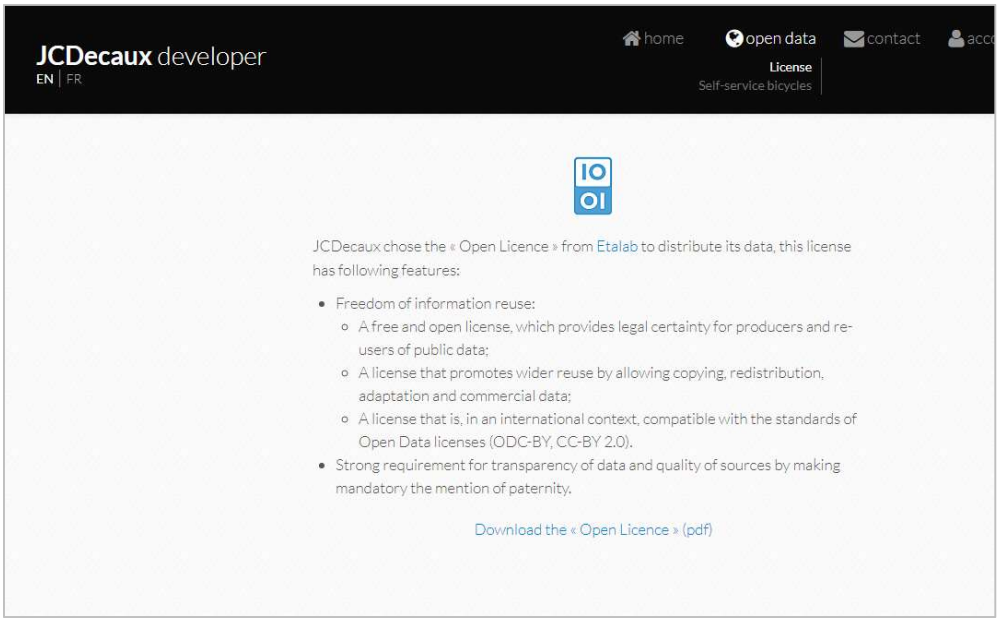
3.2.1 Collecting initial data

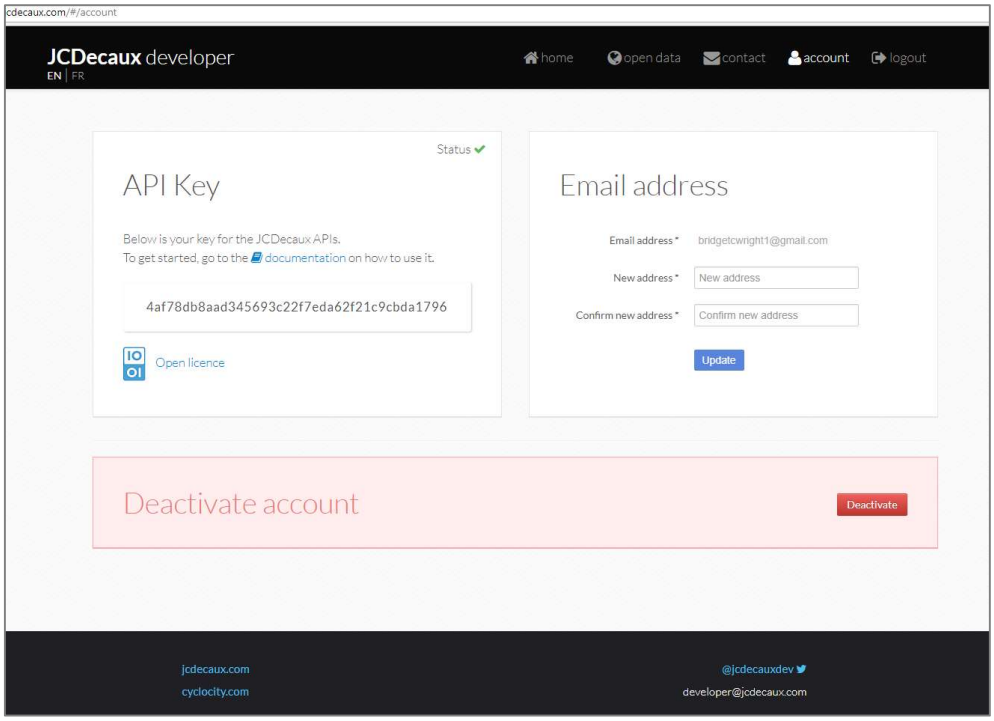
As mentioned earlier in this report, there is static and dynamic Dublinbikes data. Static data provides stable information like station position, number of bike stands and payment terminal availability. Dynamic data provides station state, number of available bikes and number of free bike stands. Static data can be downloaded manually in file format or accessed through the JC Deceaux Application Programming Interface (API). Dynamic data are refreshed every minute and can be accessed only through the API. Static data doesn't require an API key. For equal accessibility to dynamic data, a personal API key and account was required. I created an account on Sat 14th October 2017 circa 16.25pm. I agreed to the terms and conditions of the open licence agreement and gained the following API key:
'4af78db8aad345693c22f7eda62f21c9cbda1796'.

A personal API is not required for the CityBikes API. However, the creators of this API state that users of the data most acknowledge their API in projects.

Below, I have included some screen dumps of my experience setting up my user account on the JC Decaux API.







Static Dublinbikes Data

The static Dublinbikes data was downloaded from the Dublinked data portal.

Dynamic Dublinbikes Data

The dynamic Dublinbikes data was scraped from the JC Decaux API[12] using a python script and the CityBikes API[13]. The data was scraped over 8 days at 2-minute intervals from the JC Decaux API and 5-minute intervals from the CityBikes API. The same data was scraped from each API. The reason behind this was for academic learning purposes and in case there were issues with 1 of the APIs. Please see Appendix A for the python code that was used to scrape the data.

GitHub Dublinbikes Data

During the initial stages of my research for this project, I came across the James Lawlor Dublin bikes timeseries analysis project on GitHub[14]. This repository contains much of the data that I needed for my project. After careful consideration and examining my business objectives and data mining goals, I decided that a week of scraped data was not going to be able to answer my business questions. My timeframe on my project was tight and I could not wait any longer to scrape more data. I decided to use the data in James Lawlor's GitHub repository. It consisted of CSV files for each day. It contained data from the end of January 2017 to mid-August 2017.

Met Éireann and Github Weather Data

The weather data was sourced from Met Éireann[15] and GitHub (same repository mentioned above). Hourly, daily and monthly resolution data is available on the Met Éireann website. There are 15 weather stations in Dublin. The Phoenix Park Weather Station is an automatic station and contains the most comprehensive weather information and was therefore selected as the weather station of choice for this study. Scraping was not necessary to obtain this data. The data was easily and quickly obtained in an Excel CSV format from the Met Éireann website.

After I collected weather data from the Met Éireann website, I collected weather data from the James Lawlor GitHub repository. The data was contained in CSV files, a different CSV file for each day. James Lawlor noted on GitHub that the weather data was scraped from weather.com's API (I checked this website and the service is no longer available, so future projects on Dublinbikes will require the weather observation data to be sourced from Met Éireann or the Wunderground weather API[16] indicated on the landing page of the weather.com website).

3.2.2 Describe data

Static Dublinbikes Data

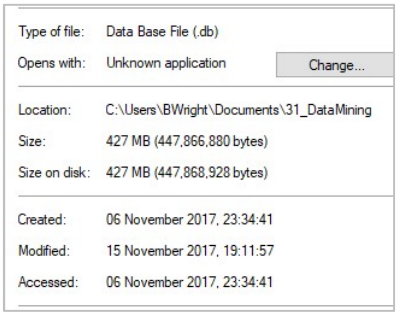
This consisted of a 5kb CSV file with 102 records representing each station (as at 11/09/2014) and 4 fields of attribute information − station number, name, Latitude co-ordinate and Longitude co-ordinate in the World Grid 1984 co-ordinate system. Below, is a snippet image of the data.

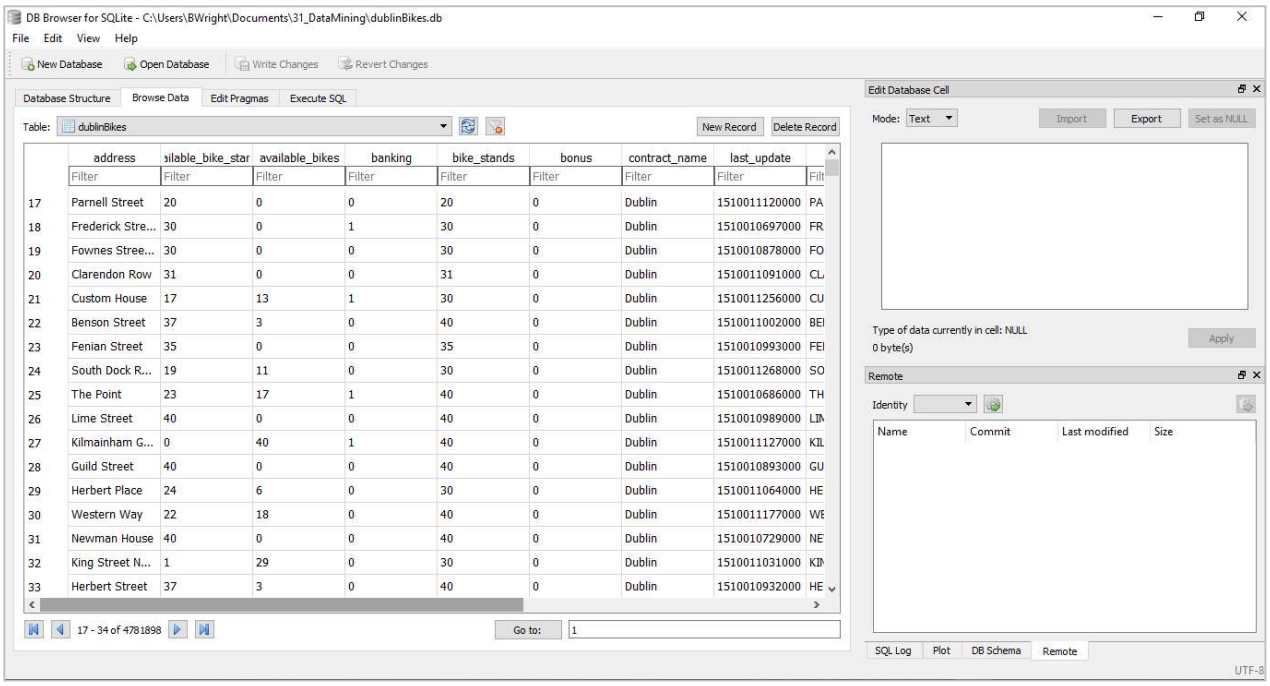
[12] https://developer.jcdecaux.com/#/login
[13] https://api.citybik.es/v2/
[14] https://github.com/jameslawlor/dublin-bikes-timeseries-analysis
[15] http://www.met.ie/climate-request/
[16] https://www.wunderground.com/weather/api

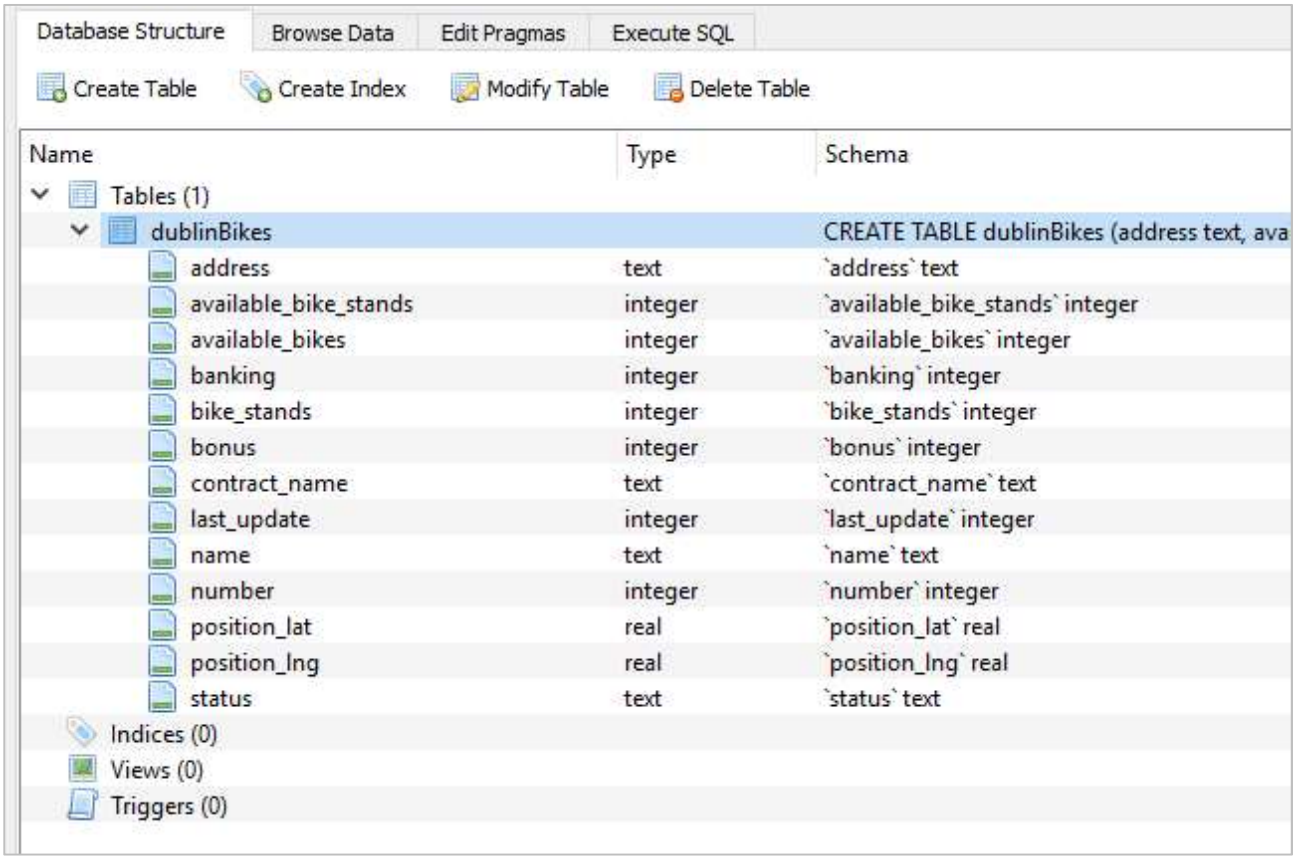
Coca-Cola Zero dublinbikes Station GPS Co-Ordinates V1.0				
as at 11/9/2014				
Stn. No.	Name	Latitude ('	Longitude (WSG84)	
1	CHATHAM	53.34096	-6.26229	
2	BLESSING'	53.35677	-6.26814	
3	BOLTON S	53.35118	-6.26986	
4	GREEK STF	53.34687	-6.27298	
5	CHARLEM(	53.33066	-6.26018	
6	CHRISTCH	53.34337	-6.27012	
7	HIGH STRE	53.34357	-6.27507	
8	CUSTOM H	53.34788	-6.24805	
9	EXCHEQUE	53.34303	-6.26358	
10	DAME STR	53.34401	-6.2668	
11	EARLSFOR	53.33402	-6.25837	
12	ECCLES ST	53.35925	-6.26978	
13	FITZWILLI/	53.33607	-6.25283	

Dynamic Dublinbikes data

The data was scraped (from the JC Decaux API) into a SQLite database and a CSV file. The SQLite database was 427 MBs and the data consisted of 4,781,898 records representing each time interval over the 8-day time period and 13 fields of attribute data – address, available bike stands, available bikes, banking, bike stands, bonus, contract name, last update, name, position_lat, position_long and status.







The CSV file (that was scraped from the CityBikes API) was 56.3 MBs and contains 623,679 records. Less fields were collected into the CSV file as I was unsure of what the size limitations for a CSV file were. If I was to repeat this process, I would include headings in the CSV file and I would do a calculation on how many records I would expect to get. For example, If I scrape at 2-minute intervals during an 8-day period from 102 stations (30 records for each hour * 24hrs * 8 days * 102 stations). This helps to identify if there were any issues during the scraping process – for example if a station was temporarily closed or if the scraping process was interrupted due to issue with Wi-fi. To identify if stations are closed, this data can be collected during the scraping process. If this data was not collected during the scaping process, it is possible to get information from the website (see image below). However, this is not ideal, when you are dealing with a vast amount of data. Also, I accidentally turned off my wi-fi on the Sunday and I only noticed when my phone could not connect to the wi-fi. If I was to repeat this process in the future – I would probably use Amazon Web Services instead of a personal computer. This was my first time scraping data and I learned a lot from the process.

The snippets below show the output file in windows explorer and Excel. The Windows explorer images show metadata on file size and when the file was created. The Excel image shows how the scraped data looked.

Name	Date modified	Type	Size
output	15/11/2017 19:11	Microsoft Excel Comma Separated Values File	47,514 KB
DBikes	01/11/2017 23:18	Python File	4 KB

Type of file:	Microsoft Excel Comma Separated Values File (.csv)
Opens with:	Excel 2016 Change...
Location:	C:\Users\BWright\Desktop
Size:	46.3 MB (48,654,251 bytes)
Size on disk:	46.4 MB (48,656,384 bytes)
Created:	06 November 2017, 23:38:56
Modified:	15 November 2017, 19:11:33
Accessed:	06 November 2017, 23:38:56

	A	B	C	D	E	F	G	H
1	0	0	35	0	0	FENIAN STREET	63	2017-11-06T23:37:04.533
2	1	0	30	1	1	CITY QUAY	99	2017-11-06T23:37:04.550
3	2	9	31	2	2	FITZWILLIAM SQUARE EAST	89	2017-11-06T23:37:04.625
4	3	30	0	3	3	BROOKFIELD ROAD	84	2017-11-06T23:37:04.519
5	4	40	0	4	4	EMMET ROAD	83	2017-11-06T23:37:04.633
6	5	35	0	5	5	ROTHE ABBEY	85	2017-11-06T23:37:04.619

GitHub Data – Bike and Weather Data

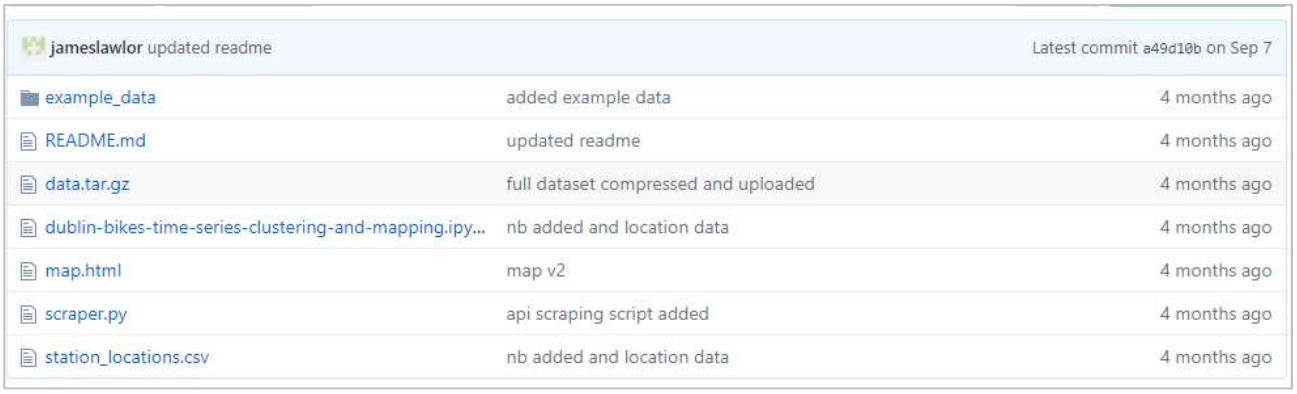
As mentioned in previous sections of this report, bike data was available on GitHub. The bike data consisted 0f 206 CSV files (1 for each day) ranging in size from 7kb to 195kb. The data was scraped at 2-minute intervals, so most files contained circa 710 records. Some days were missing records and there was a complete absence of data for some days in March. The data was prepared like a matrix with the time series down the left side and the station names across the top with the available bikes stored at the intersection.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1		FENIAN_S	CITY_QUA	FITZWILLI	BROOKFIE	EMMET_R	ROTHE_AI	KING_STR	GREEK_ST	WESTERN_	CHARLEM	PARKGATI	HIGH_STR	HEUSTON_	PORTOBEI	ECCLES_ST	MERRION_	SMITHFIEL	PORTOBEI	CUSTOM_	MERI
2	00:00:08	9	9	1	8	31	6	9	14	3	10	23	8	7	13	16	0	15	7	18	
3	00:02:09	9	9	1	8	31	6	9	14	3	10	23	8	7	13	16	0	15	7	18	
4	00:04:09	9	9	1	8	31	6	9	14	3	10	23	8	7	13	16	0	15	7	18	
5	00:06:09	9	9	1	8	31	6	9	14	3	10	23	8	7	13	16	0	15	7	18	
6	00:08:09	9	9	1	8	31	6	9	14	3	10	23	8	7	13	16	0	15	7	18	
7	00:10:09	9	9	1	8	31	6	9	14	3	10	23	8	7	13	16	0	15	7	18	

The weather data consisted 0f 206 CSV files (1 for each day) ranging in size from 1kb to 26kb. The data was scraped at 2-minute intervals, so most files contained circa 710 records. Some days were missing records and

there was a complete absence of data for some days in March. The weather data consisted of 5 fields, - the time the weather observation was scraped at, categorical weather data, numeric temperature and wind data.

	A	B	C	D	E	F
1	Time	Weather	Temperat	Feels_Like	Wind_Speed	
2	00:00:08	Mostly Cloudy	17	17	24	
3	00:02:09	Mostly Cloudy	17	17	24	
4	00:04:09	Mostly Cloudy	17	17	24	
5	00:06:09	Mostly Cloudy	17	17	24	
6	00:08:09	Mostly Cloudy	17	17	24	
7	00:10:08	Mostly Cloudy	17	17	24	
8	00:12:09	Mostly Cloudy	17	17	24	



The bike data in the GitHub repository satisfied the needs of my project. However, the weather data did not. I needed to supplement the weather data with the Met Éireann weather data. The Met Éireann weather data had precipitation as a numerical value. The GitHub data only had a categorical text value. After the initial data exploratory stage and issues with my models, I learned that the categorical value filed was not useful. There was little consistency − for example − light rain was described as drizzle, mist and so forth. I tried to re-map this data into a field with a reduced range of values, but I felt that I was introducing human error into my data and models, so I did not include this field in my final model.

3.2.3 Explore data

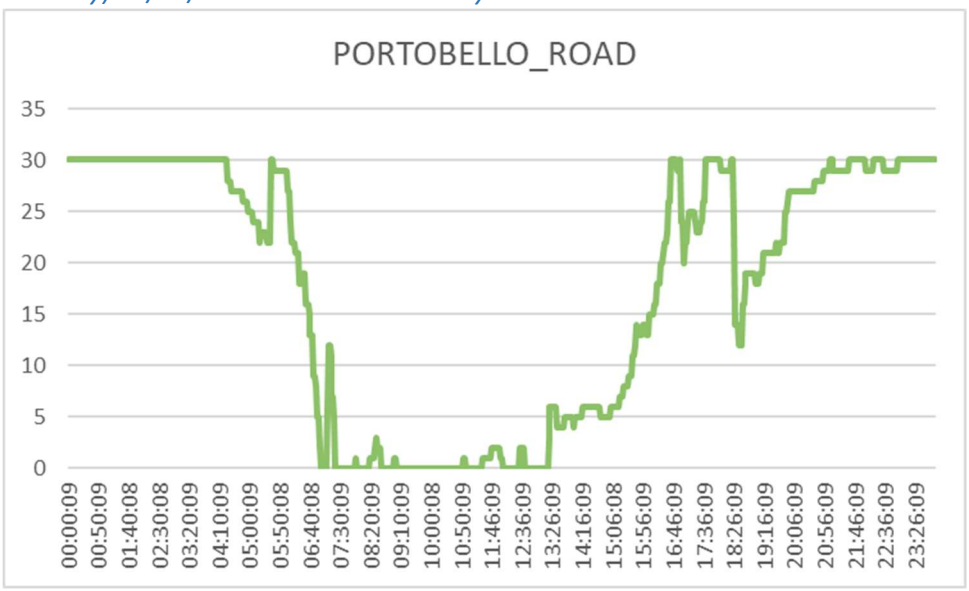
The data was explored in Excel and RapidMiner. Charts were created on snippets of data and test models were developed before loading larger amounts of data into the models. This also helped to evaluate the best models to answer my business question.

At this stage it was identified that the weather data and the bike data would have to be combined. First, they were combined separately using the Windows command prompt (see Appendix B), then they were joined together. It was also necessary to create extra fields of data such as season, working day, holiday and day of week. Some of these fields such as season and month were not used in the end because of the issue loading the large dataset into Excel and RapidMiner. I ended up only using 1 month of data for most of my business questions, so season and month were redundant values and did not add to my prediction values.

The images below show differences in use patterns on weekdays and weekends.

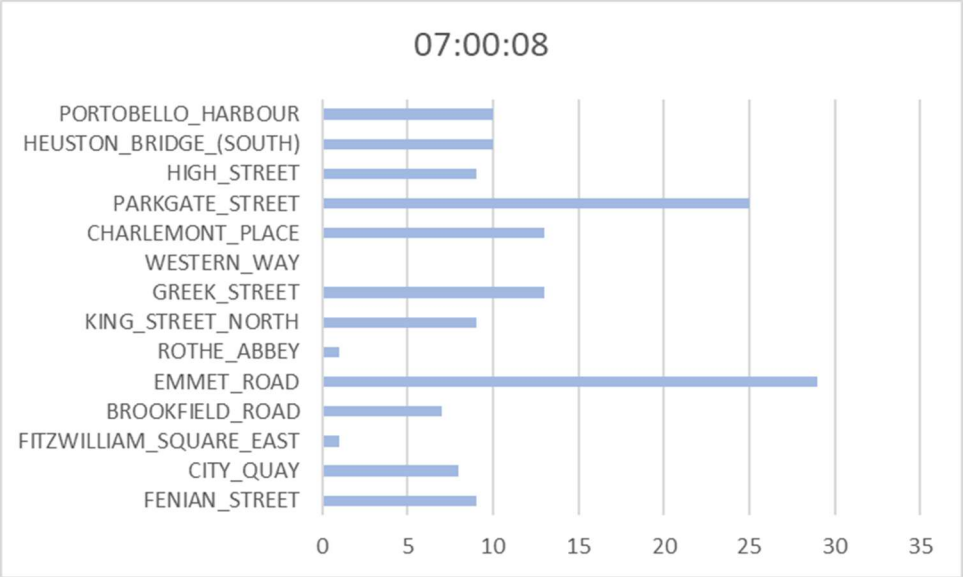
Sunday (weekend of bank holiday) 30/04/2017 – Trend across day

Wednesday (weekday) 12/04/2017 – Trend across day

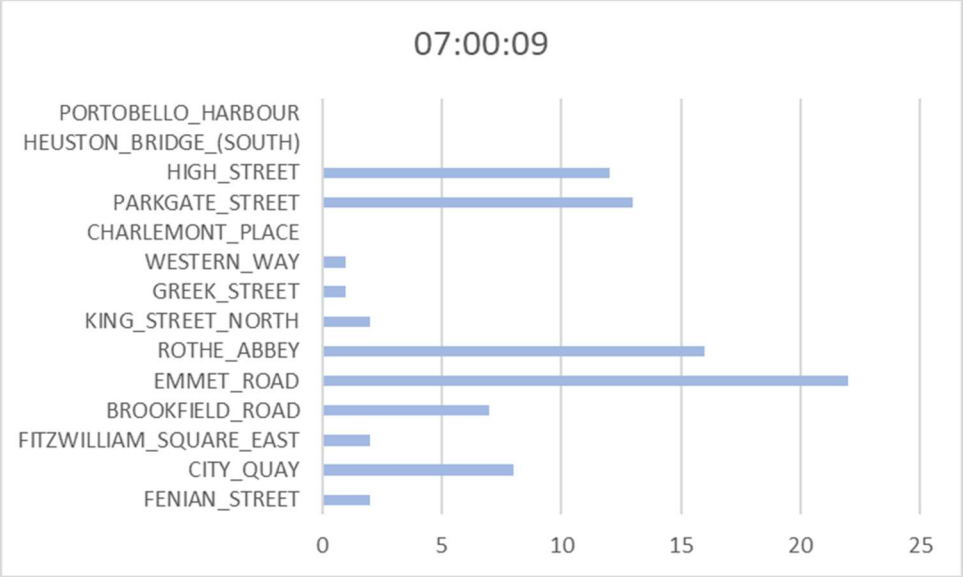


The images below show different patterns at different times of day on a weekday and a weekend day.

Sunday (weekend of bank holiday) 30/04/2017 – Trend at 7.00am in the morning across 14 stations (data not normalized – some stations have capacity for more bikes)

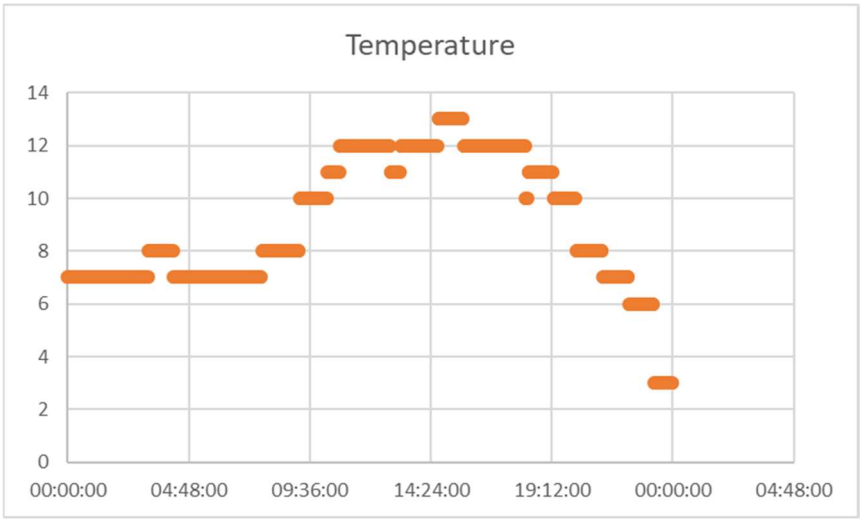


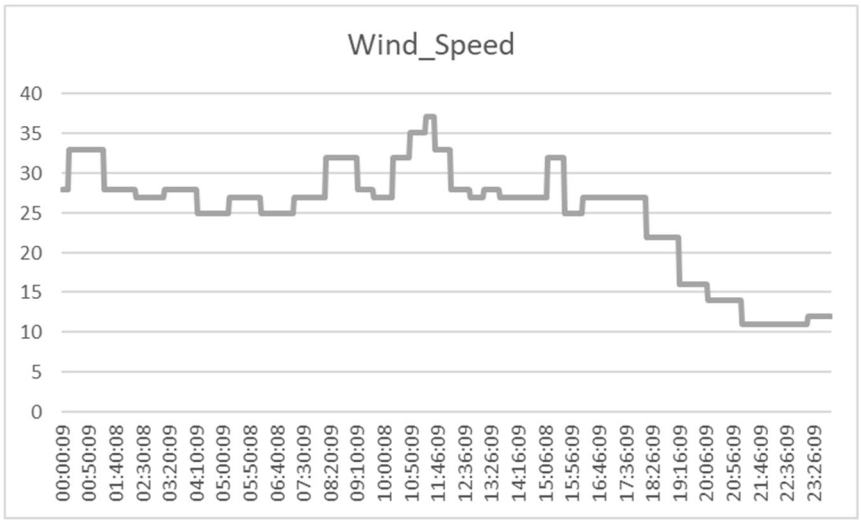
Wednesday (weekday) 12/04/2017 – Trend at 7.00am in the morning across 14 stations (data not normalized – some stations have capacity for more bikes)



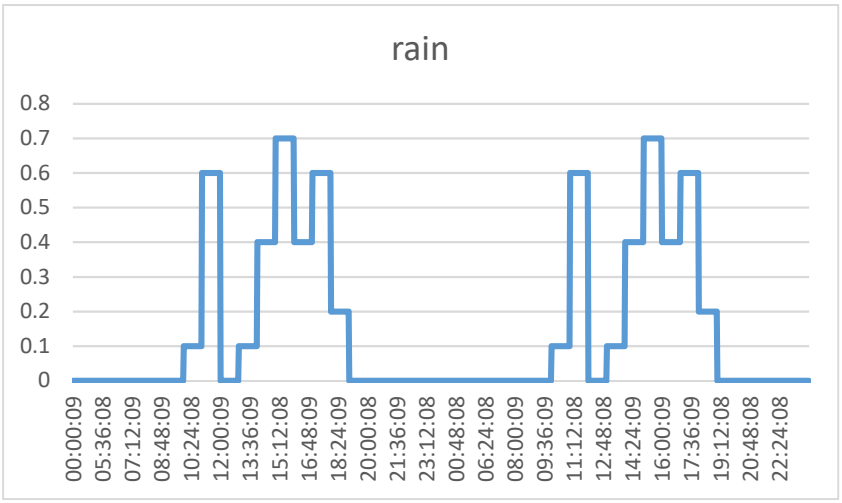
The images below show the variability in weather across the hours of a day.

Wednesday (weekday) 12/04/2017 – Temperature and windspeed across the day

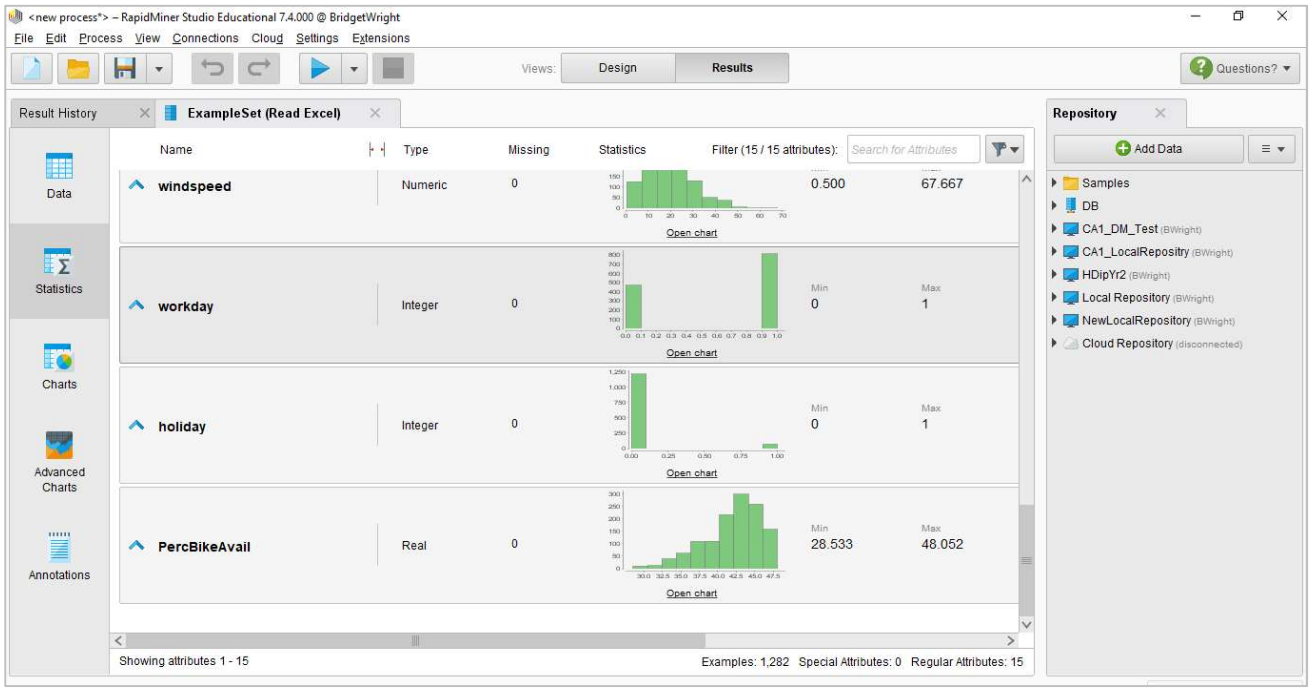




Thursday (weekday) 02/02/2017 – Rainfall across the day



The image below shows some of the exploratory data analysis in RapidMiner.



I aggregated available bikes at all the stations and averaged the 2-minute interval values into an hour value. This was not good input for my cluster model. It eliminated the variability of the data. Aggregation and averaging was more appropriate for my prediction model (I will discuss in detail later).

This initial exploratory data analysis fed into my data mining goals. It helped to refine the data description, and fed into the transformation and other data preparation steps needed for further analysis.

3.2.4 Verify data quality

As mentioned above, the scraped data and the data from GitHub was missing records. There are no missing values just missing records which is good. Therefore, it will not be necessary to replace null records. The issue with the missing records can be overcome by amending the timeframe of my analysis. Trying to analyse 6 months of data would be a mammoth task for this assignment (due to my hardware, software, time, manpower resource and expertise constraints) so the missing records helped me come to a valuable conclusion early on in my project. I believe that the data I have is fit for purpose and meaningful insight can be drawn from it. It was necessary to reduce the data time range for some of the business objectives. The scope of my business questions and data mining goals might be further modified when I get to the latter stages of the project.

3.3 STAGE THREE – DATA PREPARATION

This stage involved reviewing my data, deciding which data to use, collecting or creating additional data, considering the use of sampling techniques, documenting why I used certain data and did not use other data, cleaning the data, aggregating the data and identifying if there are outliers.

3.3.1 Select Data

As noted earlier, I collected a lot of data from different sources, - the JC Decaux API, the CityBikes API, Met Éireann and GitHub. After my initial exploratory data analysis and examining the length of time it took some models to run, I decided that I would use subsets of the data. The purpose of this assignment was to explore open data, learn data mining techniques and apply the CRISP-DM process. I did not want to get overwhelmed with vast amounts of data and not be able to produce meaningful insight. Perhaps, at a later stage (in a different project), the Dublinbikes dataset could be revisited and a wider range of data could be assessed. I decided that I would use different amounts of data to answer the different business questions

1) *To identify if there is a different pattern of use on different days (weekdays V weekends).* I will use 1 week of data for this business question

2) *To identify if there is a different pattern of use at different times throughout the day.* I will use 1 – 2 weeks of data for this business question

3) *To identify if there are similar characteristics amongst the stations (i.e. busy and quiet stations)* I will use 6 months of data for this business question

4) *To identify if bike use patterns can be accurately predicted.* I will use 1 month of data for this business question.

3.3.2 Clean Data

The datasets are relatively clean, there are no missing values just missing records. The data was captured at 2-minute intervals which is far more frequent than what is required for this project. As the GitHub data was collected by a data scientist for another project that went rough many iterations, the data that is available on GitHub is very clean.

3.3.2 Construct Data

The weather and bike data were captured in individual CSV files for each day. These files had to be combined for analysis and to incorporated into my models. I was able to combine the files using simple code in the windows command line prompt.

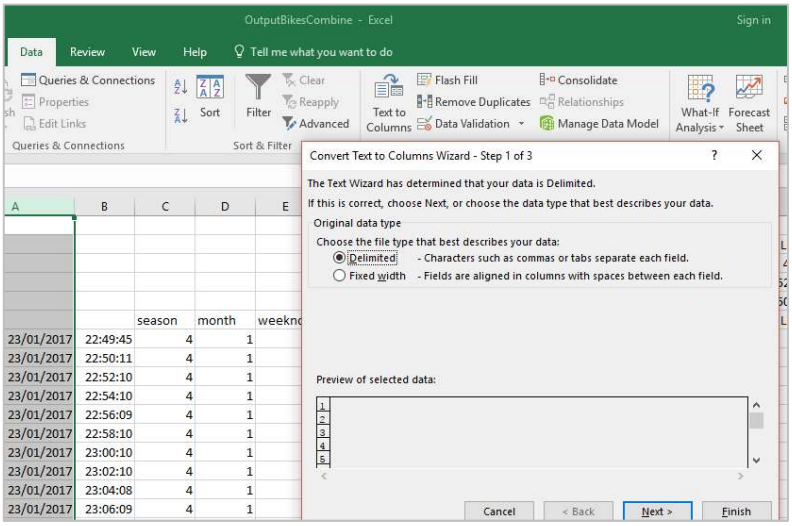
```
copy* .csv WeatherCombine.csv      (copy*.csv = code for copy all csv files
WeatherCombine.csv = the name to be allocated to the new csv file )
```

After the files were combined, an issue was identified. Each of the csv files did not contain the date vale in each of the records, it was only contained in the file name and file tab.

I needed code that would copy all the csv files and add the name of the file that each record came from. I used the following code

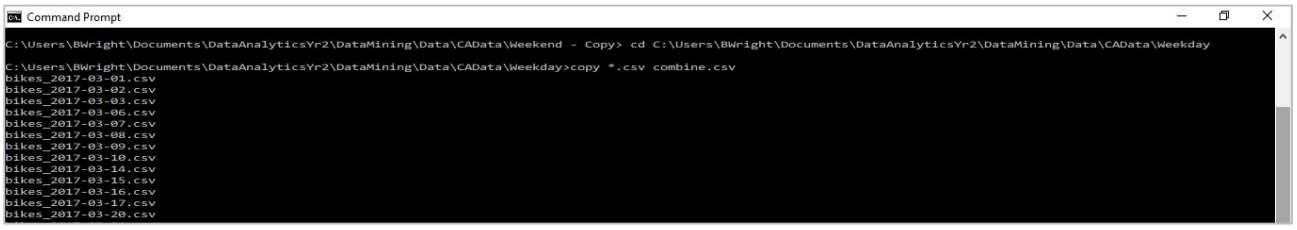
```
findstr "^" *.csv>OutputWeatherCombine.csv
```

The text to column tool in Excel was used to extract the data value from the file name / file path that was extracted using the above code. Underscores and other unnecessary characters were removed.



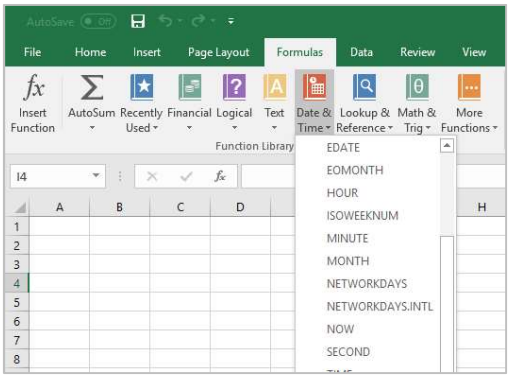
My computer struggled to handle the files with 6 months of data efficiently. There was insufficient storage space to process formulas and calculations and therefore this was a contributing factor behind my decision to scale back the data.

```
C:\Users\BWright\Documents\DataAnalyticsYr2\DataMining\Data\CAData\Weekday>copy *.csv
combine.csv
bikes_2017-03-01.csv
bikes_2017-03-02.csv
bikes_2017-03-03.csv
bikes_2017-03-06.csv
bikes_2017-03-07.csv
bikes_2017-03-08.csv
bikes_2017-03-09.csv
bikes_2017-03-10.csv
bikes_2017-03-14.csv
bikes_2017-03-15.csv
bikes_2017-03-16.csv
```



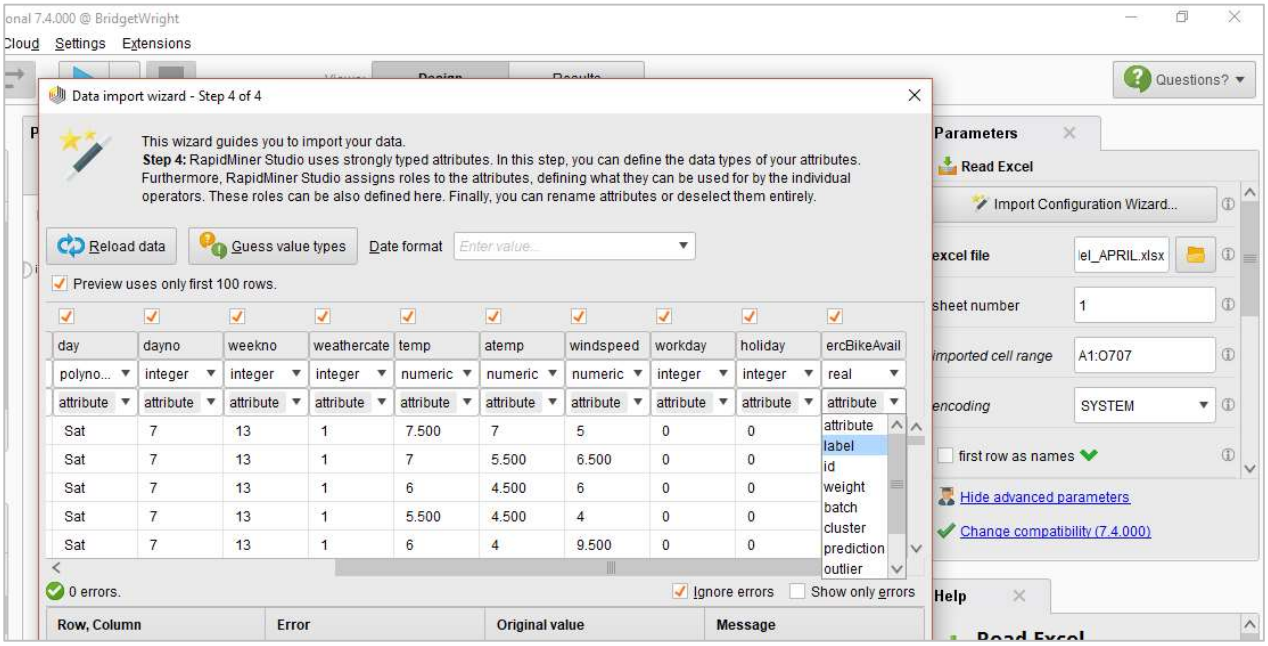
3.3.3 Integrate Data

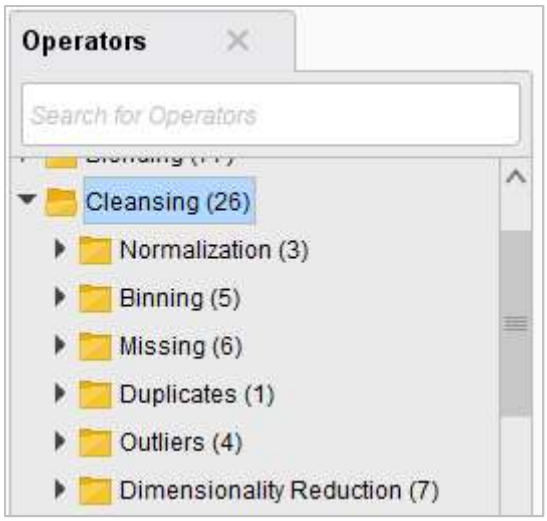
When I selected the data that I needed and was satisfied with the data for the different business questions, I then joined the weather and bike data together. After I joined the weather and bike data – I created some new fields of data – season, day of week, workday, holiday, week of year and month. I was able to create these new fields of data from the existing data using data and time formulas in Excel.



3.3.5 Format Data

The data was formatted on a case by case basis when it was incorporated into each of the models in RapidMiner. This was either completed through the Import Configuration Wizard or through operators when I thought I wanted the formatting replicated in numerous models. Sometimes, I wasn't always aware that my data had to be reformatted but the 'Wisdom of the Crowds' recommended operators and warning notifications in RapidMiner helped me to identify when it was necessary to format data. Examples include changing polynominal data to text because certain operators couldn't be implemented with certain data types and applying the label to the special attribute field. It also helped me to identify when certain fields were not necessary. Just because I have the date field, it is not always essential and by removing these fields through the import configuration tool or the cleaning operators, it helped my models to run more efficiently.





3.4 STAGE FOUR – MODELLING (AND STAGE FIVE – EVALUATION)

Various modelling techniques were considered during this stage. Parameters were tweaked and calibrated. I was aware that my business problem could be resolved using a variety of modelling techniques. While, I carried out some research on techniques at the data exploration stage, new ideas, new research material and discussions with colleagues changed my direction. Also, trial and error dictated some changes at this stage. It was also necessary to step back to the data preparation stage when I discovered that a technique I was hoping to apply wouldn't work. For example, there are 102 bike stations, some stations were missing values for extended periods of time because they were closed due to external events such as construction works or the decision to move a station. As there was a lot of stations, instead of replacing these values or averaging values, I decided to eliminate these stations from the scope of my study. While examining Kaggle data on a bike share scheme in Washington DC, I observed that there was only 1 field for the bike count. I decided that I should aggregate the data across all the stations for each time value. After spending considerable time reformatting my data and then popping it into my model, I realised that there was little variability amongst my data. This was a rookie error and one that I hope to never repeat! I did not use the insight I had gained from my clustering model where I identified that there was different types of stations and that certain time of day some stations have few bikes and other stations have lots of bikes. By aggregating this data, it seemed like there was little variability across the day. I learned the very important that it is important to use all insight gained and if new information is revealed that I should re-evaluate my business questions and data mining goals. Iteration and re-evaluation is very important in the CRISP-DM process. I am not sure why the Washington DC data set only use 1 field of data but perhaps the redistribution of their bikes is managed differently, or all their stations have the same use patterns. Nonetheless, aggregating the data this way is not suitable for my business questions.

Different modelling techniques were required for the different business objectives:

- To identify if there is a different pattern of use on different days (weekdays V weekends)
- To identify if there is a different pattern of use at different times throughout the day
- To identify if there are similar characteristics amongst the stations (i.e. busy and quiet stations)

- To identify if bike use patterns can be accurately predicted using weather values

All assumptions associated with each modelling technique were recorded.

3.4.1 Clustering model – to identify similarity amongst stations

Select modelling technique, generate test design, build model, assess model

I chose the clustering model because I wanted to find patterns / groups amongst the stations. Clustering models are for explanatory purposes and are unsupervised models. It was decided to use a clustering for business objective 4 – to identify if there are similar characteristics amongst the stations. While this is business objective 4, the data mining for this business objective was carried out first after it was identified during the exploratory data analysis stage that different stations have different pattern uses at different times of day.
I trialled a few algorithms (K-means and k-medoids) but some of the results were meaningless and difficult to interpret and therefore were not considered useful for this business problem. K-means clustering proved to be the most useful technique with meaningful results.
Clustering is an unsupervised process for finding meaningful groups in data. My data mining goal for this problem is to segregate the stations based on similar patterns of use. The data mining goals for this business problem aim to find the groupings in the data in such a way that data points within a cluster are more "similar" to each other than to data points in other clusters.

The objective of K-means clustering is to find a prototype data point (centroid) for each cluster, all the data points are then assigned to the nearest prototype which then forms a cluster.

Steps followed to build the model:

Step 1 – Data Preparation:
I checked the data types in my dataset. I also checked the requirements of the algorithm in RapidMiner. The algorithm accepts both numeric and polynomial data types. My data preparation steps at earlier stages in the assignment already limited the number of attributes to just the number of bikes available at a station. However, there is a lot of stations (102 in total but I only used 98 in this model because of data quality issues). The number of bikes available had to be normalized by converting the number to a percentage as different numbers of bikes are available at different stations.
I had to revisit the data preparation to build this model. I wanted to use 6 months of data. However, Excel and RapidMiner could not handle the size of my data file (circa 710 records for each day, circa 180 days for 98 stations). I decided to average the 2-minute interval data to create hour interval data. This proved efficient for processing times. However, due to the nature of the real-world features that this project is examining and how people interact with the bikes (the average journey is circa 15 minutes[17]) – I wanted to check if there was a different cluster when using the 2-minute interval data over a 2-day period in February. I will detail the model build below and discuss the verification process and my comments on the 2-day cluster model versus the - month cluster model.

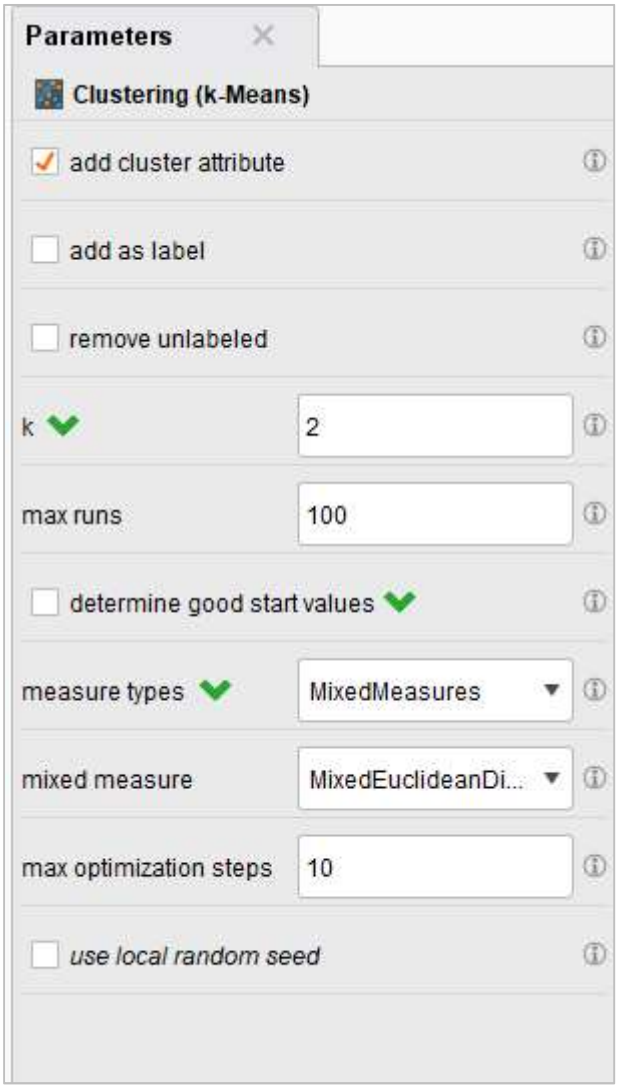
Step 2 – Modelling and Parameters – building the model:
I selected the number of clusters. The wisdom of the crowds suggested between 1 and 4 clusters. I tried a few different values but settled with 2 for the 6-month model because it had the best values under the performance vector results (I will discuss later). For the 2-day model, I applied 2k and 3k values. The results were interesting. While the 2k had a lower Davies Bouldin value, the 3k had a lower average with centroid

[17] http://www.dublinbikes.ie/

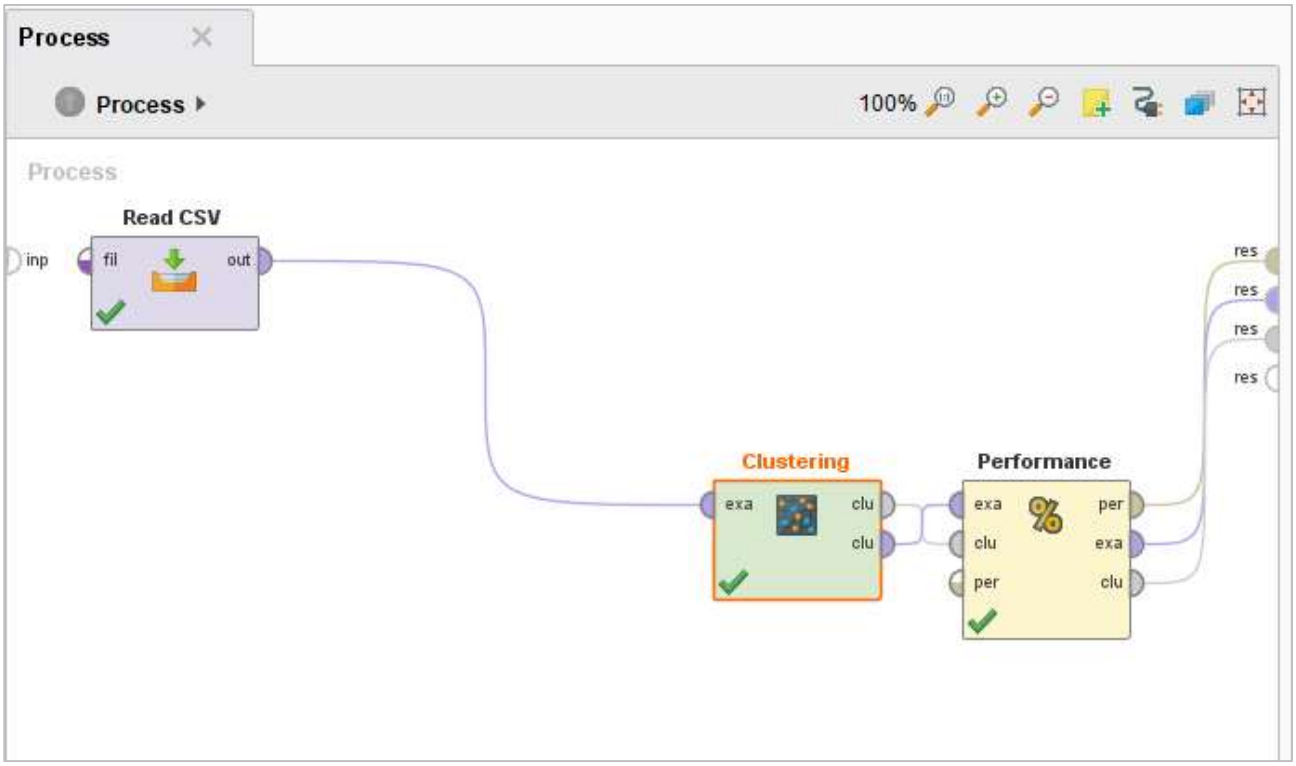
distance. It was necessary to examine the visual output to further assess the best option. The graph revealed that the 3k was a better model.

I filled out the parameters in RapidMiner as follows:

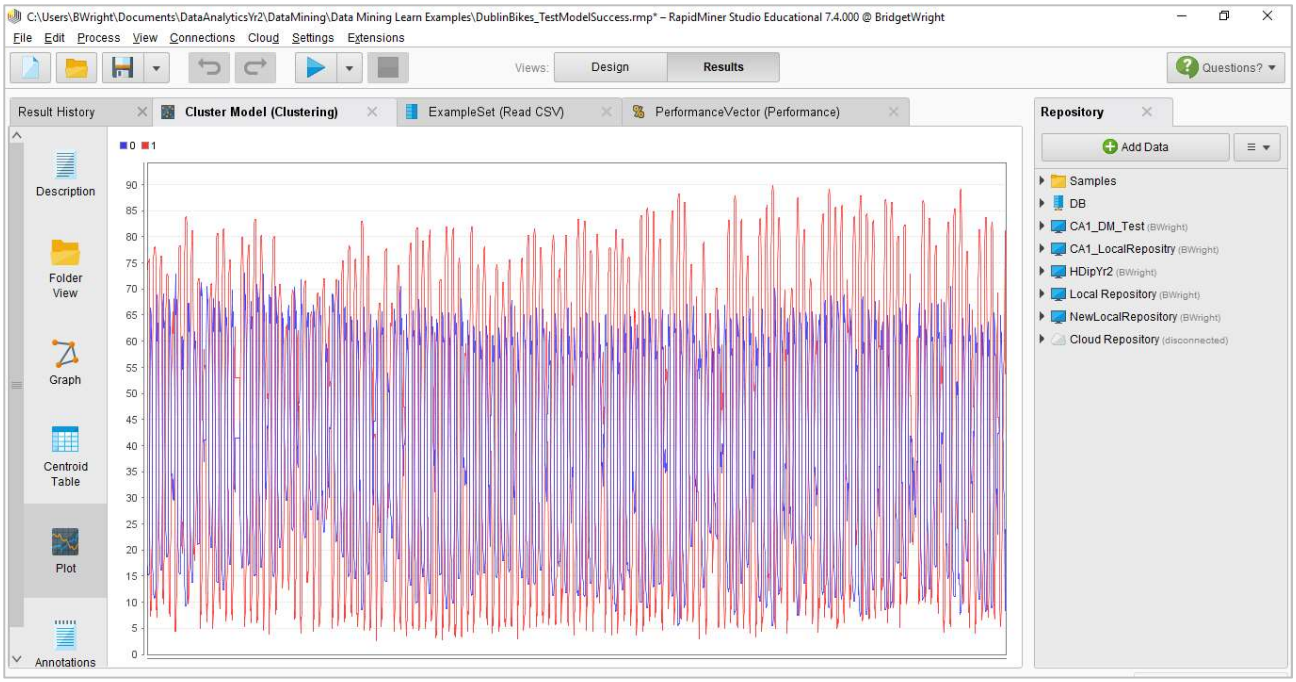
- Add cluster as attribute (this is recommended)
- I selected 10 as the number of max runs. This was necessary to permit multiple runs to select cluster with lowest SSE
- The Wisdom of the Crows recommended the Bregman Divergence measure type. This parameter serves as a proximity measure.
- I selected 100 for the max optimisation steps parameter as this was the default value. The purpose of this parameter is to assign data points to centroid and calculate new centroids.



Please see below for an image of the cluster model. As I did much of my data cleaning in Excel, I only needed 3 operators for this model – read csv, clustering and performance vector.



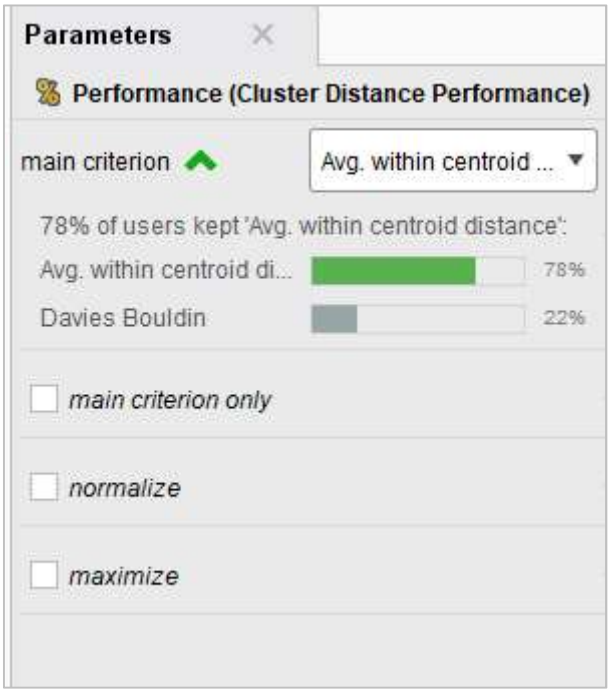
Below is an image of the cluster model plot for 6 months of data. The performance vector verified that 2k was the best value to apply. The Davies Bouldin index was lowest with 2k (-1.030) and the average within centroid distance revealed more favourable results with the 2k.

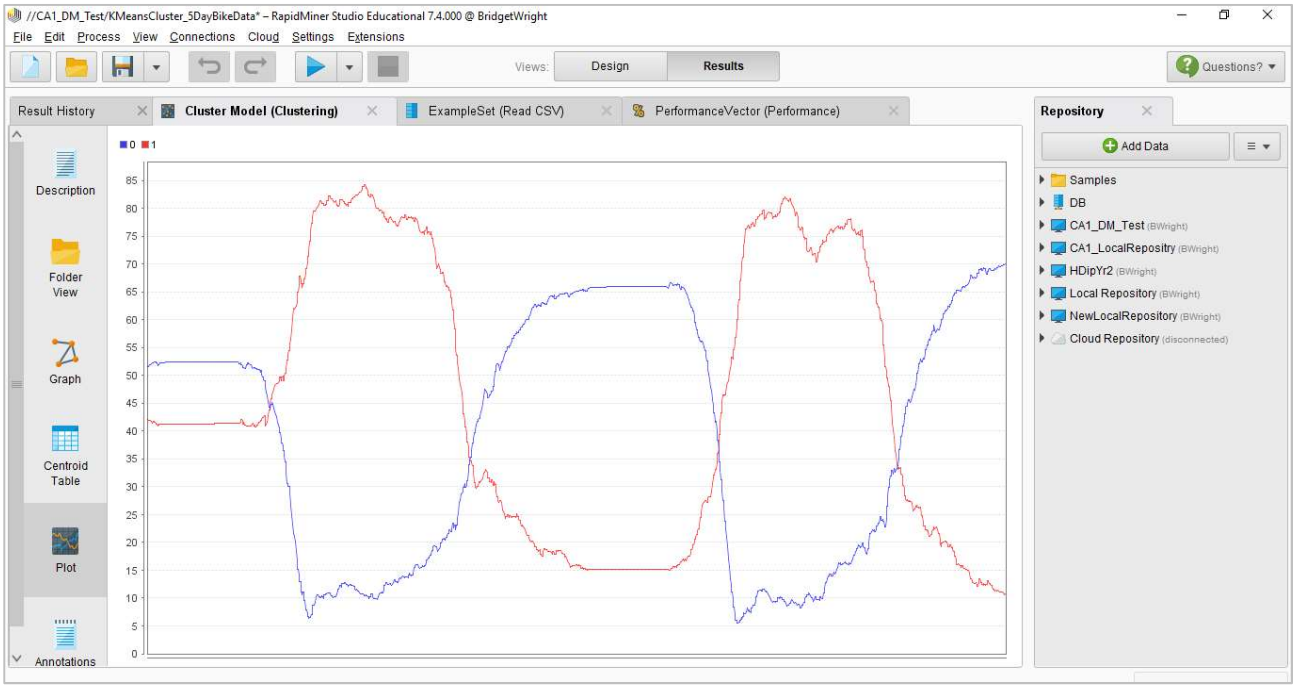


PerformanceVector

```
PerformanceVector:
Avg. within centroid distance: -933778.775
Avg. within centroid distance_cluster_0: -975464.884
Avg. within centroid distance_cluster_1: -880459.333
Davies Bouldin: -1.030
```

Step 3 – Implement Evaluation Method Cluster Distance Performance with two measurement outputs SSE and Davies-Bouldin

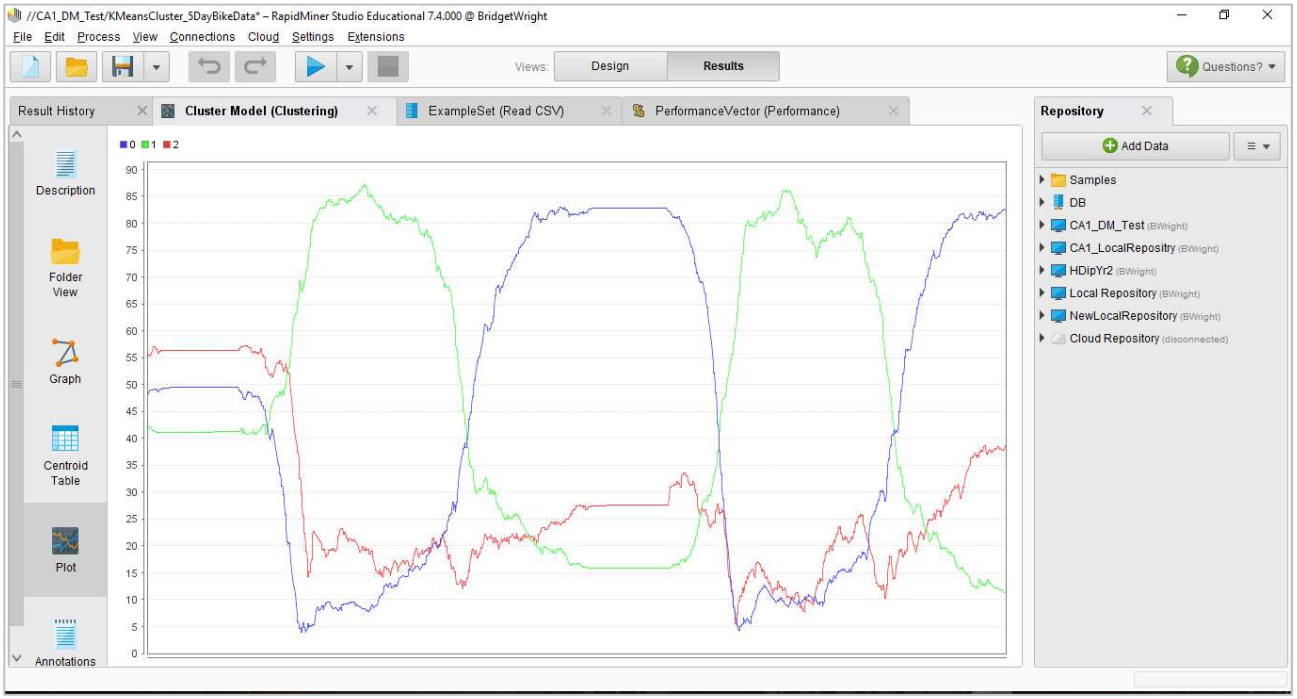




PerformanceVector

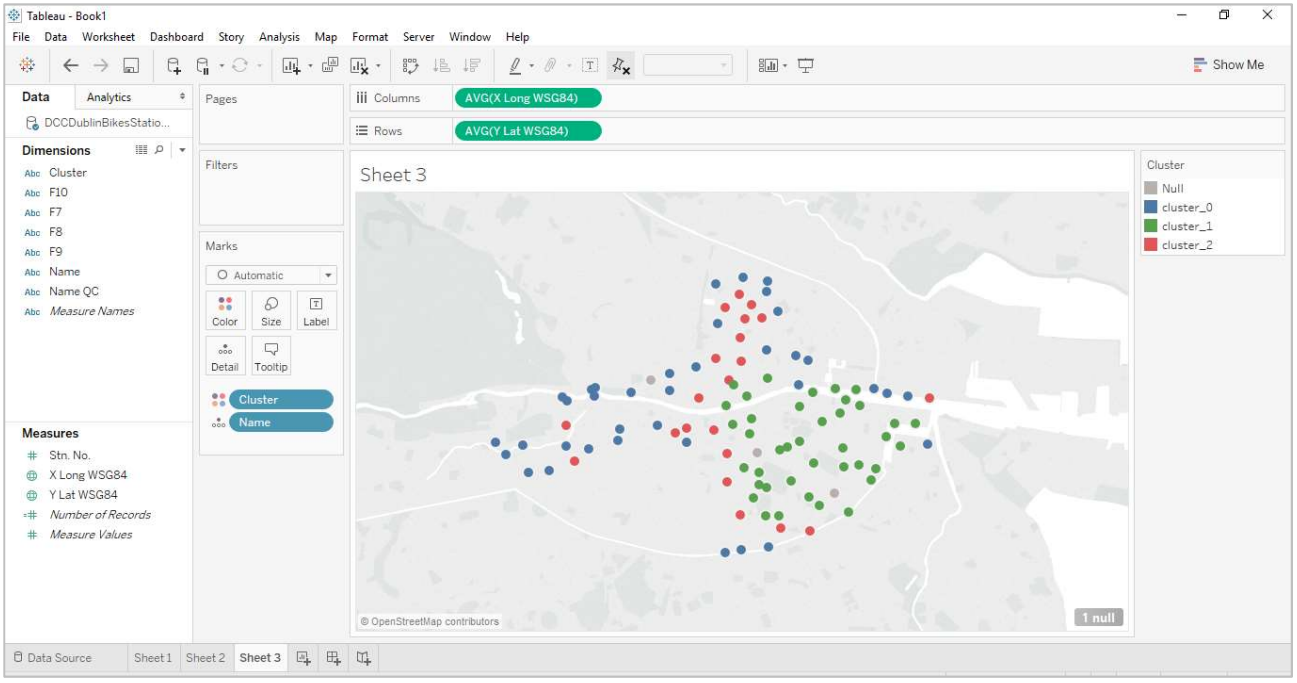
```
PerformanceVector:
Avg. within centroid distance: -933778.775
Avg. within centroid distance_cluster_0: -975464.884
Avg. within centroid distance_cluster_1: -880459.333
Davies Bouldin: -1.030
```

The time is not appearing in the image below. For visualization purposes, please note that the left-hand side of the graph is midnight and the right-hand side is 11.55pm. Stations are closed at 00.30am and open at 5.00am. all 3 clusters have no activity from 00.30am to 5.00am and the availability of bikes dips for the blue cluster at morning peak-time commuting.



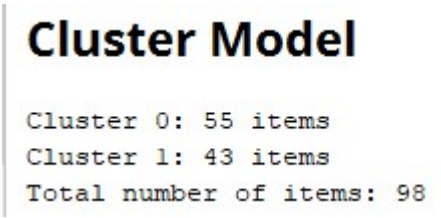
I deployed the cluster output from the RapidMiner Model into Tableau to visualise the real-world location of the clusters. It's straightforward to see that stations most closely fitting the blue line would typically be used for commuting into the city to the green stations in the morning, so naturally we'd expect that blue stations would be found in residential areas and green stations in the city centre (near office buildings). In the evening the opposite occurs as people head back home from work. Red stations have a steady supply over the day and don't fit in with either of the other two behaviours.

When we plot the stations on a map and colour code them by category an interesting picture emerges. By characterising the stations by only their usage patterns we can see this also results in a high degree of spatial clustering. While this is what I was expecting to see I was surprised by just how clear the clusters really were — the green stations are focussed around the Dublin 2 area where a lot of business offices are based, bordered by the in-between reds, and blue stations outside the city centre in more residential areas. I thought that some of the red stations near the North circular road might have come in as blue because there is a lot of residential land use here. However, it should be noted that this is near the Mater Hospital and this might be a reason why the stations around this area have a steadier flow of traffic throughout the day.



PerformanceVector

```
PerformanceVector:
Avg. within centroid distance: -749317.021
Avg. within centroid distance_cluster_0: -684551.886
Avg. within centroid distance_cluster_1: -844978.206
Avg. within centroid distance_cluster_2: -681215.238
Davies Bouldin: -1.229
```



As noted above, the performance vector values and the visual output revealed that the 3k was a better model. I will deploy the results on a spatial map in Tableau.

3.4.2 Linear Regression Models and Neural network models

Select modelling technique, generate test design, build model, assess model

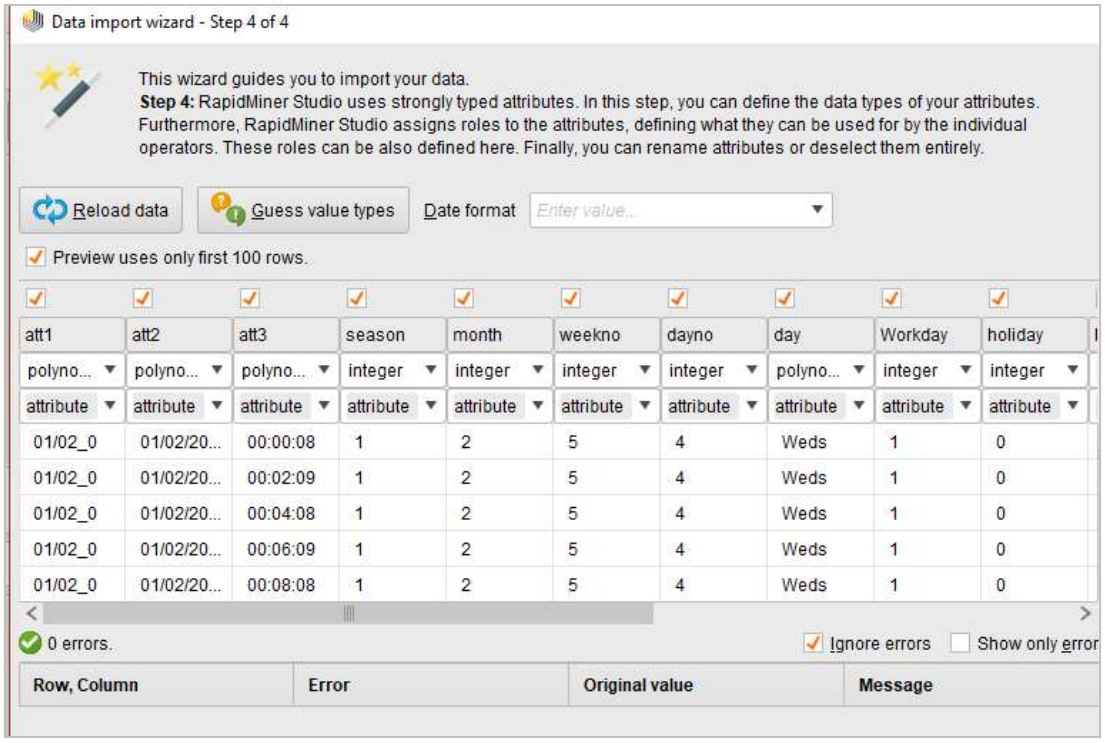
I selected the linear regression model because I wanted to predict a numeric target variable. Regression models are predictive, supervised models. I selected the Neural Network model because I wanted to evaluate it against the linear regression model performance.

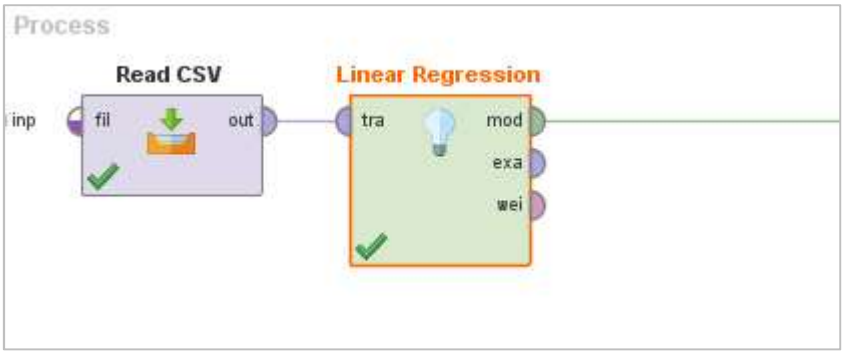
The number of bikes rented is the dependent variable which must be predicted by independent variables. Temperature, windspeed, rainfall, workday and holiday are independent variables. The weather data is absolute values. The workday value is binary. When a label is applied to a field in RapidMiner, it tells RapidMiner that it is the dependent variable

The input for the linear regression is a training dataset and the output is a model. When I ran the process, the output was a model. In some of my models, I applied the performance vector to evaluate my model.

There are a variety of performance parameters. In RapidMiner, the root mean square error parameter is the default. However, r-squared is what people are used to seeing when examining regression, so I chose this metric to evaluate the performance of my model. The R-squared parameter is the square correlation between the original dependant variable and the predicted dependent variable.

I entered the training data and then examined the output model. A small p value and a t-stat would indicate that the independent variables are significant. If the t-stats are above 2 and below negative 2, they are significant. However, this information alone, is not enough – I needed to apply the model and examine performance.



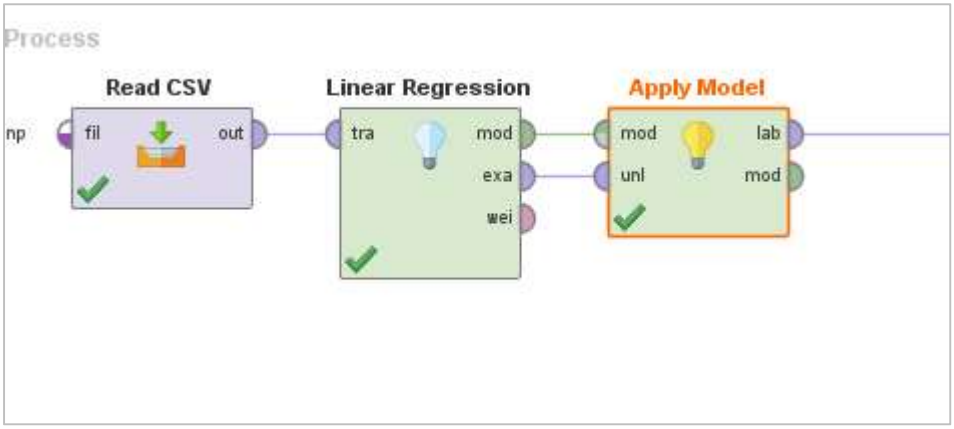


The p-values and the t-stats show the significance / insignificance of the variables.

Attribute	Coefficient	Std. Error	Std. Coefficient	Tolerance	t-Stat	p-Value	Code
Workday	85.401	1.083	0.389	0.993	78.862	0	****
hour	-1.009	0.081	-0.063	1.000	-12.498	0	****
temp	4.232	0.159	0.146	0.999	26.680	0	****
rain	-24.378	1.569	-0.078	0.984	-15.533	0	****
windspeed	-0.807	0.048	-0.093	0.993	-16.990	0	****
(Intercept)	100.500	1.796	?	?	55.971	0	****

LinearRegression

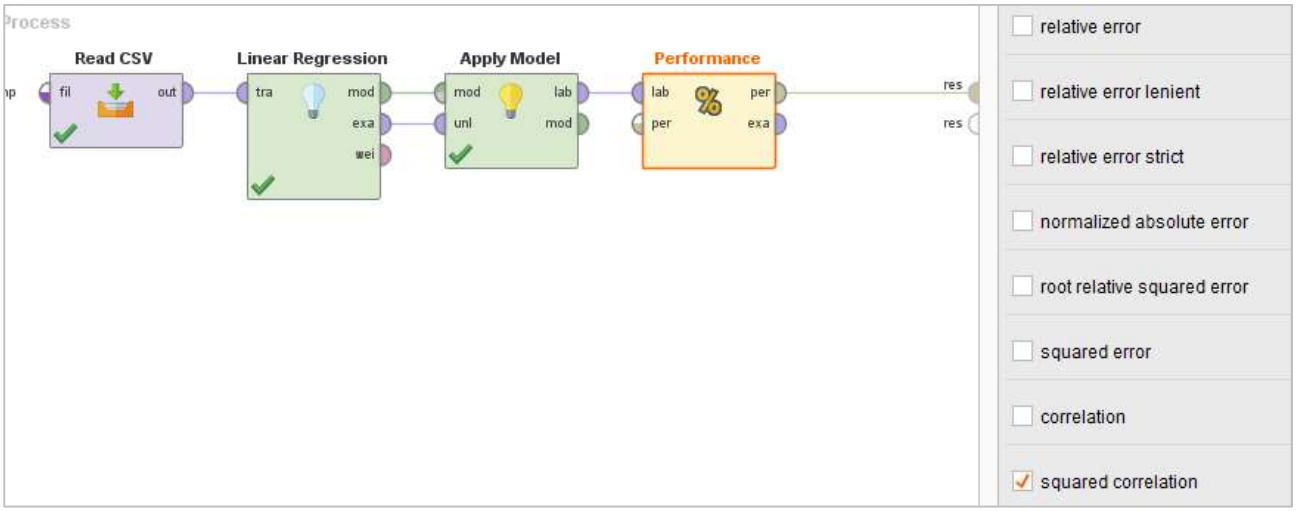
```
  85.401 * Workday
- 1.009 * hour
+ 4.232 * temp
- 24.378 * rain
- 0.807 * windspeed
+ 100.500
```

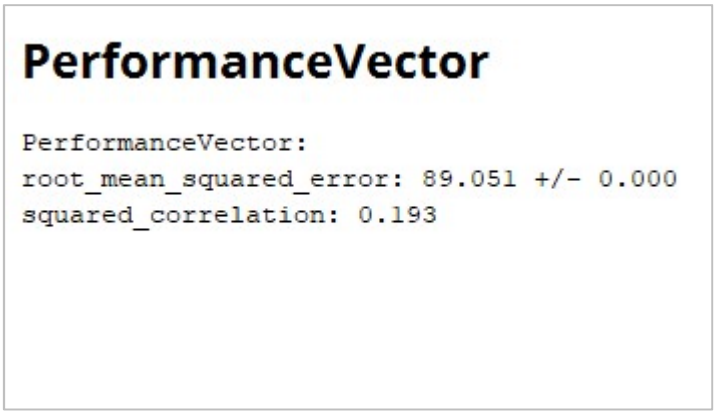


The apply model operator requires the original data and the model as input to examine the original data and the predicted values. The predicted field below shows how the model predicted the bike usage from the linear model.

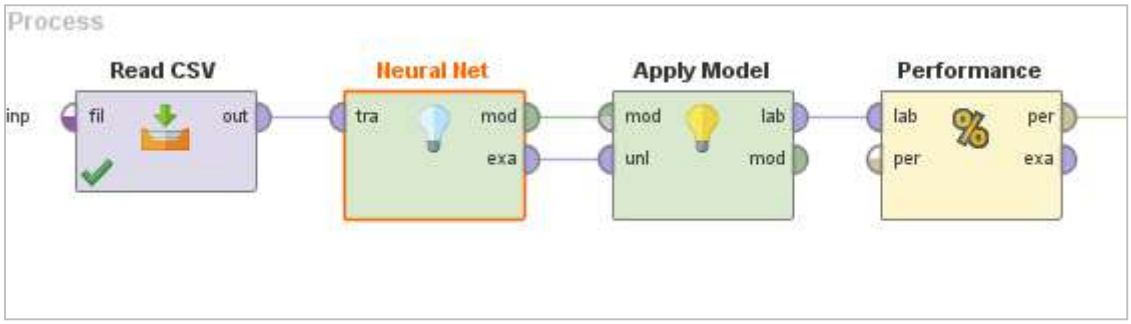
ExampleSet (33520 examples, 2 special attributes, 7 regular attributes)

Row No.	TotalinUse	prediction(T...	Workday	holiday	hour
1	118	194.578	1	0	0
2	118	194.578	1	0	0
3	119	194.578	1	0	0
4	124	194.578	1	0	0
5	124	194.578	1	0	0
6	117	194.578	1	0	0
7	117	194.578	1	0	0
8	114	194.578	1	0	0
9	115	194.578	1	0	0
10	115	194.578	1	0	0
11	111	194.578	1	0	0
12	110	194.578	1	0	0
13	110	194.578	1	0	0





Neural Network Model

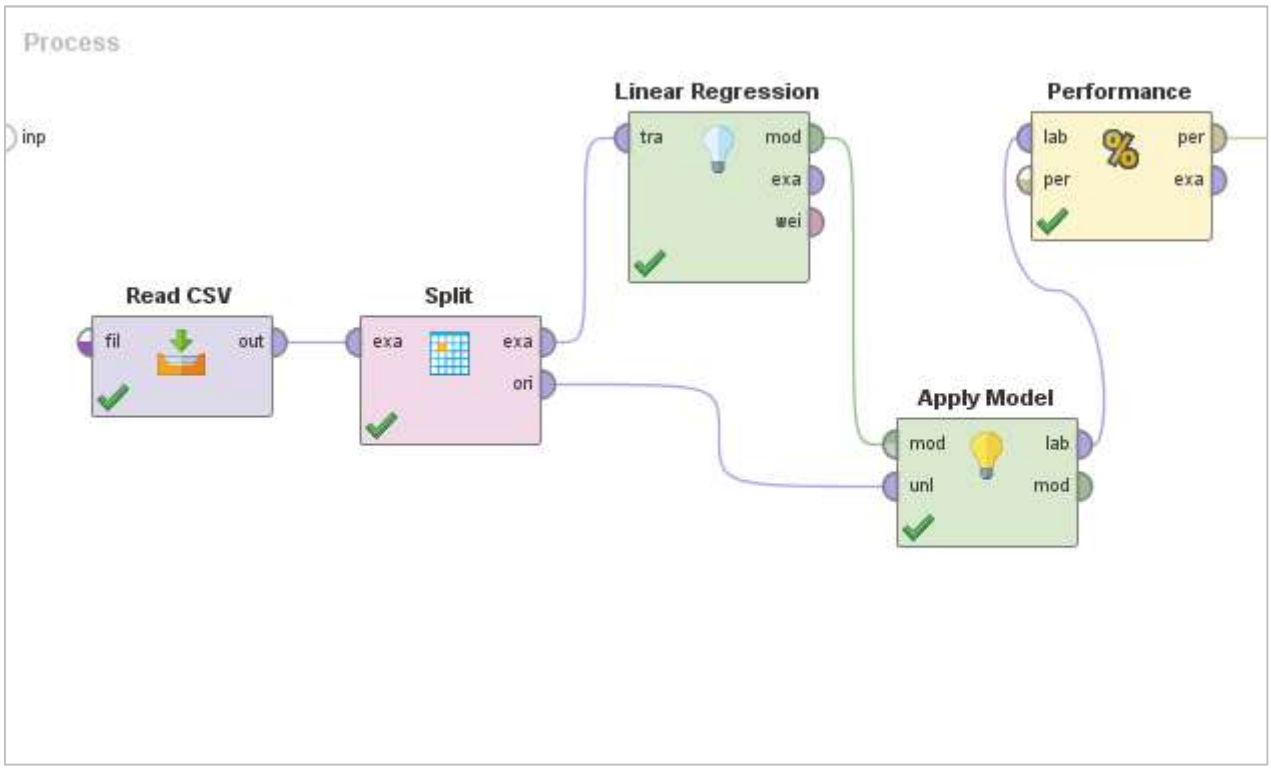


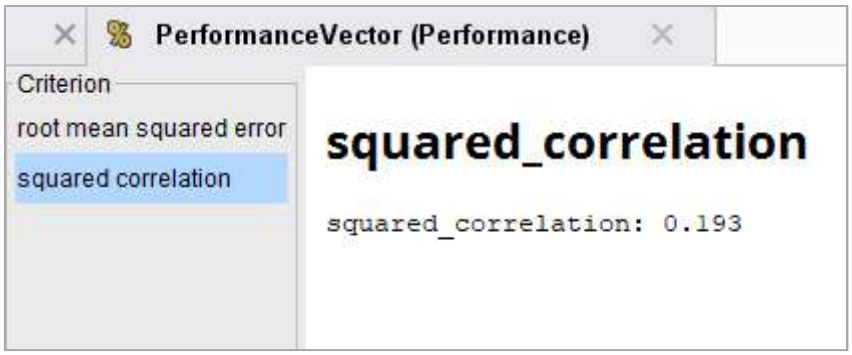
The modular approach in RapidMiner is efficient for workflow processes. I was quickly able to change my model by swapping out the linear regression model with neural network operator.

I got a squared error of 73% with the neural network operator which was better than the linear regression model. A squared error of 95% would be exceptional. I tried modifying the linear regression model by removing some of the independent variables.

Split the data

it was important to consider how I applied the model. In my previous model I applied the model using the same data that trained the model. Some data miners advise that this is not good practice, that you should apply the model on a l different dataset to show that it predicts as expected. So, I decided to split the data to see if it would improve the performance of the data. The split operator was used. I then partitioned the data with 2 partitions of equal size 0.5. The first half of the data was used to train the model (train the linear regression). The second half was to evaluate the model (apply the linear regression).





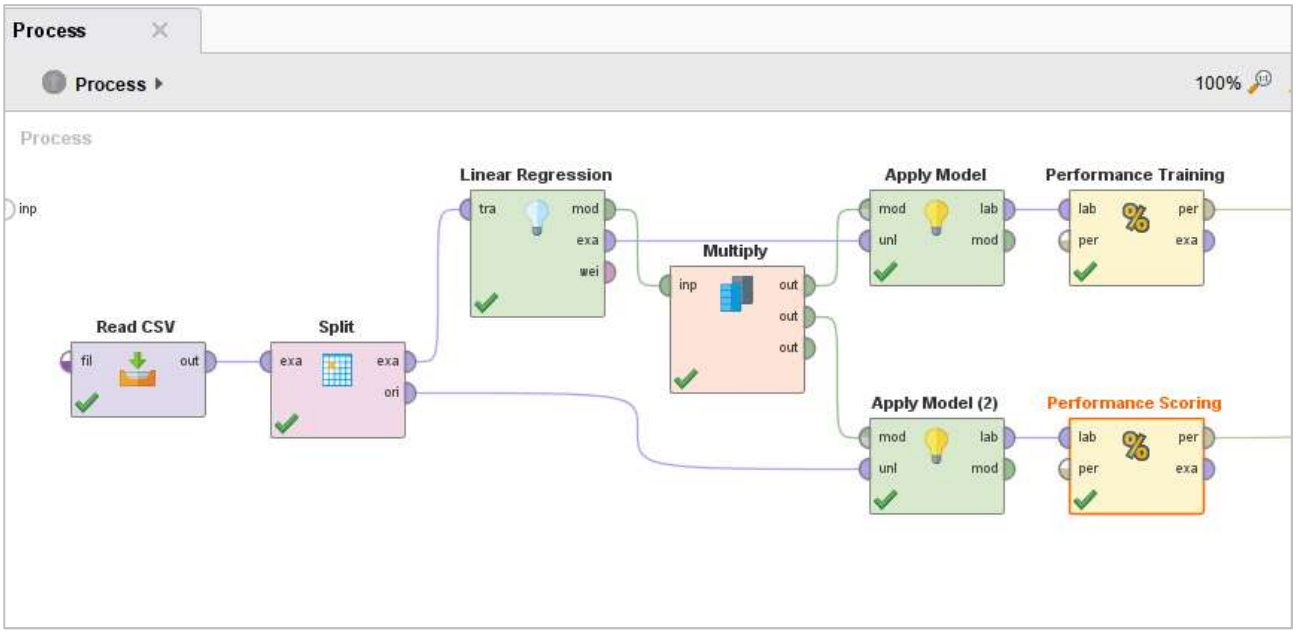
The squared correlation was still low. Splitting the data did not make much of a difference in this model.

Applying 2 performance operators

I wanted to see the performance simultaneously, so I included 2 performance operators. I applied the model over the original dataset and applied the model over the second partition. I then had 2 performance paths, the data was split in 2 halves. The first half was used to train the model, this model is evaluated over the same dataset.

I evaluated the model again by applying the second half of the dataset to the model. Please see image below for illustration.

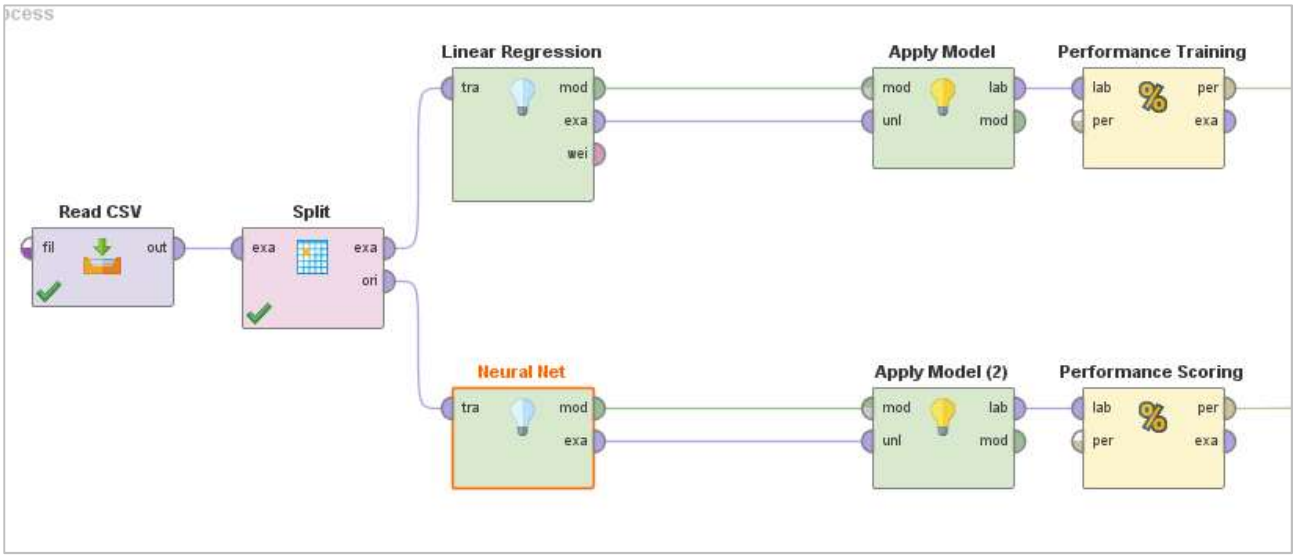
I renamed the performance operators to performance training and performance scoring to make the results easier to read and for other users to understand my model.



The squared correlation for the performance training and performance scoring were still the same at 19%.

Linear Regression and Neural Network in same model with 2 performance paths

In the previous model I had 2 performance paths, I decided to use this concept to put the linear regression operator and the neural network operator in the same model. The model was split into 2partitions - the first partition for the linear regression, the second partition for the neural network. There was no training in this model. Each data partition is used to test different parts of the model.



The output performance of this model was the same as the individual models for linear regression and neural networks. I prepared this model for learning purposes and for future use so other uses can replicate this neater model.

4.0 Conclusions

I spent a lot of time at the start of this assignment looking for data and trying to figure out how to obtain the data. If I was to do a similar assignment again and was faced with the same time constraints – I would probably pick a dataset from Kaggle or the UCI Machine learning library. This would enable me more time to focus on my models and analysis. I felt that I spent too much time on the data collection and data preparation stage of this assignment. However, I learned a lot about collecting data and the associated software and hardware constrains. I was also conflicted between developing the business case and the technical data mining goals. It was difficult switching between the 2 disciplines at times. In the real world, I understand that different people, different teams would be working on these 2 different aspects of the project.

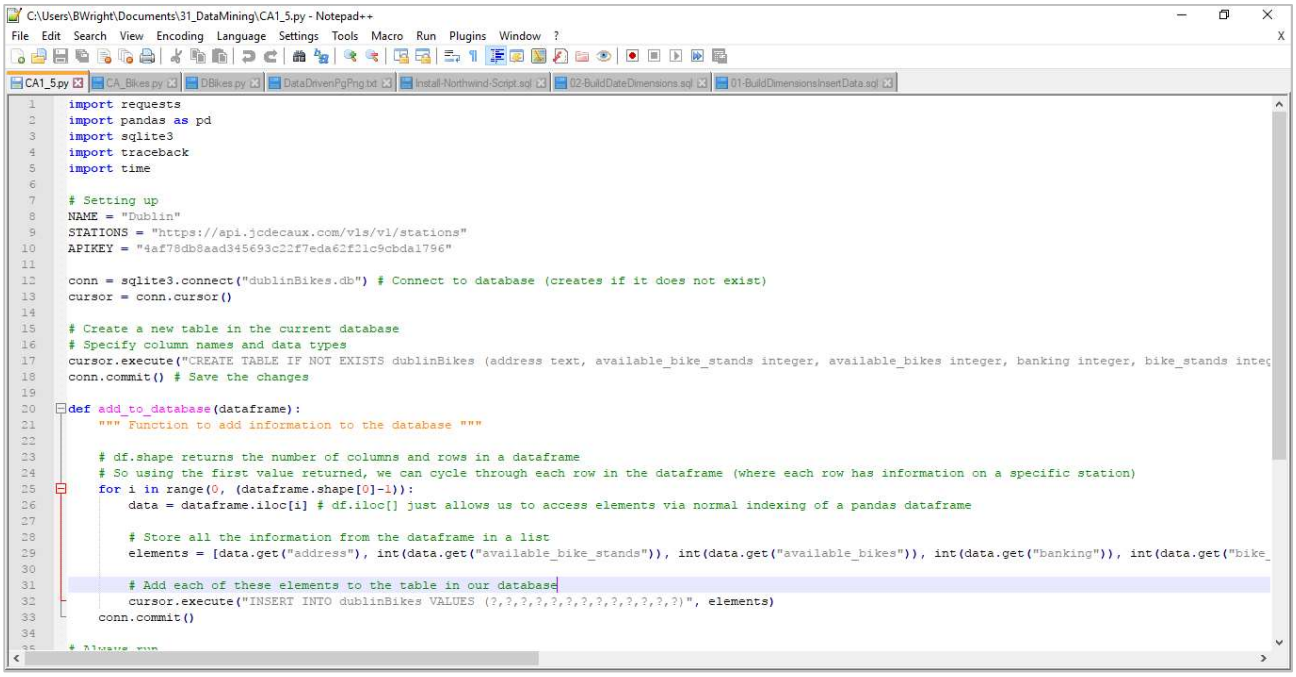
In this assignment, I developed a cluster model to identify groups of stations i.e. stations with similar use patterns. While this model was relatively straight forward, I spend a lot of time preparing data for it. However, I was happy with the output and how I was able to deploy the output in a real world spatial context on a map. I was able to identify differences in use patterns between weekdays and weekends through simple visualization charts in Excel. I carried out the same process for identifying use patterns throughout the day. Sampling was carried out at the exploratory data analysis stage. As I was faced with time constraints, I focused my remaining time on business objective 4. I was surprised that the neural network operator created a better prediction than my linear regression models. However, it should be noted that while had a lot of example data (circa 4,000 records) it all came from the 1 month. In the future, I would like to do further analysis using my season, month and holiday fields. However, an examination of a similar project on Kaggle stated that the season and month had very little impact on the number of bikes rented. It would have been useful to have 2

separate fields on casual and registered users. I think this would produce different results. I think casual users might be more influenced by poor weather conditions. I also think there are many external factors at play, that were not included in the model. Examples include the redistribution/ rebalancing of bikes by Dublin city council staff and the closure of stations. If I had a better fitting model, I would like to put in the weather and temporal data to predict the bike use. As my model only had 21 months of data, this would not be a fruitful exercise. I also tried different model operators such as decision tree, random forest and polynomial regression. Decision tree and random forest didn't provide useful results for my data. I like the polynomial regression operator, however I found that it was only suitable for data that has a unique record for each day as opposed to my data where each day had circa 720 records. The most challenging part of this assignment was using unprepared real-world data! The task would have been a lot easier if I used a clean academic dataset that was specifically designed for modelling and machine learning.

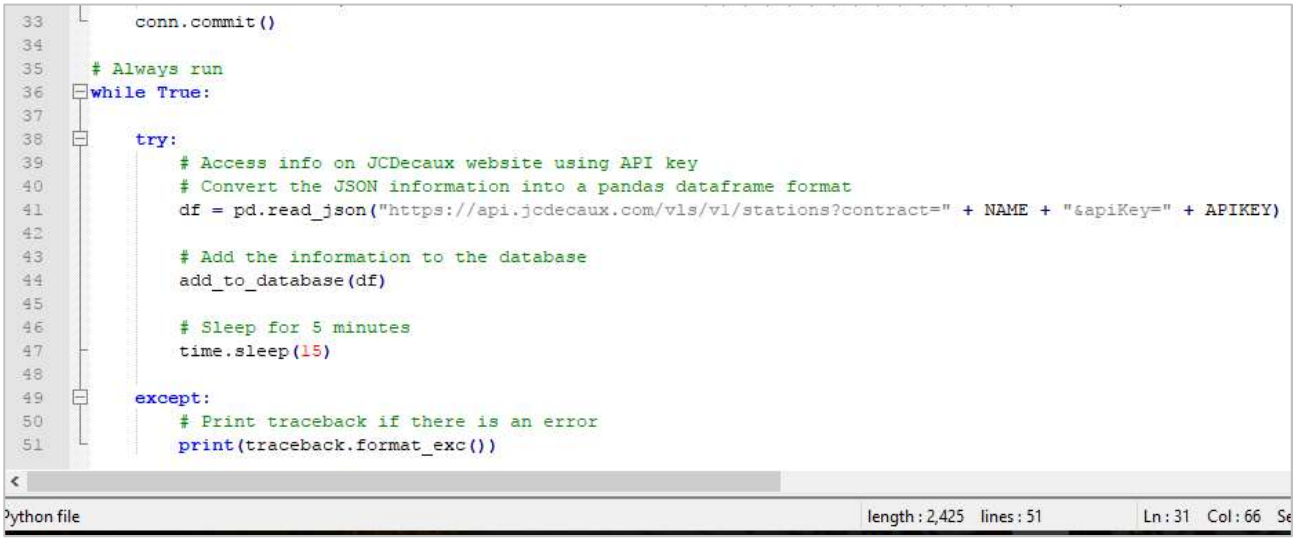
I have learned a lot about data mining and improved my RapidMiner, Excel and command prompt scripting skills. This assignment verified that high quality hardware / cloud services are better suited to processing, modelling and analysing big data. After my experience on this project, I will consider using cloud services on my final year project.

Appendix A

Python Code used to scrape Dublinbikes data into a SQLite database from the JCI Decaux API



```
import requests
import pandas as pd
import sqlite3
import traceback
import time

# Setting up
NAME = "Dublin"
STATIONS = "https://api.jcdecaux.com/vls/v1/stations"
APIKEY = "4af78db8aad345693c22f7eda62f21c9cbda1796"

conn = sqlite3.connect("dublinBikes.db") # Connect to database (creates if it does not exist)
cursor = conn.cursor()

# Create a new table in the current database
# Specify column names and data types
cursor.execute("CREATE TABLE IF NOT EXISTS dublinBikes (address text, available_bike_stands integer, available_bikes integer, banking integer, bike_stands integ
conn.commit() # Save the changes

def add_to_database(dataframe):
    """ Function to add information to the database """

    # df.shape returns the number of columns and rows in a dataframe
    # So using the first value returned, we can cycle through each row in the dataframe (where each row has information on a specific station)
    for i in range(0, (dataframe.shape[0]-1)):
        data = dataframe.iloc[i] # df.iloc[] just allows us to access elements via normal indexing of a pandas dataframe

        # Store all the information from the dataframe in a list
        elements = [data.get("address"), int(data.get("available_bike_stands")), int(data.get("available_bikes")), int(data.get("banking")), int(data.get("bike_

        # Add each of these elements to the table in our database
        cursor.execute("INSERT INTO dublinBikes VALUES (?,?,?,?,?,?,?,?,?,?,?,?)", elements)
    conn.commit()

# Always run
```



```
    conn.commit()

# Always run
while True:

    try:
        # Access info on JCDecaux website using API key
        # Convert the JSON information into a pandas dataframe format
        df = pd.read_json("https://api.jcdecaux.com/vls/v1/stations?contract=" + NAME + "&apiKey=" + APIKEY)

        # Add the information to the database
        add_to_database(df)

        # Sleep for 5 minutes
        time.sleep(15)

    except:
        # Print traceback if there is an error
        print(traceback.format_exc())
```

Python Code used to scrape Dublinbikes data into a CSV file from the CityBikes API

```
import requests
import pandas as pd
import pandas.io.sql as pdsql
from time import sleep, strftime, gmtime
import json
import sqlite3

# define the city you would like to get information from here:
# for full list see http://api.citybik.es
API_URL = "http://api.citybik.es/dublinbikes.json"

#Settings:
SAMPLE_TIME = 120                   # number of seconds between samples
SQLITE = False                      # If true - data is stored in SQLite file, if false - csv.
SQLITE_FILE = "bikedb.db"           # SQLite file to save data in
CSV_FILE = "output.csv"             # CSV file to save data in

def getAllStationDetails():
    print "\n\nScraping at " + strftime("%Y%m%d%H%M%S", gmtime())

    try:
        # this url has all the details
        decoder = json.JSONDecoder()
        station_json = requests.get(API_URL, proxies='')
        station_data = decoder.decode(station_json.content)
    except:
        print "---- FAIL ----"
        return None

    #remove unnecessary data - space saving
    # we dont need latitude and longitude
    for ii in range(0, len(station_data)):
        del station_data[ii]['lat']
        del station_data[ii]['lng']
        #del station_data[ii]['station_url']
```

```
        del station_data[ii]['lng']
        #del station_data[ii]['station_url']
        #del station_data[ii]['coordinates']

    print " --- SUCCESS --- "
    return station_data

def writeToCsv(data, filename="output.csv"):
    """
    Take the list of results and write as csv to filename.
    """
    data_frame = pd.DataFrame(data)
    data_frame['time'] = strftime("%Y%m%d%H%M%S", gmtime())
    data_frame.to_csv(filename, header=False, mode="a")

def writeToDb(data, db_conn):
    """
    Take the list of results and write to sqlite database
    """
    data_frame = pd.DataFrame(data)
    data_frame['scrape_time'] = strftime("%Y%m%d%H%M%S", gmtime())
    pdsql.write_frame(data_frame, "bikedata", db_conn, flavor="sqlite", if_exists="append", )
    db_conn.commit()

if __name__ == "__main__":

    # Create / connect to Sqlite3 database
    conn = sqlite3.connect(SQLITE_FILE) # or use :memory: to put it in RAM
    cursor = conn.cursor()
    # create a table to store the data
    cursor.execute("""CREATE TABLE IF NOT EXISTS bikedata
                      (name text, idx integer, timestamp text, number integer,
                       free integer, bikes integer, id integer, scrape_time text)
                   """)
    conn.commit()
```

```
79      conn.commit()
80
81      #run main function
82      # we need to run the full collection, parsing, and writing every minute.
83      while True:
84          station_data = getAllStationDetails()
85          if station_data:
86              if SQLITE:
87                  writeToDb(station_data, conn)
88              else:
89                  writeToCsv(station_data, filename=CSV_FILE)
90
91          print "Sleeping for 120 seconds."
92          sleep(120)
```

References and Resources

1) Smart Vision - Europe. 2017. What is the CRISP-DM methodology?. [ONLINE] Available at: http://www.sv-europe.com/crisp-dm-methodology/. [Accessed 14 October 2017].

2) YouTube. (2017). Predicting the Stock Value using RapidMiner. [online] Available at: https://www.youtube.com/watch?v=LbtZU1_i9Qk [Accessed 10 Dec. 2017].

3) YouTube. (2017). RapidMiner - Linear Regression. [online] Available at: https://www.youtube.com/watch?v=p_wNcfAHlY8 [Accessed 10 Dec. 2017].

4) YouTube. (2017). RapidMiner Advanced Analytics Demonstration: Predicting Survival of the Titanic Accident. [online] Available at: https://www.youtube.com/watch?v=54CRc33WtaE [Accessed 10 Dec. 2017].

5) Elearning.dbs.ie. (2017). POST data. [online] Available at: https://elearning.dbs.ie/pluginfile.php/634704/mod_resource/content/1/Regression%20Algorithms%20.pdf [Accessed 10 Dec. 2017].

6) Elearning.dbs.ie. (2017). *POST data*. [online] Available at: https://elearning.dbs.ie/pluginfile.php/636245/mod_resource/content/0/Clustering%20Algorithms.pdf [Accessed 15 Nov. 2017].

7) Elearning.dbs.ie. (2017). POST data. [online] Available at: https://elearning.dbs.ie/pluginfile.php/626489/mod_resource/content/0/CRISP-DM.pdf [Accessed 15 Nov. 2017].

8) Met.ie. (2017). Download Historical Climate Data - Met Éireann - The Irish Meteorological Service Online. [online] Available at: http://www.met.ie/climate-request/ [Accessed 6 Nov. 2017].

9) Api.citybik.es. (2017). Documentation | CityBikes API. [online] Available at: https://api.citybik.es/v2/ [Accessed 6 Nov. 2017].

10) Data.dublinked.ie. (2017). dublinbikes - Dublin Bikes API - Dublinked. [online] Available at: https://data.dublinked.ie/dataset/dublinbikes/resource/d008c9bf-f34b-46e0-9bc2-a099d728fa1e [Accessed 30 Oct. 2017].

11) Developer.jcdecaux.com. (2017). JCDecaux Developer. [online] Available at: https://developer.jcdecaux.com/#/opendata/vls?page=getstarted [Accessed 3 Nov. 2017].

12) GitHub. (2017). jameslawlor/Dublin-Bikes-Analysis. [online] Available at: https://github.com/jameslawlor/Dublin-Bikes-Analysis [Accessed 3 Nov. 2017].

13) GitHub. (2017). jameslawlor/dublin-bikes-timeseries-analysis. [online] Available at: https://github.com/jameslawlor/dublin-bikes-timeseries-analysis [Accessed 15 Oct. 2017].

14) Olivernash.org. (2017). Dublin bikes revisted. [online] Available at: http://olivernash.org/2011/02/02/dublin-bikes-revisited/ [Accessed 30 Oct. 2017].

15) Kaggle.com. (2017). Bike Sharing Demand | Kaggle. [online] Available at: https://www.kaggle.com/c/bike-sharing-demand [Accessed 30 Nov. 2017].