Documentation

Ruoshui Chen, Xuhong Ye

1. Dataset Description

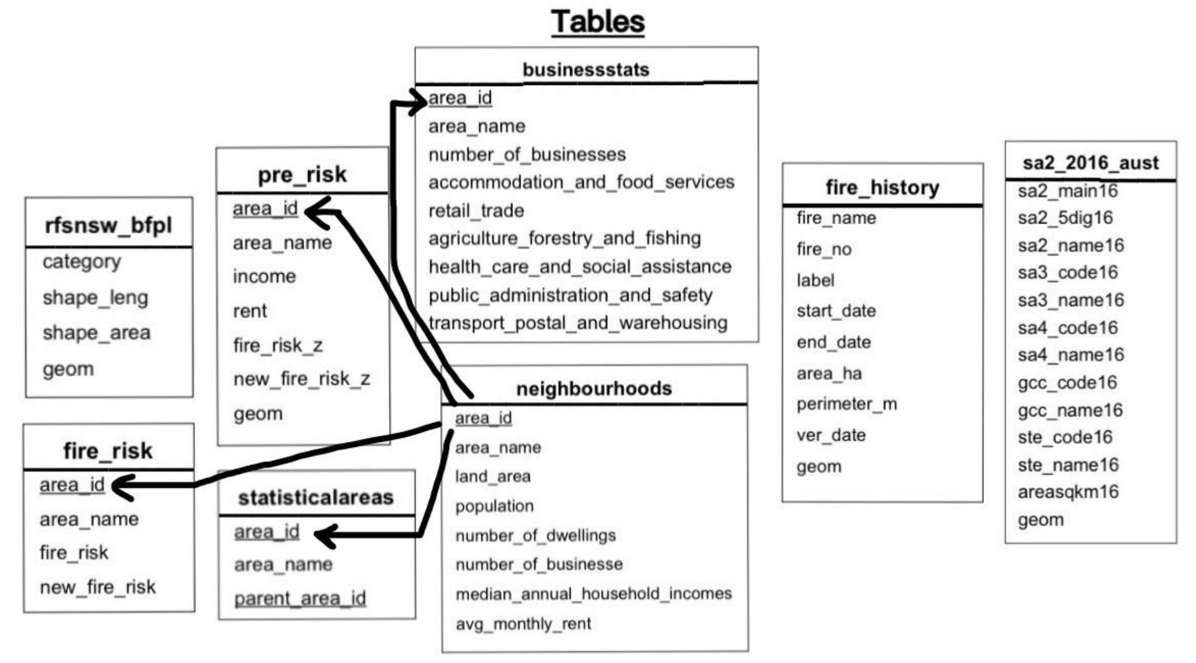
We obtained several CSV files with Statistical Area 2 (SA2) data from the Australian Bureau of Statistics (ABS), as well as some bush fire prone land vegetation spatial data from the NSW Rural Fire Service, which are all provided in canvas, and one own extra data: NPWS Fire History data from web NSW Government. Their file names are StatisticalAreas.csv, Neighbourhoods.csv, BusinessStats.csv (the BusinessStats data was acquired by the ABS via the ATO in 2018, and the ABS data was primarily acquired from the 2016 census), RFSNSW_BFPL.shp (The NSW RFS BFPL data was acquired and compiled by the NSW RFS in Nov 2020 and updated last in April 2021, original dataset - https://portal.spatial.nsw.gov.au/portal/home/item.html?id=3de03ae1965840cfa5dcd9e4018745a7)

, SA2_2016_AUST.shp and FIre_NPWSFireHistory_13052021.shp (from https://datasets.seed.nsw.gov.au/dataset/fire-history-wildfires-and-prescribed-burns-1e8b6)

To pre-process the data, we created some tables using these files and we cleaned our data using CASE statements, in cases like missing values: we turn some null or empty values and placeholders into 0; inconsistent values: we change some numbers as text into pure numbers.

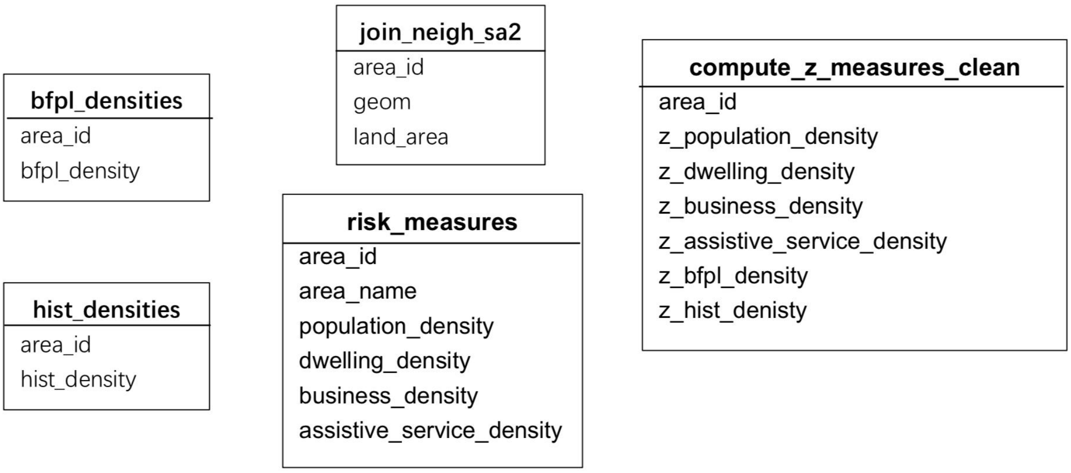
2. Database Description

We integrated our data in the public schema of user rche0014. The diagrams of our data base are shown below.



Except for the tables originally given from canvas, we also have an extra data set called *fire_history*. The sum of z values and new z values (include extra, the hist_density) corresponding to each neighborhood are saved in the column fire_risk_z and new_fire_risk_z of the table *pre_risk*. The table *fire_risk* saves the final fire risk scores and the values calculated with an extra data set called new_fire_risk of each neighborhood.

Materialized Views



Those materialized views above are used to save some intermediate values for calculating fire risk scores.

By joining *neighborhoods* and *sa2_2016_aust* as shown below, we gained *join_neigh_sa2* that have polygons of all neighborhoods which will be helpful to calculate bfpl_density and hist_density by geometry joins.

```
query = """create MATERIALIZED view join_neigh_sa2 as
            (select n.area_id, w.geom, n.land_area from sa2_2016_aust w   join
neighbourhoods n on(n.area_id = w.sa2_main16))"""
conn.execute("drop MATERIALIZED view if exists join_neigh_sa2 CASCADE")
conn.execute(query)
```

Indexes

neighbourhoods: neigh_area_id_idx (area_id), land_area_idx (land_area)

rfsnsw_bfpl: bfpl_geom_idx (geom), shape_area_idx (shape_area)

fire_history: fire_hist_geom_idx (geom)

bfpl_densities: bfpl_density_idx (bfpl_density)

risk_measures: assistive_service_density_idx (assistive_service_density)
dwelling_density_idx (dwelling_density), population_density_idx (population_density),
business_density_idx (business_density),

sa2_2016_aust: sa2_geom_idx (geom) , sa2_area_id_idx (sa2_main16)

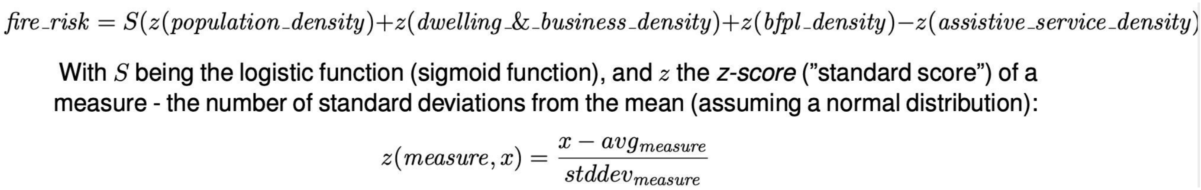
join_neigh_sa2: geom_neigh_sa2_idx (area_id), land_area_neigh_sa2_idx (land_area)

To speed up calculations and tables joining, we created indexes as above.

Specifically, indexes such as `land_area_idx`, `shape_area_idx`, `bfpl_density_idx`, `population_density_idx` and `land_area_neigh_sa2_idx` are used to improve the speed for calculating 6 densities and their z values. It is very decisive to create indexes on the land_area columns of materialized view *join_neigh_sa2* and table *neighbourhoods* since almost all the density calculations need to divide land_area. Indexes such as `neigh_area_id_idx`,and `bfpl_geom_idx` are used to speed up joining especially for joining the table *neighborhoods* with *sa2_2016_aust* (around 2000 rows) and geometry joinings (polygons of neighborhoods which saved in *join_neigh_sa2* and extra (column geom in table *fire_history*) and over 516 thousand points values in *rfsnsw_bfpl*).

3. Fire Risk Scores Analysis

The formula we applied to compute the Fire Risk score per neighbourhood is shown below:

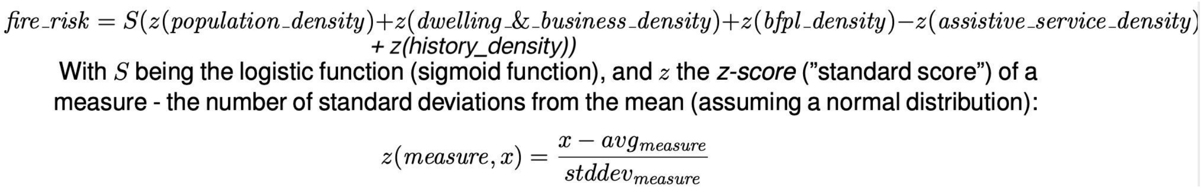


$$fire\_risk = S(z(population\_density)+z(dwelling\_\&\_business\_density)+z(bfpl\_density)-z(assistive\_service\_density)$$

With S being the logistic function (sigmoid function), and z the *z-score* (”standard score”) of a measure - the number of standard deviations from the mean (assuming a normal distribution):

$$z(measure, x) = \frac{x - avg_{measure}}{stddev_{measure}}$$

Besides, if we add our extra factor Fire history from extra data, the formula will be shown below since the fire history and fire risk may maintain a positive correlation to some degree:



$$fire\_risk = S(z(population\_density)+z(dwelling\_\&\_business\_density)+z(bfpl\_density)-z(assistive\_service\_density)$$

+ z(history_density))

With S being the logistic function (sigmoid function), and z the *z-score* (”standard score”) of a measure - the number of standard deviations from the mean (assuming a normal distribution):

$$z(measure, x) = \frac{x - avg_{measure}}{stddev_{measure}}$$

Overall, we use SQL queries to calculate 6 relative densities and their z values of neighbourhoods and save sum of the 6 z scores as the final z scores in the pre_risk table, and then calculate fire risk by python. Finally, we upload the final fire risks to the table fire_risk of the database.

a. Calculate dwelling, population, business and assistive service densities as below.

```
create MATERIALIZED view risk_measures as (
SELECT n.area_id, n.area_name,
  CASE WHEN n.population LIKE '%%,%%' THEN (CAST(REPLACE(n.population,',','') as int)/n.land_area)
   WHEN n.population = '' THEN NULL
   ELSE (cast(n.population as int)/n.land_area)
   END as population_density,
   CASE WHEN n.number_of_dwellings LIKE '%%,%%' THEN   (CAST(REPLACE(n.number_of_dwellings,',','')
as int)/n.land_area)
   ELSE (cast(n.number_of_dwellings as int)/n.land_area)
   END as dwelling_density,
   (b.number_of_businesses/n.land_area) as business_density,
   ((b.accommodation_and_food_services + b.retail_trade +
   b.agriculture_forestry_and_fishing + b.health_care_and_social_assistance +
b.public_administration_and_safety + b.transport_postal_and_warehousing)/n.land_area) as
assistive_service_density
    FROM neighbourhoods n join businessstats b on(n.area_id = b.area_id))
```

b. calculate bfpl density and hist density of each neighbourhood by geometry joins with neighbourhoods' polygon (e.g. bfpl_density)

```
create MATERIALIZED view bfpl_densities as (select sub.area_id,
SUM(c.shape_area)/sub.land_area as bfpl_density from rfsnsw_bfpl c inner join
join_neigh_sa2 sub on(st_Intersects(sub.geom, c.geom)) group by sub.land_area,
sub.area_id)
```

c. Calculate average and the standard deviation of densities respectively using the AVG( ) and STDDEV( ) statement and save the relative queries in the view compute_avg_std_measures.

d. Calculate z scores of 6 densities

```
create view compute_z_measures as (
         SELECT n.area_id, ((n.population_density - sub.avg_population_density)/sub.stddev_population_density) as
z_population_density,
         ((n.dwelling_density - sub.avg_dwelling_density)/sub.stddev_dwelling_density) as z_dwelling_density,
         ((n.business_density - sub.avg_business_density)/sub.stddev_business_density) as z_business_density,
         ((n.assistive_service_density - sub.avg_assistive_service_density)/sub.stddev_assistive_service_density)
as z_assistive_service_density,
         ((b.bfpl_density - sub.avg_bfpl_density)/sub.stddev_bfpl_density) as z_bfpl_density,
         ((b.bfpl_density - sub.avg_bfpl_density)/sub.stddev_hist_density) as z_hist_density
         FROM (risk_measures n left outer join bfpl_densities b using(area_id)) left outer join hist_densities h
using(area_id),
         compute_avg_std_measures sub)
```

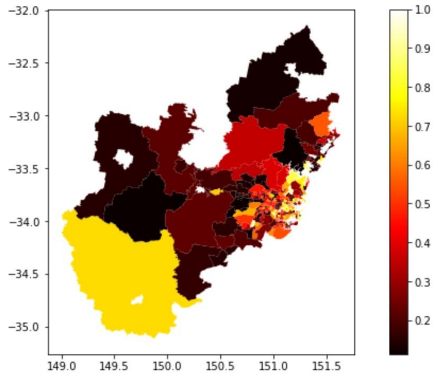
e. Calculate the sum of z scores by the function described top above and up load values to the table pre_risk.

f. Calculate the final fire risk scores by python. (sigmoid function: $f(x) = \frac{1}{1+e^{-x}}$).

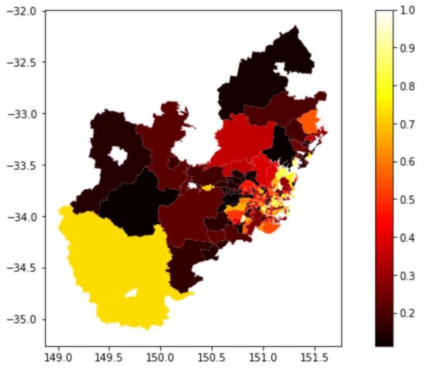
```
SQL_Query = pd.read_sql_query('''select * from pre_risk''', conn)
data_pre_risks = pd.DataFrame(SQL_Query, columns=['area_id','area_name', 'income',
'rent','fire_risk_z', 'new_fire_risk_z', 'geom'])

def sigmoid(x):
  return round(1 / round((1 + math.exp(round(-x, 5))),5),3)

risk_with_income_rent = data_pre_risks.copy()
risk_with_income_rent['fire_risk'] = [sigmoid(item) for item in
                                       data_pre_risks['fire_risk_z']]
```

According to two versions of fire risk scores and the map visualizations we obtained, we can see that they are almost the same. In each version, most scores are below 0.3, while ¼ of scores are higher than even 0.7. By joining geoms and scores, we can make two graphical maps shown below.

(Map1: Fire Risk Without Extra Data)



(Map2: Fire Risk With Extra Data: Fire History)

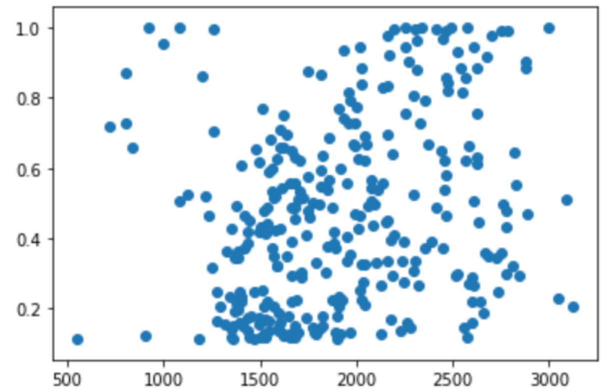


4. Correlation Analysis

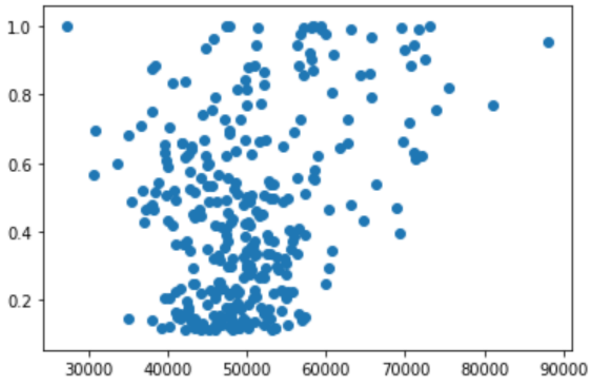
There is almost no correlation between fire risk score and both median income and average rent.

After calculating fire risk scores and saving the median income and average rent to python data frame ***risk_with_income_rent***, we use it and other python packages such as numpy and matplotlib.pyplot to draw scatter plots and calculate correlation coefficient of scores (included new one) and income and rent respectively.

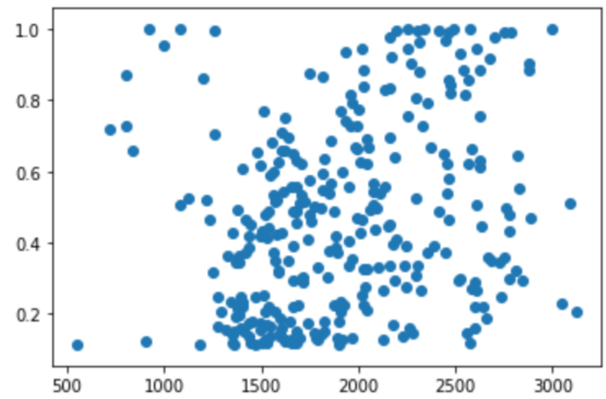
```
fire_risk: mean = 0.453, stdv = 0.267
rent: mean = 1921.542, stdv = 489.515
```



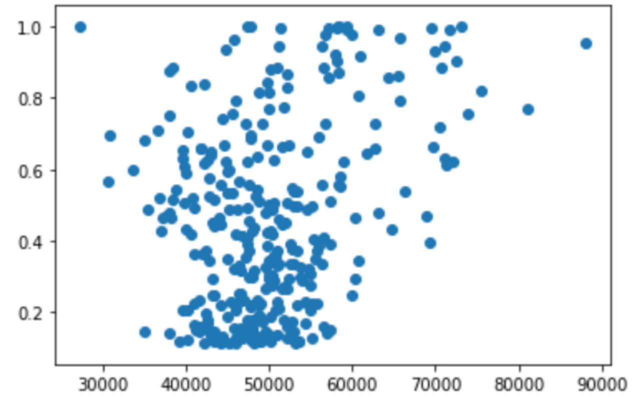
```
fire_risk: mean = 0.453, stdv = 0.267
income: mean = 50265.952, stdv = 8652.618
```



```
new_fire_risk: mean = 0.453, stdv = 0.267
rent: mean = 1921.542, stdv = 489.515
```

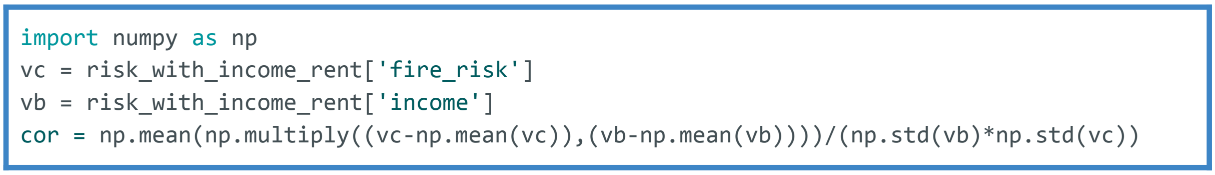


```
new_fire_risk: mean = 0.453, stdv = 0.267
income: mean = 50265.952, stdv = 8652.618
```



Although the plots above with x axis as income or rent and y axis as old and new scores have some slight differences, they are overview similar. It's hard to find any direct trend line between scores and income and rent.

To be more rigorous, we also calculate the correlation coefficient. (e.g. code for calculating fire risk and median income)



```
import numpy as np
vc = risk_with_income_rent['fire_risk']
vb = risk_with_income_rent['income']
cor = np.mean(np.multiply((vc-np.mean(vc)),(vb-np.mean(vb))))/(np.std(vb)*np.std(vc))
```

As the results of fire risk scores (with/without extra data set) and income and rent are all around 0.31 and 0.25 respectively, which is much smaller than 1. So we draw the conclusion that the fire risk scores have practically no correlation with median income and average rent.