Uncover the demand for community resources by CIE data

Li Jen Shao(N11096837)

IFN703/4 Assessment 3

Abstract

In this analysis we tried various of pre-processing techniques to retrieve missing information in the CIE data. Eventually, we want to find out any interesting patterns in the data by answering 3 core questions, which are When did the user search, What are they searching for, and Where are the users. We used DBIP API to retrieve postcode and filter out unnecessary information, and retrieved part of the What information by mixing 3 columns that are critical to know the intension of the user. Finally, by a series of visualisation we were able to find out some evidence that maybe people are using My Community Directory website for seeking community resources during the great 2022 Queensland flood. The methodology which performed in this analysis is not outstanding, but it can be a reference for the future analysis.

Introduction

My Community Directory

The idea of Community Information Exchange (CIE) is to build an integrated open and transparent platform that can easily help people to access information from various of community services and local events, no matter they are private or public. The organisation of Community Information Support Services is the non-for-profit organisation who supports the platform. Just like Ebay and Amazon in Ecommerce, this kind of integrated platform can let people search for what they want from a wide range of variety instead of access it one by one in the old days. The website of My Community Directory is the search engine and the entrance for different kinds of local community resources. This integration has effectively reduced the time consumption and increased the efficiency on connecting people’s need with community service provider. With various of community services, the social safety nets can be built to support and improve people’s wellbeing.

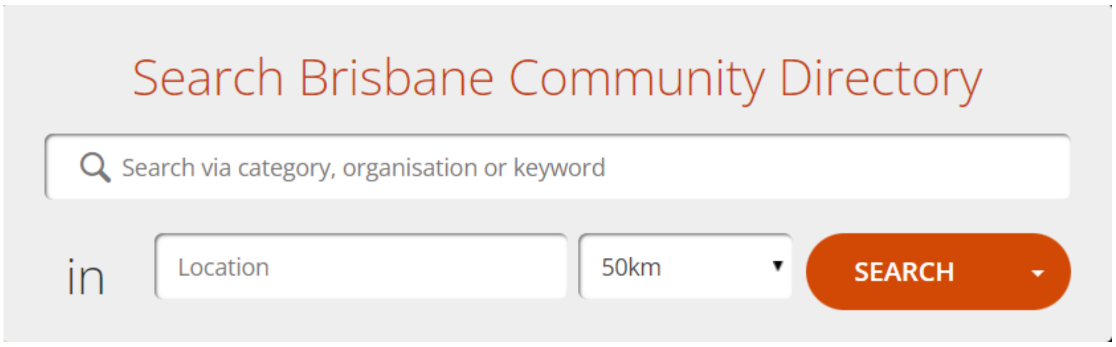

Search Brisbane Community Directory
Search via category, organisation or keyword
in
Location
50km
SEARCH

Figure 1: The search engine of My Community Directory [1].

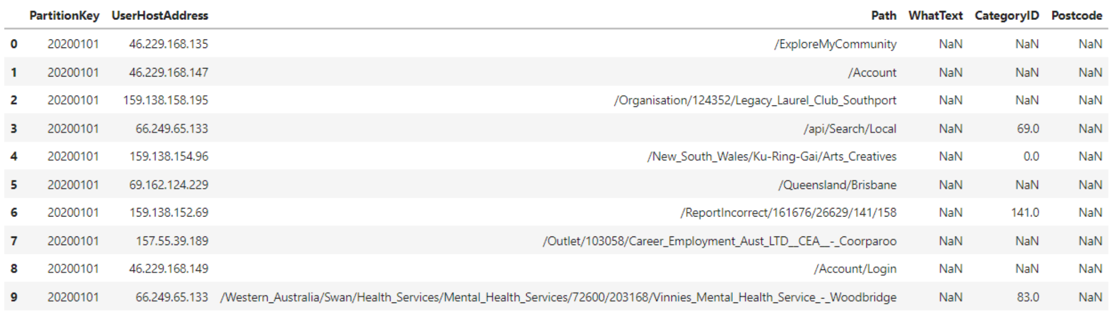
According to the talk with CIE, the website has been widely used by people for several years, it can generate more than million search query records in a month, all the records are generated from My Community Directory website. From an initial observation of how website operates, we can understand that the search query records should contain this important information of user such as:

- The free text that user used for searching.
- The location of the user.
- The geographical search range that the user would like to include from his location.

Therefore, in this analysis we would like to investigate people's demand in relates to their corresponding geographical location and time. This can help us to figure out when people search, what are people looking for, and where are the users. Eventually, try to answer an ultimate question which is "Is there any interesting trend in terms of when, what and where".

The data

The time span of the CIE data is from January 2020 to September 2022, each dataset contains a month of search queries. In this analysis we will focus on January 2020 and February 2022 dataset, the reason of using these two datasets is to show the contrast from two different space time. January 2020 would be a very normal month even before Covid-19 pandemic happen, February 2022 is the month when Queensland had big flood that impacted Southeast QLD and Wide Bay-Burnett.



	PartitionKey	UserHostAddress	Path	WhatText	CategoryID	Postcode
0	20200101	46.229.168.135	/ExploreMyCommunity	NaN	NaN	NaN
1	20200101	46.229.168.147	/Account	NaN	NaN	NaN
2	20200101	159.138.158.195	/Organisation/124352/Legacy_Laurel_Club_Southport	NaN	NaN	NaN
3	20200101	66.249.65.133	/api/Search/Local	NaN	69.0	NaN
4	20200101	159.138.154.96	/New_South_Wales/Ku-Ring-Gai/Arts_Creatives	NaN	0.0	NaN
5	20200101	69.162.124.229	/Queensland/Brisbane	NaN	NaN	NaN
6	20200101	159.138.152.69	/ReportIncorrect/161676/26629/141/158	NaN	141.0	NaN
7	20200101	157.55.39.189	/Outlet/103058/Career_Employment_Aust_LTD__CEA_-_Coorparoo	NaN	NaN	NaN
8	20200101	46.229.168.149	/Account/Login	NaN	NaN	NaN
9	20200101	66.249.65.133	/Western_Australia/Swan/Health_Services/Mental_Health_Services/72600/203168/Vinnies_Mental_Health_Service_-_Woodbridge	NaN	83.0	NaN

Figure 2: A sanity check for the initial data, these are the columns we will use in this analysis. We can

see a lot of missing value (NaN) in the dataset.

Initially, the January 2020 dataset has 8073663 samples and 63 columns. The February 2022 dataset has 14208383 samples and 66 columns. From initial observation we can understand that in the datasets we do have some columns that directory relates to the question we mentioned previously. Thus, we will focus on these 6 columns and try to extract useful information from it.

- PartitionKey: A column that directly relates to time, this can tell us when the search query was generated.
- UserHostAddress: User's IP address, a unique identifier.
- Path: Indicates the path directory of the search query, this column can help to answer what is user looking for.
- WhatText: This column directly relates to what is user looking for.
- Category ID: This column can help to answer what is user looking for.
- Postcode: This column directly relates to where is the user.

From figure 2 we also realised a very serious problem, which is the missing value. Even though we have found the columns that directly relates to when, what and where, we still cannot extract anything useful with such large amount of missing value. For instance, in January 2020 datasets, we have 8073663 samples in total, the amount of missing value in WhatText column is 7932406, in Category ID is 6310867, in Postcode is 7968427.

How the missing value generate?

The amount of missing value seems destructive to the analysis, we would potentially lose a lot of information if we arbitrary get rid of the missing value in every column. Considering how the missing value were generated, if user didn't insert any free text into the search engine when searching, WhatText column should be empty, same logic for the Postcode and Category ID. Therefore, some search queries are supposed to be missing value, but not all of them. There are several ways to conduct searching on My Community Directory website, we can either insert free text into the search engine or click on the categories as it would return a list of nearest services which are in the category. For some unknown reason we have tons of search queries that their WhatText, Category ID and Postcode are all empty, does this means user did not do anything on the website? Or they did something but is not related to searching? In the literature review we will discuss the potential option of handling the large amount of missing value in CIE data.

Literature Review

Just when we are hesitating about how to proceed the analysis in this stage, our colleague David and Sean sent us some good news. They found a breaching point which is using the column of UserHostAddress to locate the users. By using each search query's IP address, the DBIP API [2] can return not only the corresponding postcode, but also the country of the IP address and identify whether the IP address is a bot or web crawler at the same time, this method allows us to retrieve the information in postcode column as much as possible while reducing not useful information by filtering the IP address. Therefore, the only problem now is the missing value in WhatText, how do we retrieve the information in WhatText?

In the proposal we proposed a text clustering method that can help us to assign a topic for each search query, this will allow us to reduce the number of unique values of free text by categorise search queries, then we can use these categories as the WhatText information to know what people are looking for generally. Latent Dirichlet Allocation (LDA) [3] is one of such techniques. However, in the practice we found that the free text in WhatText column is too short so the algorithm could not perform clustering well. Indeed, comparing to the example that Blei, Andrew and Micheal did in their research, our free text does not have enough words for algorithm to associate them.

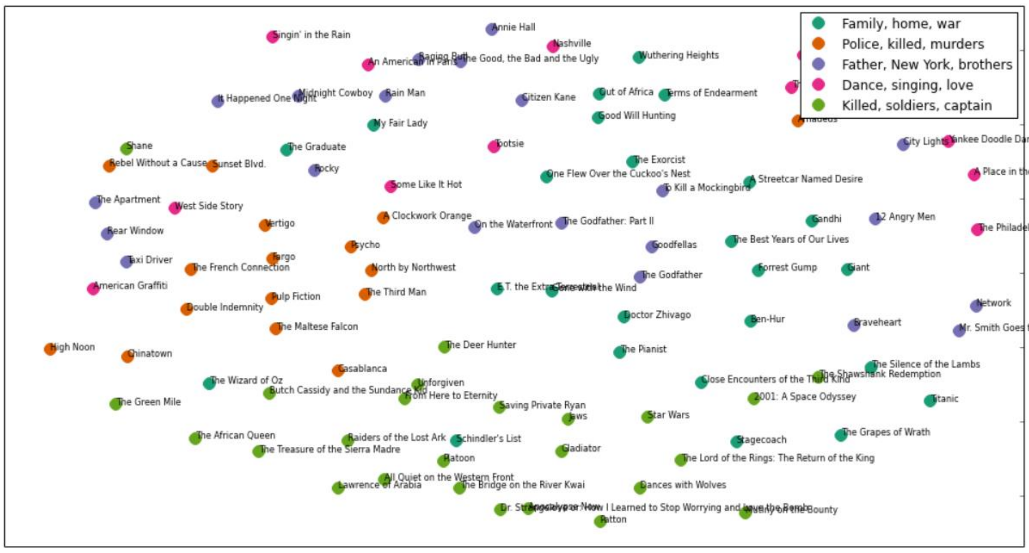


Figure 3: Ideally, categorising our WhatText should be very similar to clustering the movie names [5].

Another clustering technique is Dirichlet Multinomial Mixture Model (GSDMM) [4] for short text clustering, this method requires much shorter length of text as input than LDA, in the research Yin and Wang used google news title and tweets as example to show the short text clustering technique. However, in the practice GSDMM did not perform well as well, our free text is still too short comparing to google news title and tweets so that the algorithm had difficulties on associating different text. If we investigate further on the value of WhatText we can easily understand why text clustering algorithm performs bad on our free text. For instance, it is hard to know that the word "blue care" literally would be categorised

as mental health services, this is because there is no particular pattern in the nature of human behavior when giving a name, without additional information it would be hard to categorise customised name with machine learning algorithms.
Eventually, the final option would be cluster them by ourselves. This is the most time consuming and tedious method, but it seems the only option for now. We can try to categorise the value of WhatText based on the existing category name, then assign a category by hand if the search query has missing category. This option can ensure most search queries can have a relatively accurate category assigned, but we need to ignore the minorities because we will not have the time and patience to do this one by one until each search query has been assigned a category. With this method we can only ensure a major part of the WhatText can have category with them, and it allow us to generate somewhat useful result.

Approach

Filtering and retrieving the information of Where

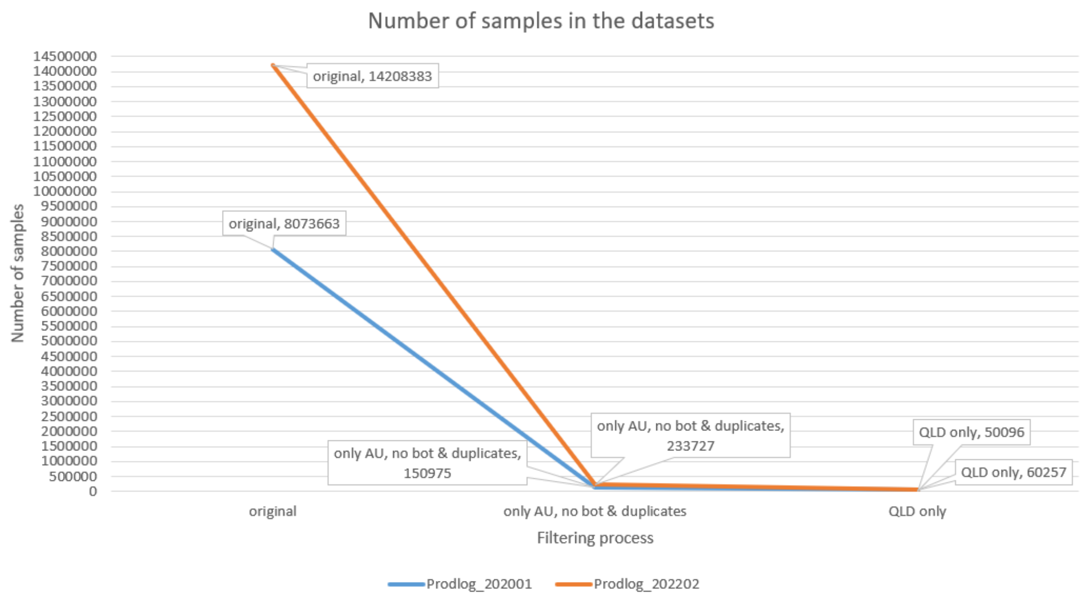


Figure 4: A visualisation of number of samples in both datasets before and after filtering.

Initially, the volume of both January 2020 and February 2022 datasets seems huge, but from the previous investigation we found there are lots of duplicate records, duplicates here means a user can have multiple attempts within one intention. For instance, if a user is seeking mental health service, he probably would search for several times to get the optimal result, but the whole process would still be treated as 1 demand for mental health service. In the first stage we want to make sure that each search query represents 1 demand so that the analysis result can reflects the demand as possible as it can. We also filtered out the non-Australia IP address by access DBIP API with user's IP address, because CIE only operates within Australia.

From the information provided by DBIP API we can know that there is a large portion of IP address are not from Australia and some of them are even google bot and web crawler. Web crawler is a type of bot that typically operated by search engines like Google, the purpose of it is to help google index the content of websites across the internet so that the websites can appear in search engine results, this type of search query records does not represent any demand from users so we need to filter them out. Finally, we filter out any IP address that is not from Queensland because we want to focus on Queensland data. From figure 4 we can see a huge amount of unnecessary data has been dropped after filtering, the true meaningful data is only a small proportion in the originals.

Access DBIP API with user's IP address can also help us to locate geographical location of users, and it allows us to fulfill the missing value in Postcode column. However, the location that DBIP API returns is actually the location of the router, which is operated by telecommunication company. That means we can only recover the location of users as accurate as possible and it will never be the true location of users, I believe this is the best result we can achieve now in terms of retrieving information for Postcode.

Retrieve the information of What

Thanks to our colleague David and Sean, we can retrieve the information for Postcode column as much as possible, and it solved our question of where the users are. In terms of What are the users looking for, we can look at the column of WhatText, Path and Category ID, WhatText indicates the terms that the user used on searching, Path indicates where did the user ended in the My Community Directory website, Category ID indicates the category that the search result belongs to, these 3 columns can directly or indirectly tell what the user is looking for. Therefore, we came up with a strategy to mix the information of the 3 columns and categorise the search query base on the mixed information.

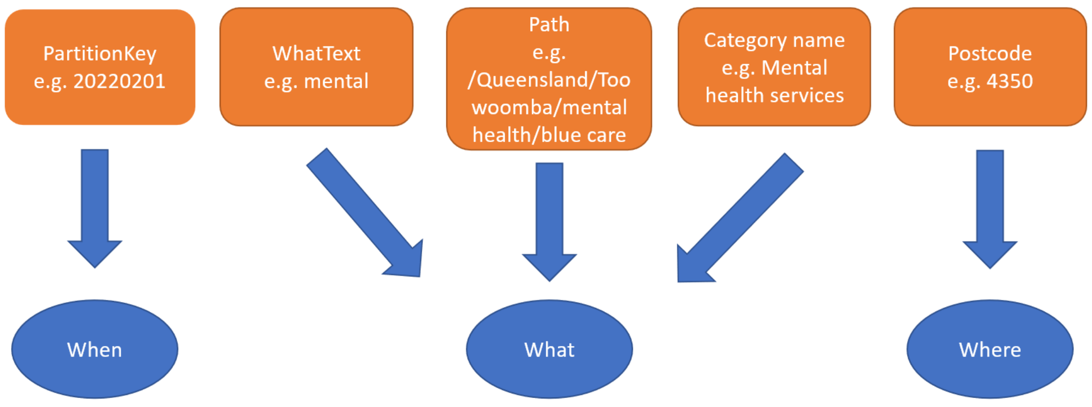
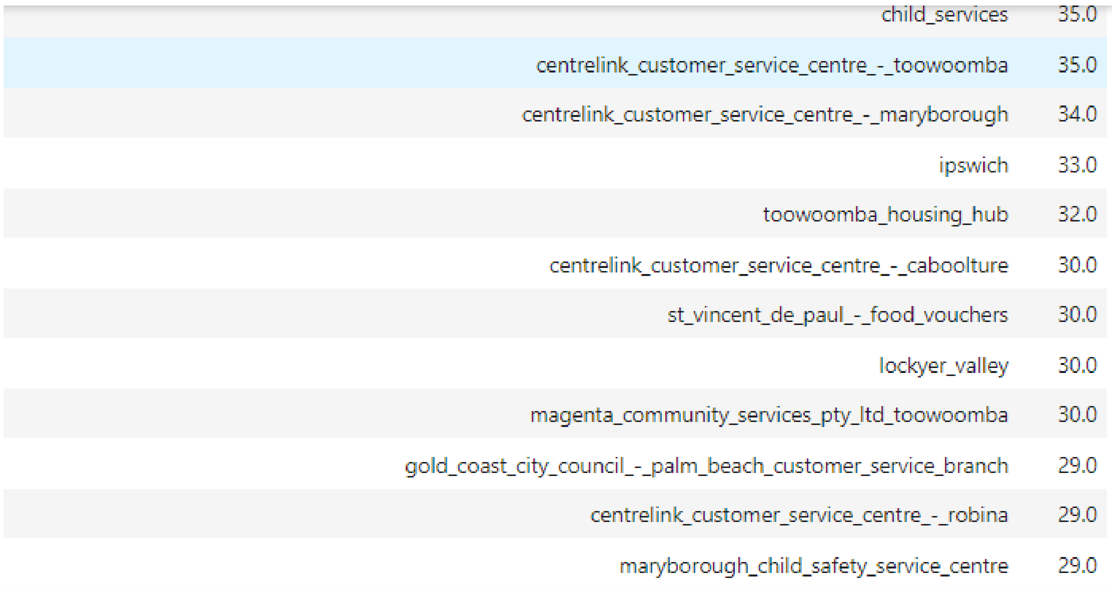

PartitionKey e.g. 20220201
WhatText e.g. mental
Path e.g. /Queensland/Toowoomba/mental health/blue care
Category name e.g. Mental health services
Postcode e.g. 4350
When
What
Where

Figure 5: A demonstration of how to answer 3 critical questions by existing information.

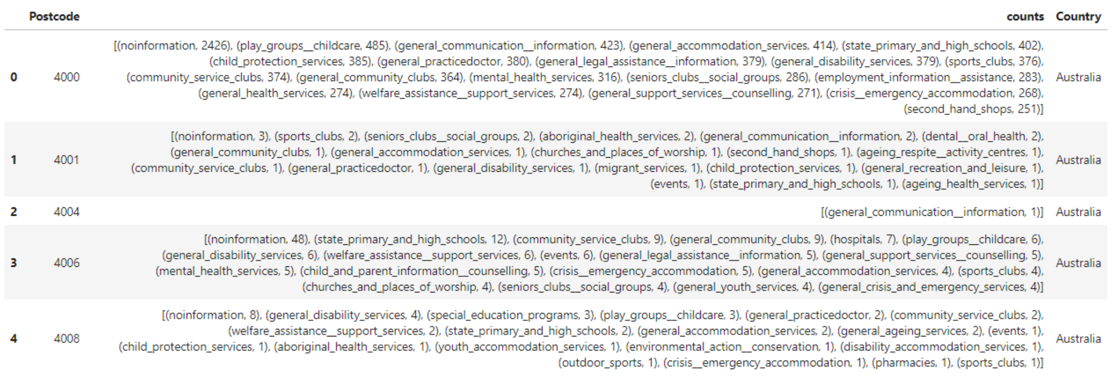
Eventually we will manually categorise the search query based on the mixed information composed by the free text from WhatText, the last part of Path and category name derived by category ID. From observation we found the Path column sometimes contains important information about what the user is looking for, so the last part of the directory from path can also be used to comprise the mix information. In order to reduce the word variation, we also converted all the alphabetic letters to lower case. Once we have the mixed information, we can start to manually categorise the search queries based on the value of it from high frequency terms to low frequency terms. Finally, the manual categorisation stops at word frequency of 5, simply because we reckon the process would significantly slower than before and not worth to do so. For those mixed information that their word frequency is less than 5, or the value of it seems cannot be categorised (e.g. ipswich), we categorise them as “No_Information” so that when analysing the final result we can still understand how much gray area we have in the analysis.



child_services	35.0
centrelink_customer_service_centre_-_toowoomba	35.0
centrelink_customer_service_centre_-_maryborough	34.0
ipswich	33.0
toowoomba_housing_hub	32.0
centrelink_customer_service_centre_-_caboolture	30.0
st_vincent_de_paul_-_food_vouchers	30.0
lockyer_valley	30.0
magenta_community_services_pty_ltd_toowoomba	30.0
gold_coast_city_council_-_palm_beach_customer_service_branch	29.0
centrelink_customer_service_centre_-_robina	29.0
maryborough_child_safety_service_centre	29.0

Figure 6: A look for the mixed information and their corresponding frequency counts, a big proportion of the search query had been assigned back with a category based on their mixed value.

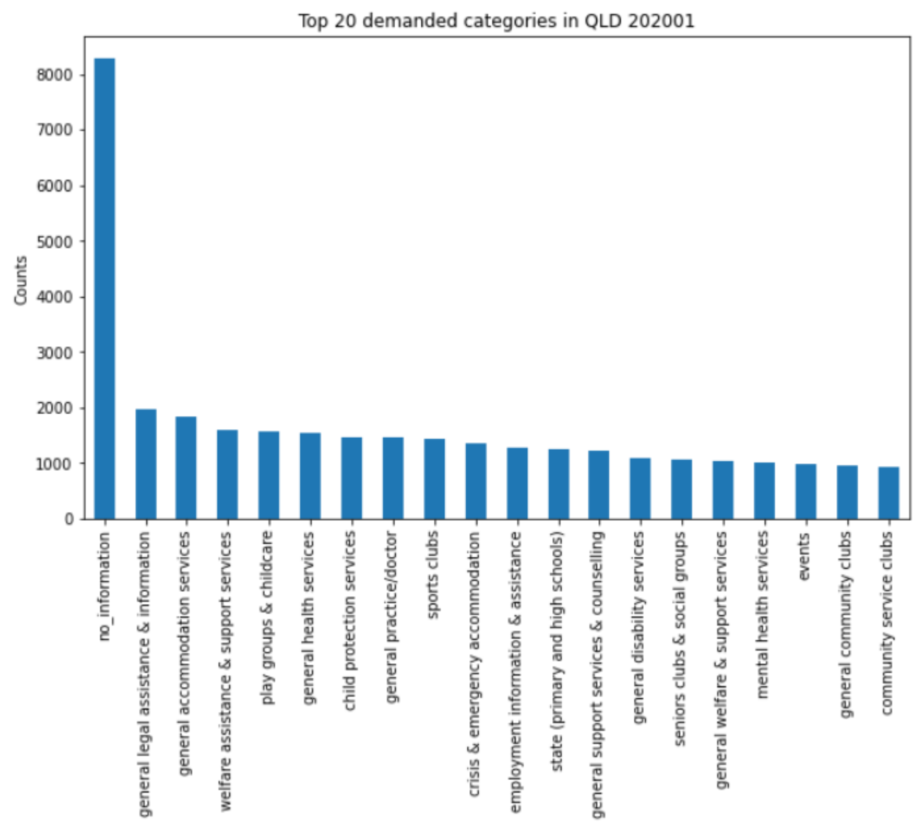
The Final step would be removing all the punctuation, stop words and tokenize all the value of new categories, then count the frequency of each new category in their corresponding postcode, this can let us know what new categories people have searched in each Queensland postcode without any missing values, then visualise the data.



	Postcode	counts	Country
0	4000	[(noinformation, 2426), (play_groups__childcare, 485), (general_communication__information, 423), (general_accommodation_services, 414), (state_primary_and_high_schools, 402), (child_protection_services, 385), (general_practicedoctor, 380), (general_legal_assistance__information, 379), (general_disability_services, 379), (sports_clubs, 376), (community_service_clubs, 374), (general_community_clubs, 364), (mental_health_services, 316), (seniors_clubs__social_groups, 286), (employment_information__assistance, 283), (general_health_services, 274), (welfare_assistance__support_services, 274), (general_support_services__counselling, 271), (crisis__emergency_accommodation, 268), (second_hand_shops, 251)]	Australia
1	4001	[(noinformation, 3), (sports_clubs, 2), (seniors_clubs__social_groups, 2), (aboriginal_health_services, 2), (general_communication__information, 2), (dental__oral_health, 2), (general_community_clubs, 1), (general_accommodation_services, 1), (churches_and_places_of_worship, 1), (second_hand_shops, 1), (ageing_respite__activity_centres, 1), (community_service_clubs, 1), (general_practicedoctor, 1), (general_disability_services, 1), (migrant_services, 1), (child_protection_services, 1), (general_recreation_and_leisure, 1), (events, 1), (state_primary_and_high_schools, 1), (ageing_health_services, 1)]	Australia
2	4004	[(general_communication__information, 1)]	Australia
3	4006	[(noinformation, 48), (state_primary_and_high_schools, 12), (community_service_clubs, 9), (general_community_clubs, 9), (hospitals, 7), (play_groups__childcare, 6), (general_disability_services, 6), (welfare_assistance__support_services, 6), (events, 6), (general_legal_assistance__information, 5), (general_support_services__counselling, 5), (mental_health_services, 5), (child_and_parent_information__counselling, 5), (crisis__emergency_accommodation, 5), (general_accommodation_services, 4), (sports_clubs, 4), (churches_and_places_of_worship, 4), (seniors_clubs__social_groups, 4), (general_youth_services, 4), (general_crisis_and_emergency_services, 4)]	Australia
4	4008	[(noinformation, 8), (general_disability_services, 4), (special_education_programs, 3), (play_groups__childcare, 3), (general_practicedoctor, 2), (community_service_clubs, 2), (welfare_assistance__support_services, 2), (state_primary_and_high_schools, 2), (general_accommodation_services, 2), (general_ageing_services, 2), (events, 1), (child_protection_services, 1), (aboriginal_health_services, 1), (youth_accommodation_services, 1), (environmental_action__conservation, 1), (disability_accommodation_services, 1), (outdoor_sports, 1), (crisis__emergency_accommodation, 1), (pharmacies, 1), (sports_clubs, 1)]	Australia

Figure 7: A sanity check of the processed data, now each postcode has its corresponding categories and counts.

Findings



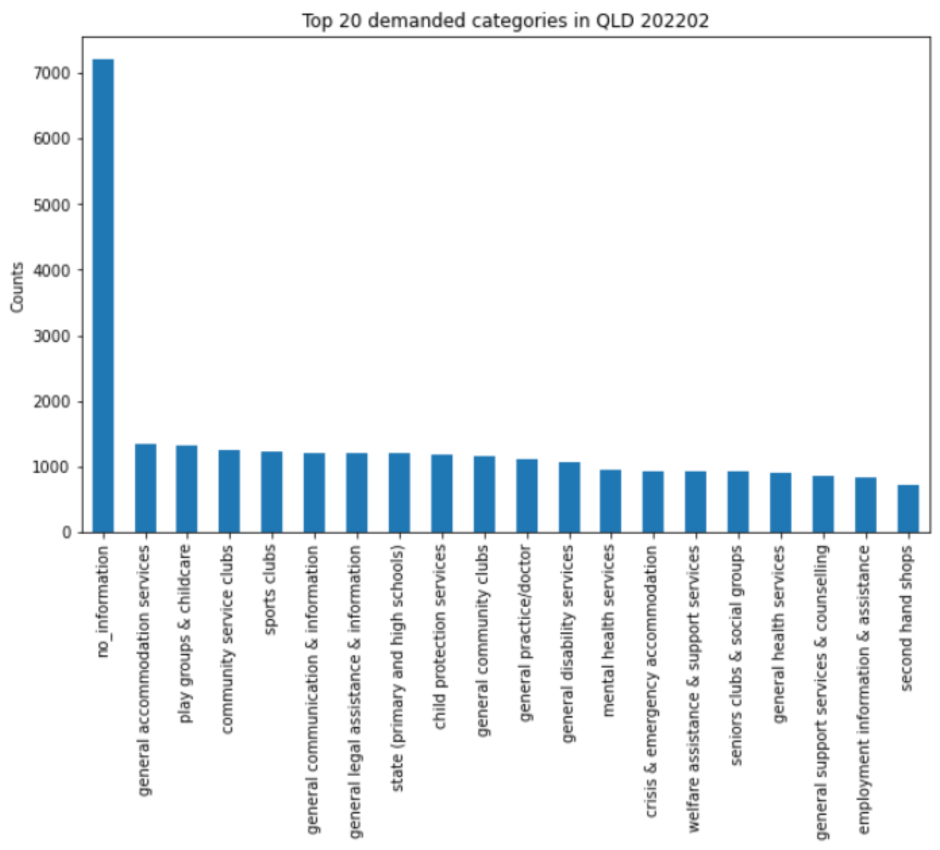
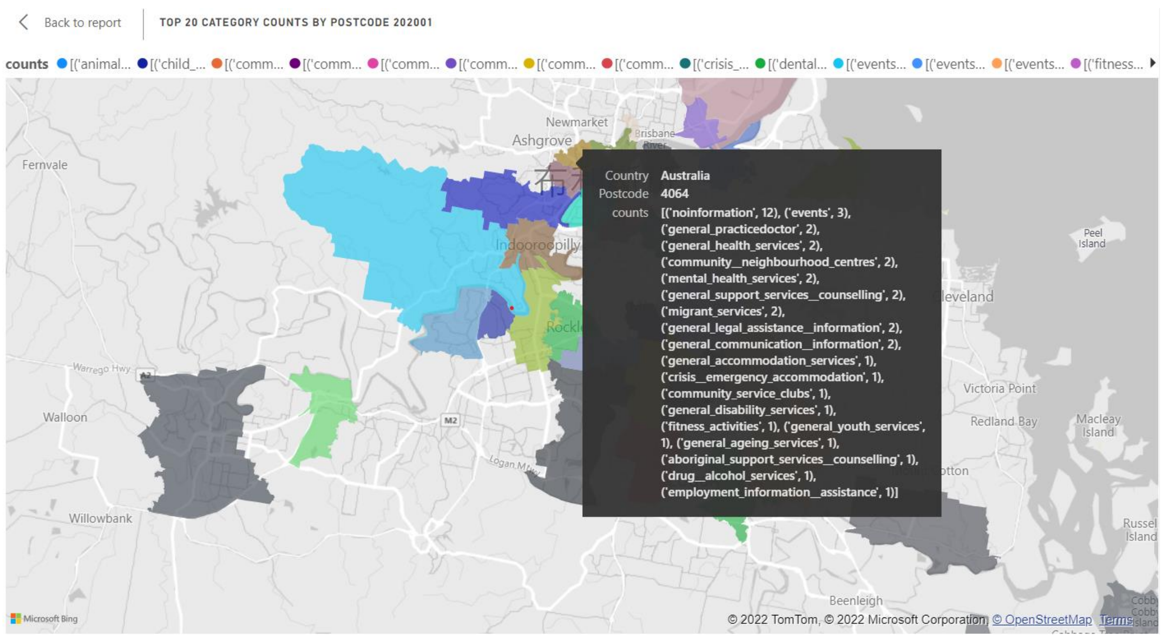


Figure 8: An overview of top 20 demanded categories in both January 2020 and February 2022.

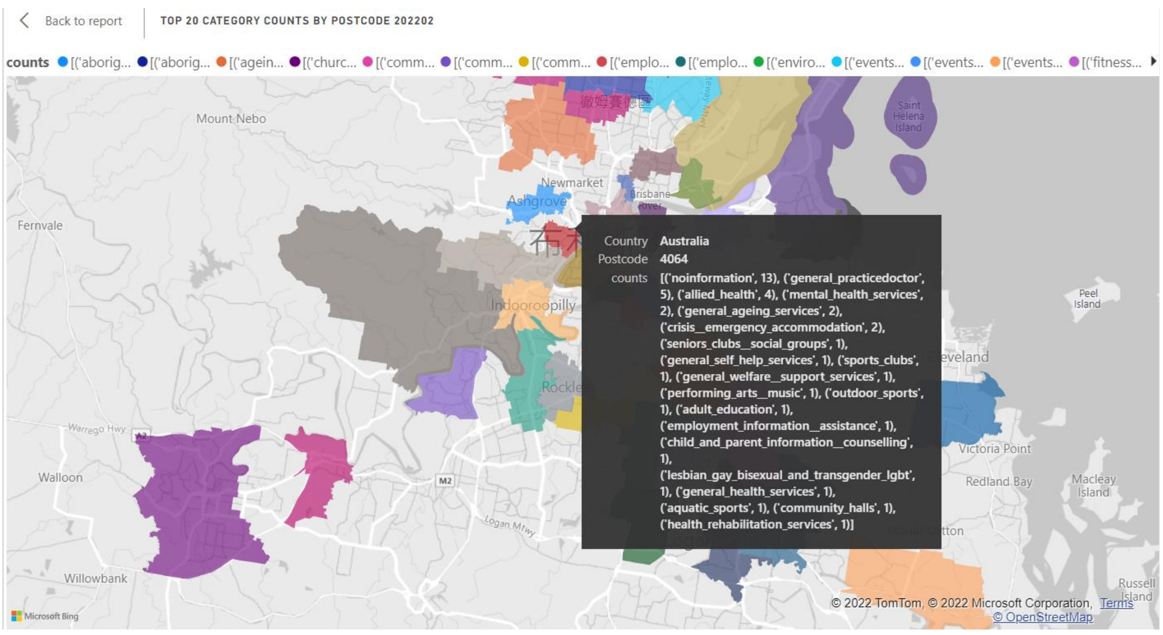
To interpret the result, we first have a look at the overview of whole Queensland.

Comparing the two histograms in figure 8 we can understand that in commonly the category of most needed service in QLD are "General accommodation services", and "Play groups & childcare". The number of No_Information is taking a big proportion of the data, this is because No_Information is composed by true missing value, the mixed information value which does not seems like a service and the mixed information value that their frequency is less than 5. Therefore, we still need to aware that a portion of the data is unexplainable.



Back to report
TOP 20 CATEGORY COUNTS BY POSTCODE 202001
counts ● [('animal... ● [('child_... ● [('comm... ● [('comm... ● [('comm... ● [('comm... ● [('comm... ● [('comm... ● [('crisis_... ● [('dental... ● [('events... ● [('events... ● [('events... ● [('fitness...
Fernvale
Newmarket
Ashgrove
Indooroopilly
Walloon
Willowbank
Cleveland
Victoria Point
Redland Bay
Macleay Island
Peel Island
Russell Island
Beenleigh
Microsoft Bing
© 2022 TomTom, © 2022 Microsoft Corporation, © OpenStreetMap Terms

Country Australia
Postcode 4064
counts [('noinformation', 12), ('events', 3), ('general_practicedoctor', 2), ('general_health_services', 2), ('community__neighbourhood_centres', 2), ('mental_health_services', 2), ('general_support_services__counselling', 2), ('migrant_services', 2), ('general_legal_assistance__information', 2), ('general_communication__information', 2), ('general_accommodation_services', 1), ('crisis__emergency_accommodation', 1), ('community_service_clubs', 1), ('general_disability_services', 1), ('fitness_activities', 1), ('general_youth_services', 1), ('general_ageing_services', 1), ('aboriginal_support_services__counselling', 1), ('drug__alcohol_services', 1), ('employment_information__assistance', 1)]



Back to report
TOP 20 CATEGORY COUNTS BY POSTCODE 202202
counts ● [('aborig... ● [('aborig... ● [('agein... ● [('churc... ● [('comm... ● [('comm... ● [('comm... ● [('emplo... ● [('emplo... ● [('enviro... ● [('events... ● [('events... ● [('events... ● [('fitness...
Mount Nebo
Fernvale
Newmarket
Ashgrove
Indooroopilly
Walloon
Willowbank
Saint Helena Island
Peel Island
Cleveland
Victoria Point
Redland Bay
Macleay Island
Russell Island
Microsoft Bing
© 2022 TomTom, © 2022 Microsoft Corporation, Terms © OpenStreetMap

Country Australia
Postcode 4064
counts [('noinformation', 13), ('general_practicedoctor', 5), ('allied_health', 4), ('mental_health_services', 2), ('general_ageing_services', 2), ('crisis__emergency_accommodation', 2), ('seniors_clubs__social_groups', 1), ('general_self_help_services', 1), ('sports_clubs', 1), ('general_welfare__support_services', 1), ('performing_arts__music', 1), ('outdoor_sports', 1), ('adult_education', 1), ('employment_information__assistance', 1), ('child_and_parent_information__counselling', 1), ('lesbian_gay_bisexual_and_transgender_lgbt', 1), ('general_health_services', 1), ('aquatic_sports', 1), ('community_halls', 1), ('health_rehabilitation_services', 1)]

Figure 9: The dynamic maps for two datasets made by Power BI, this is just a clip for demonstration, you can access the dynamic map in the CIE.pbix (a POWER BI file).

With the help by Power BI, we can generate the map like the demonstration shown in figure 9. These maps allow us to look into the details of what categories were needed during January 2020 and February 2022 in QLD, and we are able to compare the result that generated in different time. The first thing that came up with in my mind was the population density of Queensland. Indeed, the distribution of the population looks very like the distribution of each category's frequency by postcode in the map, potentially there would be more demand for community services with more population within an area.

Any evidence for the 2022 Queensland flood?

The 2022 Queensland flood happened during 22 February to 5 April, since we have processed the February data, we would like to know whether we can find anything abnormal in the February data. The assumption here is people were looking for temporary accommodations and welfare assistance services if they had been affected by the great flood. Thus, here we focus on the category of general accommodation services, crisis & emergency accommodation, and Welfare assistance & support services.

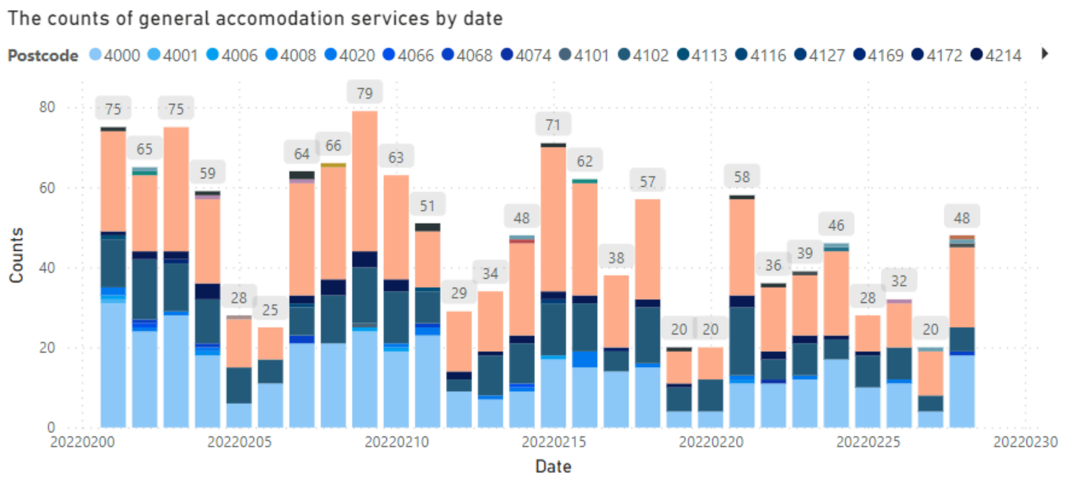


Figure 10: A histogram shows the demand of general accommodation services in QLD by date, we do not see any special trends in here.

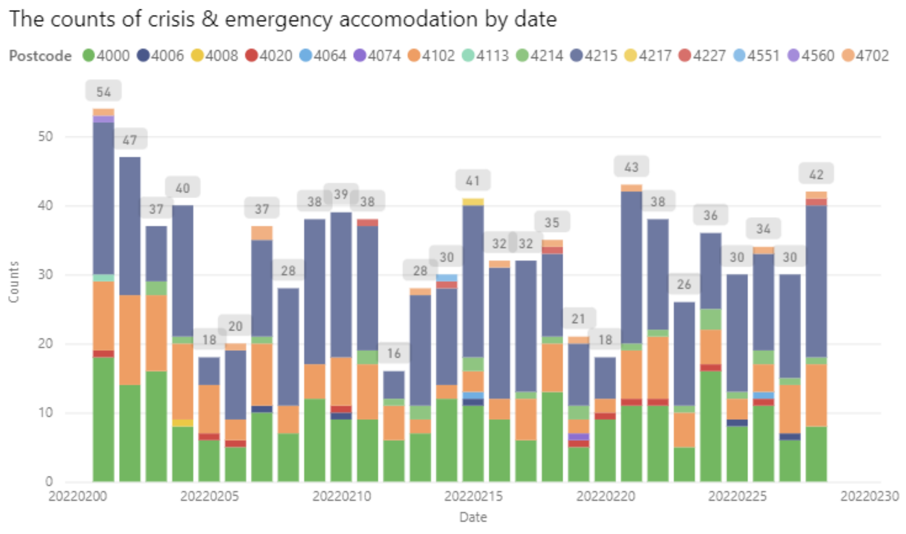


Figure 11: A histogram shows the demand of crisis & emergency accommodation in QLD by date, we do

see the demand increased slowly from 21st February. However, the trend is not significant.

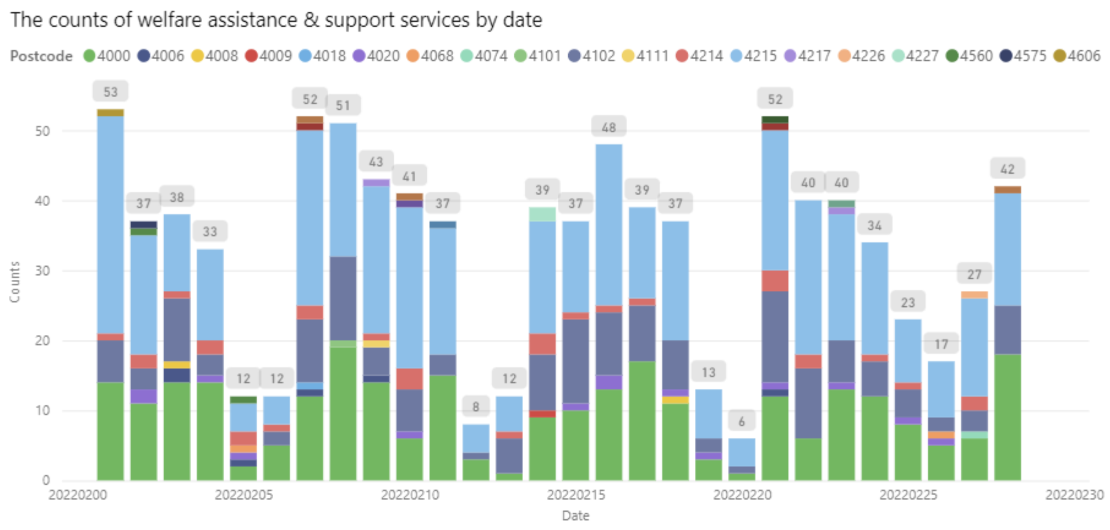


Figure 12: A histogram shows the demand of welfare assistance & support services in QLD by date, we do not see any special trends in here.

The first thing we noticed is the difference between weekdays and weekends. Generally, people seem to have less demand on weekend no matter which categories, the demand on Monday seems higher than average, this pattern suggest that we should compare the visualisations week by week. According to figure 10, we do not see any special pattern in terms of the demand for general accommodation services. However, from figure 11 and 12 we do see the demand for crisis & emergency accommodation and welfare assistance in the last week of February are slightly higher than before especially in the weekend, this could be a sign that some people were seeking for help on the My Community Directory website.

Reflection

This analysis is a precious experience of handling different kinds of limitations in the data analysis. I did my best try to retrieve the missing value, the method is definitely not perfect and still with a lot of limitations, but at least we can try to approach the core question one by one. I realised there are still a lot of limitation in the world of data analytics and machine learning, and this is the first time I encounter the short text clustering problem. Unfortunately, I could not complete the analysis in an advanced statistical manner, but it does leave a question mark of how to do the short text clustering in my mind. Honestly, the way David and Sean try to deal with the IP address and postcode's missing value is outstanding, I did not expect that API can be a breach point to this analysis, I am glad that I can learn with them.

I believe there are some hidden information in the No_Information category, sadly we need to ignore them for now due to time limitation, maybe we can see a stronger pattern in the histograms if we categorise more mixed value. The limitation of this analysis is obvious, we have a lot of missing value initially in the datasets, we tried to retrieve useful information with various method, for the postcode we can only track the location of the user’s nearest routers. For the WhatText we manually categorised them and had to ignore the less frequency terms due to time limitation. These factors have brought lots of uncertainty to this analysis, if anything changed during the process, it would have impact on the result very much. This kind of analysis pipeline is what we should be careful of.

References

[1] “My Coomunity Directory” Available: https://www.mycommunitydirectory.com.au/Resources/amc [Accessed: 6-Nov-2022].

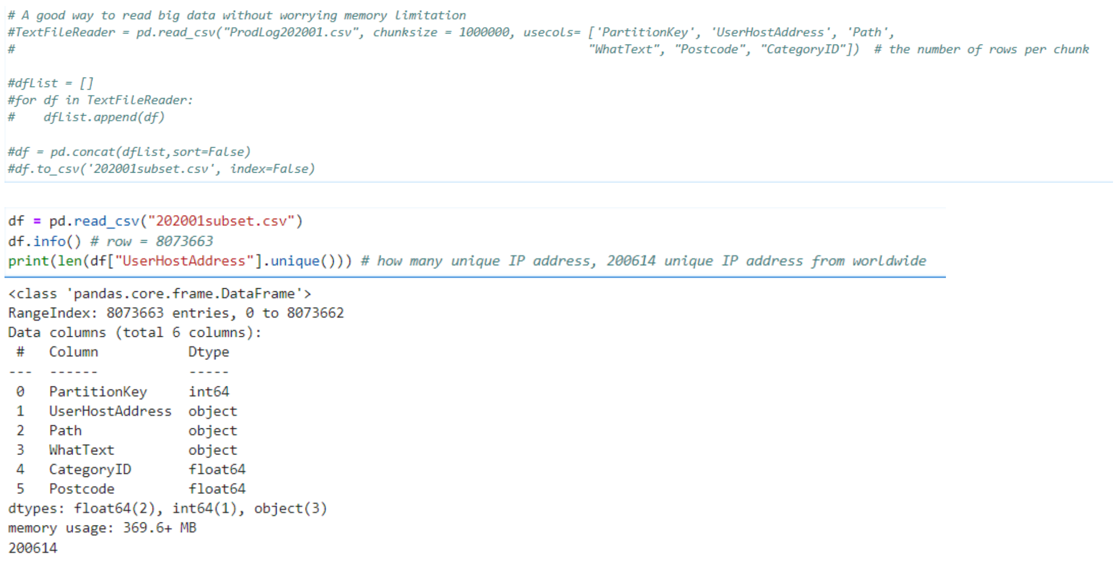
[2] “DBIP – IP geolocation API and database” Available: https://db-ip.com/ [Accessed: 6-Nov-2022].

[3] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent Dirichlet Allocation." Journal of machine Learning research 3. Jan (2003): 993-1022

[4] J. Yin and J. Wang, ‘A dirichlet multinomial mixture model-based approach for short text clustering’, in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York New York USA, Aug. 2014, pp. 233–242. doi: 10.1145/2623330.2623715.

[5] “Document Clustering with Python” Available: http://brandonrose.org/clustering [Accessed: 6-Nov-2022].

Appendix (python code)



```
# A good way to read big data without worrying memory limitation
#TextFileReader = pd.read_csv("ProdLog202001.csv", chunksize = 1000000, usecols= ['PartitionKey', 'UserHostAddress', 'Path',
#                                                                                 "WhatText", "Postcode", "CategoryID"])  # the number of rows per chunk

#dfList = []
#for df in TextFileReader:
#    dfList.append(df)

#df = pd.concat(dfList,sort=False)
#df.to_csv('202001subset.csv', index=False)
```

```
df = pd.read_csv("202001subset.csv")
df.info() # row = 8073663
print(len(df["UserHostAddress"].unique())) # how many unique IP address, 200614 unique IP address from worldwide
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8073663 entries, 0 to 8073662
Data columns (total 6 columns):
 #   Column           Dtype
---  ------           -----
 0   PartitionKey     int64
 1   UserHostAddress  object
 2   Path             object
 3   WhatText         object
 4   CategoryID       float64
 5   Postcode         float64
dtypes: float64(2), int64(1), object(3)
memory usage: 369.6+ MB
200614
```

```python
# drop UserHostAddress and Path duplicates so that each row now represents 1 demand.
df = df.drop_duplicates(subset=['UserHostAddress', 'Path'], keep='first')
print(len(df["UserHostAddress"].unique()))
print(df.shape)
df = df.drop_duplicates(subset=['UserHostAddress', 'CategoryID'], keep='first')
print(len(df["UserHostAddress"].unique()))
print(df.shape)
df = df.drop_duplicates(subset=['UserHostAddress', 'WhatText'], keep='first')
print(len(df["UserHostAddress"].unique()))
print(df.shape)

#df = df[df.WhatText != "-"]
#df = df[df.WhatText != " "]
df["WhatText"] = df["WhatText"].replace("-", np.nan)
df["WhatText"] = df["WhatText"].replace(" ", np.nan)
```

```
200614
(3994249, 6)
200614
(568964, 6)
200614
(202730, 6)
```

```python
# preprocessing for postcode
ip_info = pd.read_csv("ip_info_202001.csv", usecols= ['countryName', "zipCode", "isCrawler", "ipaddress"])
ip_info = ip_info.rename(columns = {'ipaddress': 'UserHostAddress'}, inplace = False)
df = pd.merge(df, ip_info, on='UserHostAddress', how='left')
df = df[df.UserHostAddress.notnull()]
print(df.shape)
df.head()
```

```
C:\Users\mib67\AppData\Local\Temp\ipykernel_9484\515865353.py:2: DtypeWarning: Columns (24) have mixed types. Specify dtype optio
=False.
  ip_info = pd.read_csv("ip_info_202001.csv", usecols= ['countryName', "zipCode", "isCrawler", "ipaddress"])
(202730, 9)
```

	PartitionKey	UserHostAddress	Path	WhatText	CategoryID	Postcode	countryName	zipCode	isCrawler
0	20200101	46.229.168.135	/ExploreMyCommunity	NaN	NaN	NaN	United States	20068	True
1	20200101	46.229.168.147	/Account	NaN	NaN	NaN	United States	20068	True
2	20200101	159.138.158.195	/Organisation/124352/Legacy_Laurel_Club_Southport	NaN	NaN	NaN	Hong Kong	NaN	False
3	20200101	66.249.65.133	/api/Search/Local	NaN	69.0	NaN	United States	94043	True
4	20200101	159.138.154.96	/New_South_Wales/Ku-Ring-Gai/Arts_Creatives	NaN	0.0	NaN	Hong Kong	NaN	False

```python
# Filter out the non-Australia IP and crawler IP
# We only want QLD postcode here
df = df[df.countryName == "Australia"]
df = df[df.isCrawler == False]
print(df.shape)
df["Postcode"] = df['Postcode'].fillna(df["zipCode"])
df = df[df.Postcode.notnull()]
df["Postcode_4XXX"] = df["Postcode"].str.startswith("4")
df = df[df.Postcode_4XXX == True]
df.drop("Postcode_4XXX", inplace=True, axis=1)
df.drop("zipCode", inplace=True, axis=1)
print(df.shape) # 60257 rows after filter out the countryName and isCrawler of IP, also filter out missing value in Postcode
df.head()
```

```
(150975, 9)
(60257, 8)
```

	PartitionKey	UserHostAddress	Path	Wh
170	20200101	120.22.5.65	/api/a/pagestatus	
182	20200101	2001:8004:13c1:ba59:2917:4cd9:5ca3:a5f1	/Western_Australia/Canning/Health_Services/General_Health_Services/19775/159217/Unicare_Health_-_Bentley	
255	20200101	2001:8003:1c10:ca00:3500:f473:207f:e078	/api/Locations/id/15384	
334	20200101	106.71.190.191	/Queensland/Brisbane/Christmas_Fireworks/71929	
448	20200101	122.148.129.117	/Queensland/Moreton_Bay/Welfare_Assistance___Services/General_Welfare___Support_Services/16048/134593/Caboolture_Community_Action	

```
# preprocessing for WhatText
category = pd.read_csv("CIE_category.csv")
category = category[["ID", "Name"]]
category = category.rename(columns = {'ID': 'CategoryID'}, inplace = False)
category = category.rename(columns = {'Name': 'CategoryName'}, inplace = False)
category['CategoryID'] = category['CategoryID'].astype("float64")
category = category.drop_duplicates(keep='first') # There are fucking duplicates in categories, are you fucking serious!?
df = pd.merge(df, category, on='CategoryID', how='left')
print(df.shape)
df.head()
```

```
(60257, 9)
```

	PartitionKey	UserHostAddress	Path	WhatT
0	20200101	120.22.5.65	/api/a/pagestatus	N
1	20200101	2001:8004:13c1:ba59:2917:4cd9:5ca3:a5f1	/Western_Australia/Canning/Health_Services/General_Health_Services/19775/159217/Unicare_Health_-_Bentley	N
2	20200101	2001:8003:1c10:ca00:3500:f473:207f:e078	/api/Locations/id/15384	N
3	20200101	106.71.190.191	/Queensland/Brisbane/Christmas_Fireworks/71929	N
4	20200101	122.148.129.117	/Queensland/Moreton_Bay/Welfare_Assistance__Services/General_Welfare__Support_Services/16048/134593/Caboolture_Community_Action	N

```
# get the category name for corresponding category ID, use category name to fill the missing value in WhatText
df["WhatText"] = df['WhatText'].fillna(df["CategoryName"])
df["Path_1"] = df["Path"].str.split("/").str[-1]
df["Path_2"] = df["Path"].str.split("/").str[-2]
df["WhatText"] = df['WhatText'].fillna(df["Path_1"])
df["WhatText"] = df['WhatText'].str.lower()
df["Path_2"] = df['Path_2'].str.lower()

df.to_csv('202001_preprocessed.csv', index=False)
```

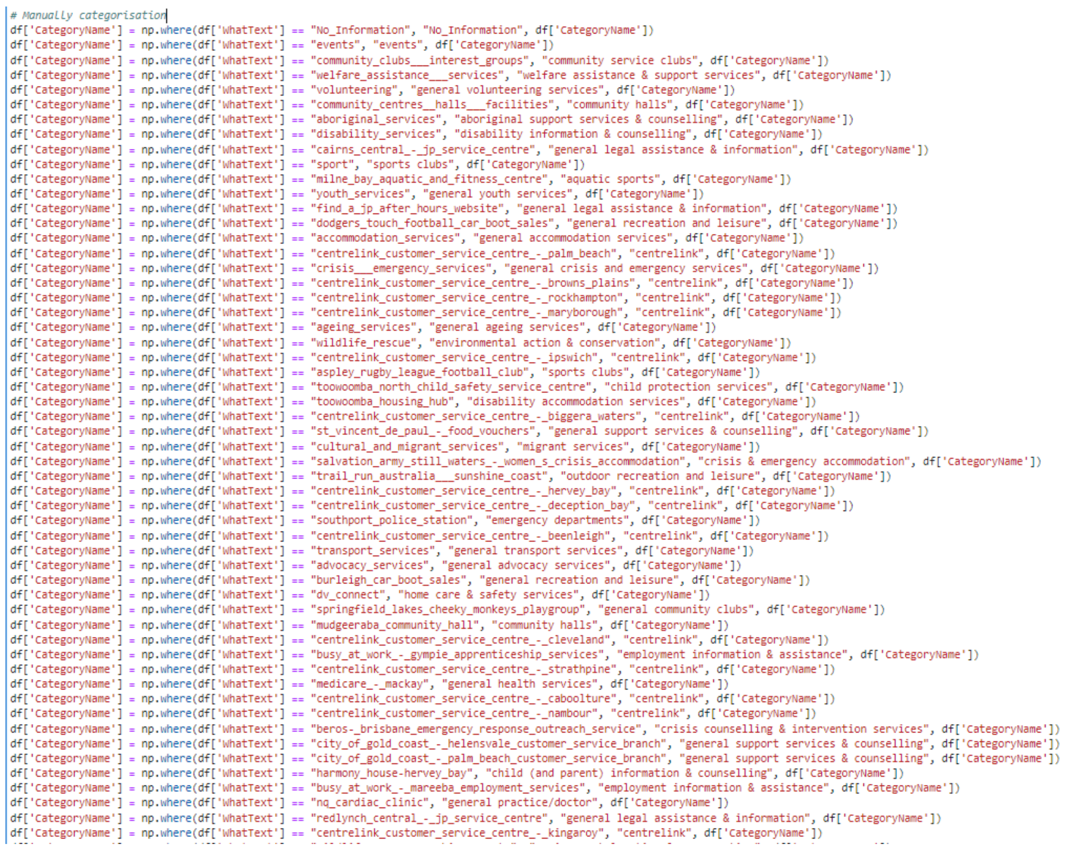
```
df = pd.read_csv("202001_preprocessed.csv")
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 60257 entries, 0 to 60256
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   PartitionKey     60257 non-null  int64  
 1   UserHostAddress  60257 non-null  object 
 2   Path             60257 non-null  object 
 3   WhatText         57440 non-null  object 
 4   CategoryID       39642 non-null  float64
 5   Postcode         60257 non-null  int64  
 6   countryName      60257 non-null  object 
 7   isCrawler        60257 non-null  bool   
 8   CategoryName     37303 non-null  object 
 9   Path_1           57372 non-null  object 
 10  Path_2           57614 non-null  object 
dtypes: bool(1), float64(1), int64(2), object(7)
memory usage: 4.7+ MB
```

```
# filter out values that related to system
df = df[df.WhatText != "pagestatus"]
df = df[df.WhatText != "api-info"]
df = df[df.WhatText != "getresults"]
df = df[df.WhatText != "terms_and_conditions"]
df = df[df.WhatText != "register"]
df = df[df.WhatText != "local"]
df = df[df.WhatText != "search"]
df = df[df.WhatText != "nearme"]
df = df[df.WhatText != "opendata"]
df = df[df.WhatText != "tags"]
df = df[df.WhatText != "organisations"]
df = df[df.WhatText != "getservicebadges"]
df = df[df.WhatText != "services"]
df = df[df.WhatText != "log"]
df = df[df.WhatText != "id"]
df = df[df.WhatText != "councils"]
df = df[df.WhatText != "council"]
df = df[df.WhatText != "getcouncil"]
df = df[df.WhatText != " "]
df = df[df.WhatText != "getappid"]
df = df[df.WhatText != "click"]
df = df[df.WhatText != "savedevicetoken"]
df = df[df.WhatText != "login"]
df = df[df.WhatText != "file"]
df = df[df.WhatText != "verifyemail"]
df = df[df.WhatText != "redirect"]
df = df[df.WhatText != "unsubscribe"]
df = df[df.WhatText != "account"]
df = df[df.WhatText != "bylocalityid"]
df = df[df.WhatText != "categories"]
df = df[df.WhatText != "category"]
df = df[df.WhatText != "directory"]
df = df[df.WhatText != "results"]
df = df[df.WhatText != "save"]
df = df[df.WhatText != "updatestatus"]
```

```
# replace all numeric search queries with No_Information value
df['WhatText'] = df['WhatText'].mask(pd.to_numeric(df['WhatText'], errors='coerce').notna())
df["WhatText"] = df['WhatText'].fillna(df["Path_2"])

df['WhatText'] = df['WhatText'].mask(pd.to_numeric(df['WhatText'], errors='coerce').notna())
df["WhatText"] = df['WhatText'].fillna("No_Information")

#second time filtering
df = df[df.WhatText != "id"]
df = df[df.WhatText != "councils"]
df = df[df.WhatText != "council"]
df = df[df.WhatText != "getcouncil"]
df = df[df.WhatText != " "]
df = df[df.WhatText != "log"]
df = df[df.WhatText != "getappid"]
df = df[df.WhatText != "click"]
df = df[df.WhatText != "savedevicetoken"]
df = df[df.WhatText != "login"]
df = df[df.WhatText != "file"]
df = df[df.WhatText != "verifyemail"]
df = df[df.WhatText != "redirect"]
df = df[df.WhatText != "unsubscribe"]
df = df[df.WhatText != "account"]
df = df[df.WhatText != "bylocalityid"]
df = df[df.WhatText != "categories"]
df = df[df.WhatText != "category"]
df = df[df.WhatText != "directory"]
df = df[df.WhatText != "results"]
df = df[df.WhatText != "save"]
df = df[df.WhatText != "updatestatus"]
df = df[df.WhatText != "contactus"]
df = df[df.WhatText != "meetings_"]
df = df[df.WhatText != "holidaynotice"]
df = df[df.WhatText != "location"]
df = df[df.WhatText != "memberfeatures"]
df = df[df.WhatText != "getrecentupdates"]
df = df[df.WhatText != "eventwidget"]
df = df[df.WhatText != "exploremycommunity"]
df = df[df.WhatText != "advanced"]

df = df.reset_index(drop = True)
df.drop("Path", inplace=True, axis=1)
df.drop("isCrawler", inplace=True, axis=1)
df.drop("Path_1", inplace=True, axis=1)
df.drop("Path_2", inplace=True, axis=1)
df.drop("UserHostAddress", inplace=True, axis=1)
df.drop("CategoryID", inplace=True, axis=1)
df.drop("countryName", inplace=True, axis=1)
```

```
# find out the unique_values that's not in the category dictionary, this is so hard...
name = category["CategoryName"].unique()
name = pd.DataFrame (name, columns = ['CategoryName'])
name = name.rename(columns = {'CategoryName': 'unique_values'}, inplace = False)
name["unique_values"] = name['unique_values'].str.lower()
value_counts = df["WhatText"].value_counts().rename_axis('unique_values').reset_index(name='counts')
df_diff = pd.concat([value_counts, name]).drop_duplicates("unique_values", keep=False)
```



```python
# Manually categorisation
df['CategoryName'] = np.where(df['WhatText'] == "No_Information", "No_Information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "events", "events", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "community_clubs___interest_groups", "community service clubs", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "welfare_assistance___services", "welfare assistance & support services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "volunteering", "general volunteering services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "community_centres__halls___facilities", "community halls", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "aboriginal_services", "aboriginal support services & counselling", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "disability_services", "disability information & counselling", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "cairns_central_-_jp_service_centre", "general legal assistance & information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "sport", "sports clubs", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "milne_bay_aquatic_and_fitness_centre", "aquatic sports", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "youth_services", "general youth services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "find_a_jp_after_hours_website", "general legal assistance & information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "dodgers_touch_football_car_boot_sales", "general recreation and leisure", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "accommodation_services", "general accommodation services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_palm_beach", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "crisis___emergency_services", "general crisis and emergency services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_browns_plains", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_rockhampton", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_maryborough", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "ageing_services", "general ageing services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "wildlife_rescue", "environmental action & conservation", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_ipswich", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "aspley_rugby_league_football_club", "sports clubs", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "toowoomba_north_child_safety_service_centre", "child protection services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "toowoomba_housing_hub", "disability accommodation services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_biggera_waters", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "st_vincent_de_paul_-_food_vouchers", "general support services & counselling", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "cultural_and_migrant_services", "migrant services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "salvation_army_still_waters_-_women_s_crisis_accommodation", "crisis & emergency accommodation", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "trail_run_australia___sunshine_coast", "outdoor recreation and leisure", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_hervey_bay", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_deception_bay", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "southport_police_station", "emergency departments", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_beenleigh", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "transport_services", "general transport services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "advocacy_services", "general advocacy services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "burleigh_car_boot_sales", "general recreation and leisure", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "dv_connect", "home care & safety services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "springfield_lakes_cheeky_monkeys_playgroup", "general community clubs", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "mudgeeraba_community_hall", "community halls", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_cleveland", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "busy_at_work_-_gympie_apprenticeship_services", "employment information & assistance", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_strathpine", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "medicare_-_mackay", "general health services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_caboolture", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_nambour", "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "beros-_brisbane_emergency_response_outreach_service", "crisis counselling & intervention services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "city_of_gold_coast_-_helensvale_customer_service_branch", "general support services & counselling", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "city_of_gold_coast_-_palm_beach_customer_service_branch", "general support services & counselling", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "harmony_house-hervey_bay", "child (and parent) information & counselling", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "busy_at_work_-_mareeba_employment_services", "employment information & assistance", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "nq_cardiac_clinic", "general practice/doctor", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "redlynch_central_-_jp_service_centre", "general legal assistance & information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'] == "centrelink_customer_service_centre_-_kingaroy", "centrelink", df['CategoryName'])
```

```
df['CategoryName'] = np.where(df['WhatText'].str.contains("jp"), "general legal assistance & information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("justice_of_the_peace"), "general legal assistance & information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("court"), "general legal assistance & information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("hospital"), "hospitals", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("police"), "emergency departments", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("mental_health"), "mental health services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("centrelink"), "centrelink", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("dental"), "dental & oral health", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("oral"), "dental & oral health", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("legal"), "general legal assistance & information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("wildlife"), "environmental action & conservation", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("busy_at_work"), "employment information & assistance", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("medicare"), "general health services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("aquatic"), "aquatic sports", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("aqua"), "aquatic sports", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("child_safety"), "child protection services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("migrant"), "migrant services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("car_boot_sales"), "general recreation and leisure", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("immigration"), "migrant services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("church"), "churches and places of worship", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("psychiatric"), "psychiatric services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("women_s_association"), "sexual support services & counselling", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("red_cross"), "general health services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("scout"), "children & youth clubs", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("job_access"), "employment information & assistance", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("football_club"), "sports clubs", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("salvation_army"), "crisis & emergency accommodation", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("hope_centre"), "churches and places of worship", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("homeless"), "general accommodation services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("cardiology"), "specialist outpatient clinics", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("baptist"), "churches and places of worship", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("community_centre"), "community & neighbourhood centres", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("richmond_fellowship"), "mental health services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("rfq"), "mental health services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("trustee"), "general legal assistance & information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("act_for_kids"), "child (and parent) information & counselling", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("council"), "general communication & information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("university_of_the_third_age"), "seniors clubs & social groups", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("u3a"), "seniors clubs & social groups", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("first_aid"), "ses, rural fire, lifesavers, search & rescue", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("playgroup"), "play groups & childcare", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("employment"), "employment information & assistance", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("rehabilitation"), "health rehabilitation services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("ymca"), "play groups & childcare", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("housing_service"), "general accommodation services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("construction"), "industry bodies", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("screen"), "health screening services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("market"), "food vans and kitchens", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("doctor"), "general practice/doctor", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("carer"), "carer information", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("libraries"), "general community facilities", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("hall"), "community halls", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("home_assist"), "home care & safety services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("girl_guides"), "general youth services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("youth_justice"), "child protection services", df['CategoryName'])
df['CategoryName'] = np.where(df['WhatText'].str.contains("state_school"), "state (primary and high schools)", df['CategoryName'])
```

```
df["CategoryName"] = df['CategoryName'].fillna("No_Information")
df["CategoryName"] = df['CategoryName'].str.lower()
df.head()
```

	PartitionKey	WhatText	Postcode	CategoryName
0	20200101	general health services	4215	general health services
1	20200101	christmas_fireworks	4020	no_information
2	20200101	general welfare & support services	4215	general welfare & support services
3	20200101	No_Information	4000	no_information
4	20200101	bunbury_acute_psychiatric_unit	4215	psychiatric services

```
df.to_csv('202001_categorised.csv', index=False)
```

```
# Monthly count
df = pd.read_csv("202001_categorised.csv")
df.drop("PartitionKey", inplace=True, axis=1)
df['Postcode'] = df['Postcode'].astype("int")
df["CategoryName"] = df['CategoryName'].str.lower()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55213 entries, 0 to 55212
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   WhatText      55213 non-null  object
 1   Postcode      55213 non-null  int32 
 2   CategoryName  55213 non-null  object
dtypes: int32(1), object(2)
memory usage: 1.1+ MB
```

```
# remove punctuation
import string
def remove_punctuation(text):
    punctuationfree = "".join([i for i in text if i not in string.punctuation])
    return punctuationfree
#storing the puntuation free text

#df['WhatText'] = df['WhatText'].str.replace("/", " ")
#df['WhatText'] = df['WhatText'].str.replace("_", " ")
df['CategoryName'] = df['CategoryName'].apply(lambda x:remove_punctuation(x))
df["CategoryName"] = df["CategoryName"].str.replace(" ", "_")
df.head()
```

	WhatText	Postcode	CategoryName
0	general health services	4215	general_health_services
1	christmas_fireworks	4020	noinformation
2	general welfare & support services	4215	general_welfare__support_services
3	No_Information	4000	noinformation
4	bunbury_acute_psychiatric_unit	4215	psychiatric_services

```
# word tokenization and lower case
import nltk
from nltk.tokenize import word_tokenize
df["CategoryName"] = df['CategoryName'].str.lower()
df['CategoryName'] = df.apply(lambda row: nltk.word_tokenize(row['CategoryName']), axis=1)
df.head()
```

	WhatText	Postcode	CategoryName
0	general health services	4215	[general_health_services]
1	christmas_fireworks	4020	[noinformation]
2	general welfare & support services	4215	[general_welfare__support_services]
3	No_Information	4000	[noinformation]
4	bunbury_acute_psychiatric_unit	4215	[psychiatric_services]

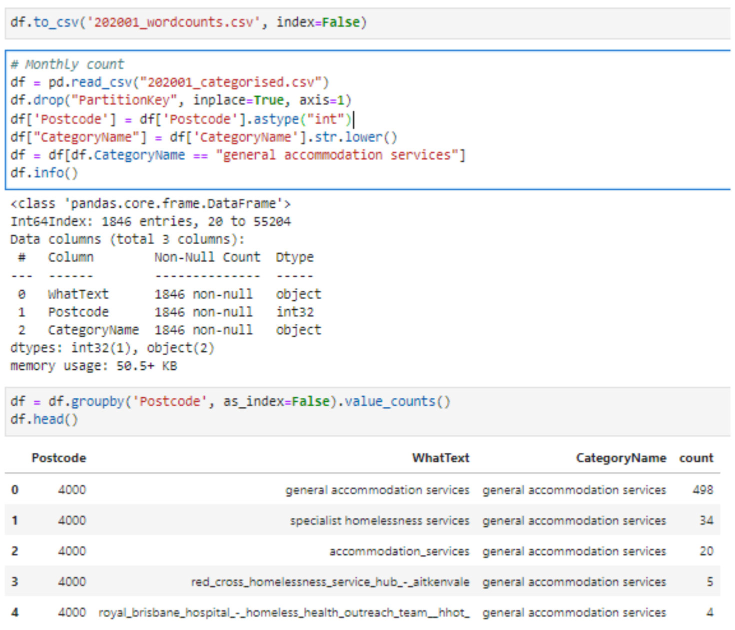
```
df = df.groupby('Postcode', as_index=False).sum()
```

```
from collections import Counter

word_count = []
for line in df["CategoryName"]:
    counts = Counter(line)
    #counts = sorted(counts.items(), key=lambda pair: pair[1], reverse=True)
    counts = counts.most_common(20)
    word_count.append(counts)

df["counts"] = word_count

df.drop("WhatText", inplace=True, axis=1)
df.drop("CategoryName", inplace=True, axis=1)
df["Country"] = "Australia"
```

```
df.head()
```

	Postcode	counts	Country
0	4000	[(noinformation, 2713), (general_legal_assistance__information, 675), (general_accommodation_services, 592), (play_groups__childcare, 564), (general_health_services, 543), (welfare_assistance__support_services, 527), (general_practicedoctor, 520), (child_protection_services, 490), (employment_information__assistance, 458), (sports_clubs, 455), (state_primary_and_high_schools, 450), (general_support_services__counselling, 436), (crisis__emergency_accommodation, 413), (seniors_clubs__social_groups, 389), (general_disability_services, 376), (general_community_clubs, 365), (churches_and_places_of_worship, 328), (mental_health_services, 326), (general_welfare__support_services, 325), (community_service_clubs, 320)]	Australia
1	4001	[(noinformation, 12), (events, 5), (food_vans_and_kitchens, 2), (crisis__emergency_accommodation, 2), (community_information__referral_services, 2), (state_primary_and_high_schools, 1), (environment__conservation_local_history, 1), (general_communication__information, 1), (welfare_assistance__support_services, 1), (children__youth_clubs, 1), (community__neighbourhood_centres, 1), (seniors_clubs__social_groups, 1), (general_legal_assistance__information, 1), (general_ageing_services, 1), (mental_health_services, 1), (general_youth_services, 1), (centrelink, 1), (fitness_activities, 1), (outdoor_sports, 1), (general_welfare__support_services, 1)]	Australia
2	4004	[(noinformation, 1)]	Australia
3	4006	[(noinformation, 31), (events, 16), (general_health_services, 13), (general_accommodation_services, 8), (general_practicedoctor, 8), (general_legal_assistance__information, 7), (crisis__emergency_accommodation, 7), (drug__alcohol_services, 7), (child_protection_services, 7), (general_support_services__counselling, 7), (community_halls, 6), (general_disability_services, 6), (general_communication__information, 5), (play_groups__childcare, 5), (state_primary_and_high_schools, 5), (community_service_clubs, 5), (centrelink, 5), (general_ageing_services, 4), (welfare_assistance__support_services, 4), (allied_health, 4)]	Australia
4	4008	[(noinformation, 6), (events, 3), (community_halls, 2), (general_disability_services, 2), (mental_health_services, 2), (general_accommodation_services, 1), (local_advocacy_services, 1), (specialist_outpatient_clinics, 1), (outdoor_recreation_and_leisure, 1), (general_practicedoctor, 1), (general_community_clubs, 1), (general_support_services__counselling, 1), (general_health_services, 1), (food_vans_and_kitchens, 1), (pharmacies, 1), (general_child_services, 1), (general_crisis_and_emergency_services, 1), (disability_respite_and_activity_centres, 1), (general_advocacy_services, 1), (employment_information__assistance, 1)]	Australia

```
df.to_csv('202001_wordcounts.csv', index=False)
```



```
df.to_csv('202001_wordcounts.csv', index=False)
```

```
# Monthly count
df = pd.read_csv("202001_categorised.csv")
df.drop("PartitionKey", inplace=True, axis=1)
df['Postcode'] = df['Postcode'].astype("int")
df["CategoryName"] = df['CategoryName'].str.lower()
df = df[df.CategoryName == "general accommodation services"]
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1846 entries, 20 to 55204
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   WhatText      1846 non-null   object
 1   Postcode      1846 non-null   int32 
 2   CategoryName  1846 non-null   object
dtypes: int32(1), object(2)
memory usage: 50.5+ KB
```

```
df = df.groupby('Postcode', as_index=False).value_counts()
df.head()
```

	Postcode	WhatText	CategoryName	count
0	4000	general accommodation services	general accommodation services	498
1	4000	specialist homelessness services	general accommodation services	34
2	4000	accommodation_services	general accommodation services	20
3	4000	red_cross_homelessness_service_hub_-_aitkenvale	general accommodation services	5
4	4000	royal_brisbane_hospital_-_homeless_health_outreach_team__hhot_	general accommodation services	4