APPLIED STATISTICS & DATA MINING

A report on the risk of Cancer due to air toxicity in the USA and the customer sentiment on the top 30 burger restaurants in Thailand.

Ambareesh Jonnavittula

@00600894

Table of Contents

1. Classification – High risk vs. Low risk of Cancer (labelled value)

1.1 Abstract

This module of the project aims to support and discover the insights from the research excerpts of National Air Toxicity Assessment (NATA) executed by the US Environmental Protection Agency (EPA) in 2011 and 2014. Using R and SAS Enterprise Miner tools, we predict the total cancer risk per a million people in a tract of a county within a state in all USA basing on the other possible causes of cancer risks. To proceed with the Data mining process, we could use any of the Data mining algorithms like SEMMA or CRISP-DM. We used CRISP-DM methodology to perform Data mining for the classification task. An R markdown file has been used to prepare the classification – decision tree model with the slidy presentation rendering an HTML output. The business or operational understanding and data understanding is performed as part of the requirements gathering. Later we import the dataset and clean it as part of data preparation. To optimize the functionality and reduce the redundancy, we use R functions enabling to break down or decompose a problem into smaller chunks. In addition, the code can be reproducible and reusable, and it was prepared in a systematic, organised, robust and an efficient manner. 'rpart' and 'rattle' libraries were used to perform this data mining task. We first train the model and discover the fitting, and if necessary, we perform tuning operation to improve the results. To identify the goodness of fit within classification, we utilise the confusion matrix, accuracy rate, error rate, control variable, precision, recall, and f1 score. Plots derived from ggplot2 were used wherever necessary to showcase the quality and aesthetics of the graphs. We later utilise SAS Enterprise Miner as a secondary data mining tool where we produce process flow diagrams and parameterize the tasks and compare the results with R.

1.2 Introduction

In 2011 and 2014, the US government agency of the United States Environmental Protection Agency (EPA) assessed the national air toxicity and released a dataset to the public, and this study is titled as the National Air Toxicity Assessment (NATA). EPA developed NATA as a screening tool for state, local and tribal air agencies. NATA's results help these agencies identify which pollutants, emission sources and places they may wish to study further to better understand any possible risks to public health from air toxics. There is now enough evidence that pollutants like acetaldehyde, benzene, cyanide, particulate matter components of diesel engine emissions (namely, diesel PM), toluene, and 1,3-butadiene have been proved to be the root cause for cancer across a wide scale of patients.

Air quality specialists use NATA results to learn which air toxics and emission source types may raise health risks in certain places. They can then study these places in more detail, focusing where the risks to people may be highest. NATA uses a 4-step methodology to develop the assessment:

1. Compile a national emissions inventory of outdoor air toxics sources.
2. Estimate ambient concentrations of air toxics across the United States.
3. Estimate population exposures across the United States.
4. Determine potential public health risks from breathing air toxics.

In this task, we will classify the level of cancer based on the assessment and create a decision tree model to classify and predict the amount of risk of cancer across the USA.

1.2.1 Brief background of the task

The total cancer risk per million of a given chemical from all the source types is the 'response variable' which is going to be used to classify the amount of the threat for a million people at a time.

Below is the table showing the predictor variables along with their descriptions. Kindly note that all the variables are essentially the average risk of cancer per million due to various causes.

S.No	Variable Name	Type	Brief Description
1	Total_crpm	**Response**	Total average cancer risk of a given chemical from all source types
2	railyards_crpm	*Predictor*	Average cancer risk from point sources and railyards
3	airport_crpm	*Predictor*	Average cancer risk from airport sources
4	rwc_crpm	*Predictor*	Average cancer risk from residential wood combustion sources
5	cmv_crpm	*Predictor*	Average cancer risk from commercial marine vessels
6	biogenics_crpm	*Predictor*	Average cancer risk from biogenic sources
7	fires_crpm	*Predictor*	Average cancer risk from fires
8	secondary_crpm	*Predictor*	Average cancer risk due to secondary formation, which is a process by which chemicals are transformed in the air into other chemicals
9	np_10m_releaseheight_crpm	*Predictor*	Average cancer risk from non-point sources with 10m release height
10	np_low_releaseheight_crpm	*Predictor*	Average cancer risk from non-point sources with low release height
11	cmv_loco_crpm	*Predictor*	Average cancer risk from non-road sources (e.g., airplanes, trains, lawn mowers, construction vehicles, farm machinery).
12	lightduty_crpm	*Predictor*	Average cancer risk from on-road light duty mobile sources
13	heavyduty_crpm	*Predictor*	Average cancer risk from on-road heavy duty mobile sources

1.2.2 Formulation of the research question

To objectify the task, the total cancer risk per million variable is split into various levels of threat depending on the study and the variable range. We are going to classify the data into one of these groups and predict the outcomes with these levels.

Level of Threat	Significance (based on Total Cancer risk per million)
L0	Less than 0.5
L1	Less than or equals to 5
L2	Less than or equals to 10
L3	Greater than 10

1.2.3 Justification: Why did I choose this topic/dataset?

According to the American Association for Cancer Research, new study suggests that air pollution is also associated with increased risk of mortality for several other types of cancer, including breast, liver, and pancreatic cancer.

Following heart disease, cancer is the second leading cause of death in the United States and around the world. In 2018, an estimated 9.5 million people died of cancer worldwide. That's about 26,000 people each day and 1 out of every 6 deaths. About 600,000 cancer deaths happen in the U.S. each year and about 80,000 in Canada. The rest happen in countries all around the world. About 7 out of every 10 deaths from the disease happen in low- or middle-income countries.

Cancers develop when something goes wrong in the DNA of a cell. Studying the DNA of people who develop cancer, and of those who don't, can be key in identifying people with a particularly high risk. It also helps in the search for new drugs and in choosing the best treatments for patients.

1.3 Aim and Objective of the task

Over the years, there have been various carcinogenic factors which have caused Cancer among the people across all the states of USA according to NATA.

The objective of this task is to be able to prepare a classification model to utilize the predictor variables that can identify and/or predict the factors which potentially pose a range of risk of cancer among the people of USA.

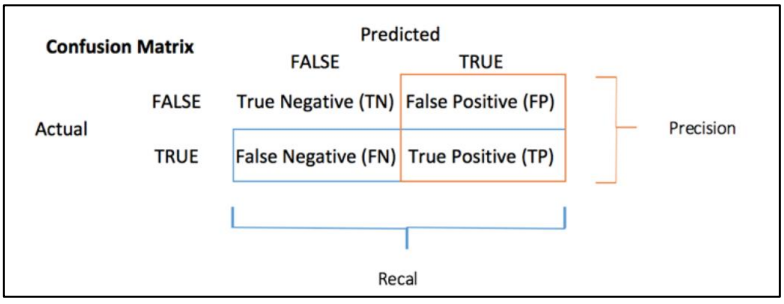
1.4 Brief Literature Review

Classification is essentially a supervised machine learning model and a data mining technique to predict group membership for data instances (Fernandez et al 2010). Popular classification techniques include decision trees and neural networks (Rajen & Gopal 2006). Classification tree (also known as decision tree) methods were a good choice when the data mining task is classification or prediction of outcomes. Classification tree labels records and assigns them to discrete classes. A classification tree may also provide the measure of confidence that the classification is correct.

Classification tree is built through a process known as binary recursive partitioning. This is an iterative process of splitting the data into partitions and then splitting it up further on each of the branches. Classification is efficient in decision-making, however, a problem that occurs in classification is when a person or an entity must be put into a class based on the predefined properties of the person or entity. Traditional methods available in statistical for classification including discriminate analysis were used in the past for decision making under uncertainty using Bayes theorem (Kazienko & Kajdanowicz2010). In this work, a new probability-based model has been proposed by the authors to compute the posterior probability which is used for effective classification. Neural networks provide several advantages in the effective classification of data. First, neural networks-based classification techniques are flexible and based on weight assignment. Second, they use function approximations with respect to the activation functions. Third, they are useful to form rules for learning and decision making. Finally, training and testing for neural networks are easy to implement.

The confusion matrix is a better choice to evaluate the classification performance. The general idea is to count the number of times True instances are classified are False. Accuracy and Error rates are calculated based on the following aspects. Let us imagine we are trying to create a predictive model to prove that there is diabetes within a patient.

1. **true positives (TP):** These are cases in which we predicted yes, and they do have the diabetes.
2. **true negatives (TN):** We predicted no, and they don't have the diabetes.
3. **false positives (FP):** We predicted yes, but they don't have the diabetes. (Also known as a "Type I error.")
4. **false negatives (FN):** We predicted no, but they do have the diabetes. (Also known as a "Type II error.")



Precision is defined as the number of relevant diabetes patients identified divided by the total number of patients identified. Recall is defined as the number of relevant diabetic patients identified divided by the total number of diabetic patients in existence. ROC shows a relation between the sensitivity and the specificity of the algorithm. F1-score is a composite measure which benefits algorithms with higher sensitivity and challenges algorithms with higher specificity.

We will proceed as follow: Tune the hyper-parameters

- Construct function to return accuracy
- Tune the maximum depth

- Tune the minimum number of samples a node must have before it can split
- Tune the minimum number of samples a leaf node must have

Rpart() function uses the Gini impurity measure to split the node. The higher the Gini coefficient, the more different instances within the node.

1.5 Explanation and preparation of datasets

1.5.1 Description of the dataset

- **Data Source** – Link
- **Data File (in .xlsx format)** – Link (Size – 193 MB)

- The dataset contains various exposure concentrations, hazard indices, average cancer risk per million, and the population – all grouped by state, EPA region, county, tract, FIPS, and the pollutant. To deal with performance issues with R, we will take only limited columns for this task.

- **Categorical Variables** –
    - State: State in the United States
    - EPA Region: EPA has ten regional offices, each of which is responsible for the execution of its programs within several states and territories.
    - County: A county is a political and administrative division of a state, providing certain local governmental services.
    - FIPS: FIPS (Federal Information Processing Standards) are a set of standards that describe document processing, encryption algorithms and other information technology standards for use within non-military government agencies and by government contractors and vendors who work with the agencies
    - Tract – Numeric code designating census tract from U.S. Census Bureau. Census tracts are Land areas defined by the U.S. Census Bureau. Tracts can vary in size but each typically contains about 4,000 residents. Census tracts are usually smaller than 2 square miles in cities but are much larger in rural areas.
    - Pollutant Name: Name of chemical

- **Continuous Variables** –
    - Population – Number of people in given census tract
    - Total_crpm – Total average cancer risk of a given chemical from all source types
    - railyards_crpm – Average cancer risk from point sources and railyards
    - airport_crpm – Average cancer risk from airport sources
    - rwc_crpm – Average cancer risk from residential wood combustion sources
    - cmv_crpm – Average cancer risk from commercial marine vessels
    - biogenics_crpm – Average cancer risk from biogenic sources
    - fires_crpm – Average cancer risk from fires
    - secondary_crpm – Average cancer risk due to secondary formation, which is a process by which chemicals are transformed in the air into other chemicals
    - np_10m_releaseheight_crpm – Average cancer risk from non-point sources with 10m release height
    - np_low_releaseheight_crpm – Average cancer risk from non-point sources with low release height
    - cmv_loco_crpm - Average cancer risk from non-road sources (e.g., airplanes, trains, lawn mowers, construction vehicles, farm machinery).

- lightduty_crpm - Average cancer risk from on-road light duty mobile sources
- heavyduty_crpm – Average cancer risk from on-road heavy duty mobile sources

- **Environment setup :– The following libraries Installed and activated**
  - dplyr : Data manipulations
  - tidyverse : Data science tasks
  - readxl : to Import the .xlsx file
  - skimr : Statistical summary
  - corrplot : Correlation matrix
  - rpart : Classification – Decision Trees
  - rattle – Decision tree visualization
  - vioplot : Violin plots
  - ggplot2 : Plotting graphs
  - RcolorBrewer : Colour palette

Steps performed in R:

1. Setup the working directory using setwd(<filepath>).
2. Install the readxl package to import the dataset into R using read_excel() function & rename the columns for simplification.

```
## Importing / Reading the data into "df_cea_raw" data frame
df_cea_raw <- read_excel("ARM Dataset.xlsx", sheet = 1)

# Inspect the raw data
head(df_cea_raw)
```

3. Top 6 rows using head():

```
## # A tibble: 6 x 20
##   State County      FIPS  Tract Population Pollutant.Name Point..includes.raily~
##   <chr> <chr>      <dbl>  <dbl>      <dbl> <chr>                           <dbl>
## 1 AK    Aleutians~  2013 2.01e9       3141 ACETALDEHYDE                0
## 2 AK    Aleutians~  2016 2.02e9       1185 ACETALDEHYDE                0.00000248
## 3 AK    Aleutians~  2016 2.02e9       4376 ACETALDEHYDE                0
## 4 AK    Anchorage~  2020 2.02e9       5736 ACETALDEHYDE                0.0000288
## 5 AK    Anchorage~  2020 2.02e9       5259 ACETALDEHYDE                0.0000607
## 6 AK    Anchorage~  2020 2.02e9       4110 ACETALDEHYDE                0.000135
## # ... with 13 more variables: Airport.Cancer.Risk..per.million. <dbl>,
## #   OR.Lightduty..includes.refueling..Cancer.Risk..per.million. <dbl>,
## #   OR.Heavyduty.Cancer.Risk..per.million. <dbl>,
## #   NR..no.airports..CMV..locomotives..Cancer.Risk..per.million. <dbl>,
## #   NP.10m.ReleaseHeight.Cancer.Risk..per.million. <dbl>,
## #   NP.Low.ReleaseHeight.Cancer.Risk..per.million. <dbl>,
## #   ResidentialWoodCombustion..RWC..Cancer.Risk..per.million. <dbl>, ...
```

4. Bottom 6 rows using tail():

```
tail(df_cea_raw)
```

```
## # A tibble: 6 x 20
##   State County        FIPS Tract Population Pollutant.Name Point..includes.rail~
##   <chr> <chr>        <dbl> <dbl>      <dbl> <chr>                           <dbl>
## 1 WY    Sweetwater   56037     0      43806 TOLUENE                             0
## 2 WY    Teton        56039     0      21294 TOLUENE                             0
## 3 WY    Uinta        56041     0      21118 TOLUENE                             0
## 4 WY    Washakie     56043     0       8533 TOLUENE                             0
## 5 WY    Weston       56045     0       7208 TOLUENE                             0
## 6 WY    Entire state 56000     0     563624 TOLUENE                             0
## # ... with 13 more variables: Airport.Cancer.Risk..per.million. <dbl>,
## #   OR.Lightduty..includes.refueling..Cancer.Risk..per.million. <dbl>,
## #   OR.Heavyduty.Cancer.Risk..per.million. <dbl>,
## #   NR..no.airports..CMV..locomotives..Cancer.Risk..per.million. <dbl>,
## #   NP.10m.ReleaseHeight.Cancer.Risk..per.million. <dbl>,
## #   NP.Low.ReleaseHeight.Cancer.Risk..per.million. <dbl>,
## #   ResidentialWoodCombustion..RWC..Cancer.Risk..per.million. <dbl>, ...
```

5. Identify the column names using names():

```
names(df_cea_raw)
```

```
##  [1] "State"
##  [2] "County"
##  [3] "FIPS"
##  [4] "Tract"
##  [5] "Population"
##  [6] "Pollutant.Name"
##  [7] "Point..includes.railyards..Cancer.Risk..per.million."
##  [8] "Airport.Cancer.Risk..per.million."
##  [9] "OR.Lightduty..includes.refueling..Cancer.Risk..per.million."
## [10] "OR.Heavyduty.Cancer.Risk..per.million."
## [11] "NR..no.airports..CMV..locomotives..Cancer.Risk..per.million."
## [12] "NP.10m.ReleaseHeight.Cancer.Risk..per.million."
## [13] "NP.Low.ReleaseHeight.Cancer.Risk..per.million."
## [14] "ResidentialWoodCombustion..RWC..Cancer.Risk..per.million."
## [15] "NR.CommercialMarineVessel..CMV..Cancer.Risk..per.million."
## [16] "Biogenics.Cancer.Risk..per.million."
## [17] "Fires..ag..prescribed..and.wild..Cancer.Risk..per.million."
## [18] "Secondary.Cancer.Risk..per.million."
## [19] "Background.Cancer.Risk..per.million."
## [20] "Total.Cancer.Risk..per.million."
```

6. Summary statistics using summary() – This is where we see Length, class and mode for categorical variables, and Min, Max, Mean, Median, 1st Quartile and 3rd Quartile for continuous variables.

```
summary(df_cea_raw)
```

```
##     State              County              FIPS           Tract
##  Length:464075      Length:464075      Min.   : 1000   Min.   :0.000e+00
##  Class :character   Class :character   1st Qu.:13073   1st Qu.:1.208e+10
##  Mode  :character   Mode  :character   Median :28073   Median :2.616e+10
##                                        Mean   :28527   Mean   :2.720e+10
##                                        3rd Qu.:42003   3rd Qu.:4.102e+10
##                                        Max.   :78030   Max.   :7.803e+10
##    Population       Pollutant.Name
##  Min.   :       0   Length:464075
##  1st Qu.:    2910   Class :character
##  Median :    4080   Mode  :character
##  Mean   :   12121
##  3rd Qu.:    5512
##  Max.   :37253884
```

```
##  Total.Cancer.Risk..per.million.
##  Min.   : 0.000
##  1st Qu.: 0.000
##  Median : 0.000
##  Mean   : 1.878
##  3rd Qu.: 3.486
##  Max.   :54.085
```

7. Check the internal structure using str()

```
str(df_cea_raw)
```

```
## tibble [464,075 x 20] (S3: tbl_df/tbl/data.frame)
##  $ State                                                 : chr [1:464075] "AK" "AK" "AK" "AK" ...
##  $ County                                                : chr [1:464075] "Aleutians East Borough"
"Aleutians West Census Area" "Aleutians West Census Area" "Anchorage Municipality" ...
##  $ FIPS                                                  : num [1:464075] 2013 2016 2016 2020 2020 ...
##  $ Tract                                                 : num [1:464075] 2.01e+09 2.02e+09 2.02e+09
2.02e+09 2.02e+09 ...
##  $ Population                                            : num [1:464075] 3141 1185 4376 5736 5259 ...
##  $ Pollutant.Name                                        : chr [1:464075] "ACETALDEHYDE" "ACETALDEHYDE"
"ACETALDEHYDE" "ACETALDEHYDE" ...
##  $ Point..includes.railyards..Cancer.Risk..per.million.  : num [1:464075] 0.00 2.48e-06 0.00 2.88e-05 6.07e-
05 ...
##  $ Airport.Cancer.Risk..per.million.                     : num [1:464075] 0.000995 0.000119 0.005605 0.00465
0.002675 ...
```

8. Check the dimensionality – We observe the dataset has 464075 rows and 20 columns.

```
dim(df_cea_raw)
```

```
## [1] 464075     20
```

9. Rename the columns here using names() function.

```
# Renaming the columns as part of simplification
names(df_cea_raw)[names(df_cea_raw) == 'State'] <-  'state'
names(df_cea_raw)[names(df_cea_raw) == 'County'] <-  'county'
names(df_cea_raw)[names(df_cea_raw) == 'Pollutant.Name'] <-  'pollutant'
names(df_cea_raw)[names(df_cea_raw) == 'Point..includes.railyards..Cancer.Risk..per.million.'] <-  'railyards_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Airport.Cancer.Risk..per.million.'] <-  'airport_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'OR.Lightduty..includes.refueling..Cancer.Risk..per.million.'] <-  'lightduty_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'OR.Heavyduty.Cancer.Risk..per.million.'] <-  'heavyduty_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'NR..no.airports..CMV..locomotives..Cancer.Risk..per.million.'] <-  'cmv_loco_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'NP.10m.ReleaseHeight.Cancer.Risk..per.million.'] <-  'np_10m_releaseheight_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'NP.Low.ReleaseHeight.Cancer.Risk..per.million.'] <-  'np_low_releaseheight_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'ResidentialWoodCombustion..RWC..Cancer.Risk..per.million.'] <-  'rwc_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'NR.CommercialMarineVessel..CMV..Cancer.Risk..per.million.'] <-  'cmv_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Biogenics.Cancer.Risk..per.million.'] <-  'biogenics_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Fires..ag..prescribed..and.wild..Cancer.Risk..per.million.'] <-  'fires_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Secondary.Cancer.Risk..per.million.'] <-  'secondary_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Total.Cancer.Risk..per.million.'] <-  'total_crpm'
```

10. Subset the dataframe to consider only required columns, replace missing values with 0 and drop the raw dataframe.

```
# 4.1 Subset to consider only required columns
df_cea <- select(filter(df_cea_raw)
                 ,c(state, county, pollutant,
                    railyards_crpm,
                    airport_crpm,
                    lightduty_crpm,
                    heavyduty_crpm,
                    cmv_loco_crpm,
                    np_10m_releaseheight_crpm,
                    np_low_releaseheight_crpm,
                    rwc_crpm,
                    cmv_crpm,
                    biogenics_crpm,
                    fires_crpm,
                    secondary_crpm,
                    total_crpm  ))

# 4.3 To replace missing values with 0
df_cea[is.null(df_cea)] = 0

# 4.2 Drop the raw dataframe
remove(df_cea_raw)
```

11. Inspect the subset quickly using skimr() function.

Data summary

Name	df_cea
Number of rows	464075
Number of columns	16
Column type frequency:	
character	3
numeric	13
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
state	0	1	2	2	0	53	0
county	0	1	3	33	0	1959	0
pollutant	0	1	7	17	0	6	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
railyards_crpm	0	1	0.01	0.10	0	0	0	0.00	14.44	▇▁▁▁▁
airport_crpm	0	1	0.01	0.03	0	0	0	0.00	3.19	▇▁▁▁▁
lightduty_crpm	0	1	0.67	1.35	0	0	0	0.68	39.19	▇▁▁▁▁
heavyduty_crpm	0	1	0.04	0.08	0	0	0	0.06	5.84	▇▁▁▁▁
cmv_loco_crpm	0	1	0.24	0.56	0	0	0	0.23	20.22	▇▁▁▁▁
np_10m_releaseheight_crpm	0	1	0.03	0.12	0	0	0	0.00	9.93	▇▁▁▁▁
np_low_releaseheight_crpm	0	1	0.03	0.11	0	0	0	0.01	8.07	▇▁▁▁▁
rwc_crpm	0	1	0.13	0.32	0	0	0	0.09	9.55	▇▁▁▁▁
cmv_crpm	0	1	0.00	0.01	0	0	0	0.00	1.61	▇▁▁▁▁
biogenics_crpm	0	1	0.08	0.21	0	0	0	0.00	1.73	▇▁▁▁▁
fires_crpm	0	1	0.05	0.13	0	0	0	0.03	1.85	▇▁▁▁▁
secondary_crpm	0	1	0.58	1.38	0	0	0	0.00	6.71	▇▁▁▁▁
total_crpm	0	1	1.88	2.60	0	0	0	3.49	54.09	▇▁▁▁▁

12. To flatten the correlation matrix, a function **flat_cm()** has been created.

3. Function to Flatten the correlation matrix

```
# c is the corr. coeff. matrix
# p is the corr. p-values matrix
flat_cm <- function(c, p) {
  ut <- upper.tri(c)
  data.frame(
    row = rownames(c)[row(c)[ut]],
    column = rownames(c)[col(c)[ut]],
    cor  =(c)[ut],
    p = p[ut]
  )
}
```

13. Summarize the Total cancer risk per million variable and classify the level of threat into L0, L1, L2, L3.

a. Cut() function has not been used since the slicing criterion is based on human experience. So, even if one person is closer to cancer, that should be considered as a threat in a society, hence equal slices of the variable cannot be considered.

b. The frequency of the levels of threat is noted.

```
summary(df_cea$total_crpm)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   0.000   1.878   3.486  54.085
```

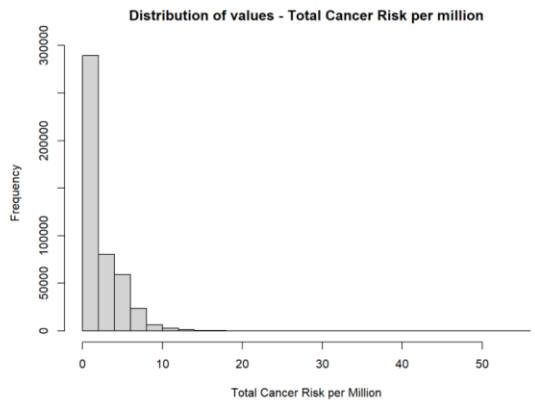
```
df_cea$level_of_threat <- as.factor(ifelse(df_cea$total_crpm <= 0.5, 'L0',
                                ifelse(df_cea$total_crpm <= 5, 'L1',
                                       ifelse(df_cea$total_crpm <= 10, 'L2', 'L3'))))

# df_cea$level_of_threat <- cut(df_cea$total_crpm, 4, labels = c('L0', 'L1', 'L2', 'L3'))

# Check frequency
as.data.frame(table(df_cea$level_of_threat))
```

```
##   Var1   Freq
## 1   L0 246071
## 2   L1 157928
## 3   L2  55109
## 4   L3   4967
```

1.5.2 Identify independent dependent variables (if any)

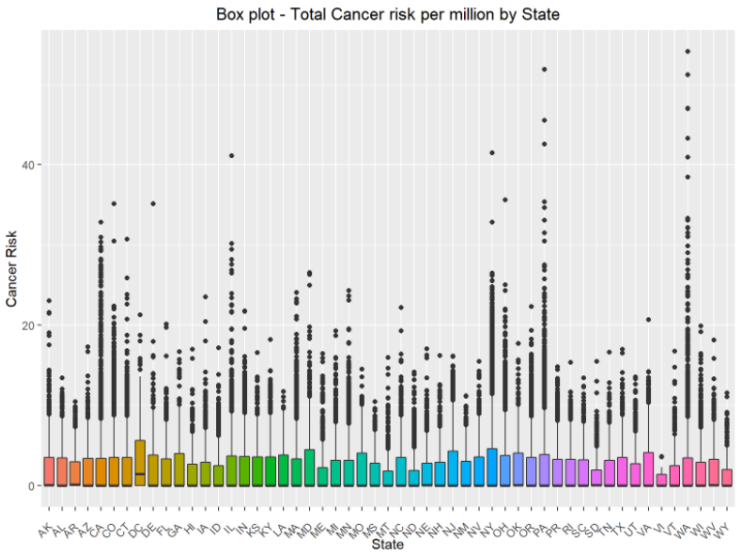
1. Observe the distribution of values within the response variable – Total_crpm.

```
# 5.1 - Histogram - Total CRPM
hist(df_cea$total_crpm, xlab="Total Cancer Risk per Million", main = "Distribution of values - Total Cancer Risk per million")
```



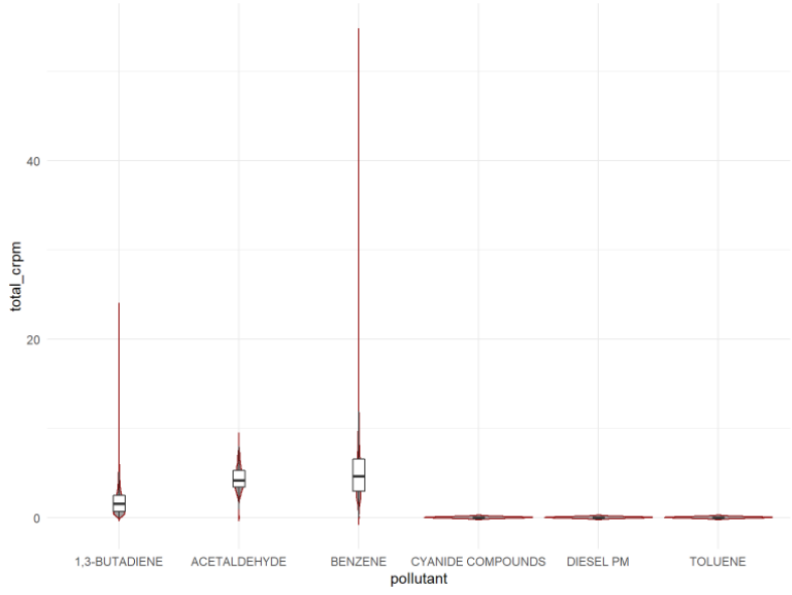
2. Apply the Box plot for Total_crpm distribution and split the distributions across State

```
# 5.2 - Box plot - Total Cancer risk by State
ggplot(df_cea, aes(x=state, y=total_crpm, fill=state)) +
  geom_boxplot()+
  labs(title="Box plot - Total Cancer risk per million by State",x="State", y = "Cancer Risk")+
  theme(legend.position = "none") +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5, hjust=1))+
  theme(plot.title = element_text(hjust = 0.5))
```



3. Distribution of Total_crpm by Pollutant. We use violin plot to observe this distribution. We can observe that the major threat is posed by Benzene, followed by 1,3-Butadiene, and Acetaldehyde.

```
# 5.3 - Violin Plot - Total CRPM by Pollutant
ggplot(df_cea, aes(x=pollutant, y=total_crpm)) +
  geom_violin(trim=FALSE, fill='#A4A4A4', color="darkred")+
  geom_boxplot(width=0.1, outlier.shape = NA) +  theme_minimal()
```



4. Applying correlation for the continuous variables. As we can see, the output is not very user friendly.

```
# 5.4 Correlation Matrix for continuous variables
df_cea_cv <- df_cea[, c(4:16)]
df_cea_cor <- cor(df_cea_cv)
round(df_cea_cor,2)
```

```
##                            railyards_crpm airport_crpm lightduty_crpm
## railyards_crpm                       1.00         0.05           0.17
## airport_crpm                         0.05         1.00           0.24
## lightduty_crpm                       0.17         0.24           1.00
## heavyduty_crpm                       0.11         0.27           0.73
## cmv_loco_crpm                        0.12         0.26           0.84
## np_10m_releaseheight_crpm            0.15         0.01           0.33
## np_low_releaseheight_crpm            0.12         0.11           0.71
## rwc_crpm                             0.09         0.13           0.62
## cmv_crpm                             0.05         0.06           0.20
## biogenics_crpm                      -0.02        -0.04          -0.15
## fires_crpm                           0.20         0.04           0.43
## secondary_crpm                      -0.02        -0.03          -0.15
## total_crpm                           0.18         0.21           0.78
##                            heavyduty_crpm cmv_loco_crpm
## railyards_crpm                       0.11          0.12
## airport_crpm                         0.27          0.26
## lightduty_crpm                       0.73          0.84
## heavyduty_crpm                       1.00          0.70
## cmv_loco_crpm                        0.70          1.00
```

5. We then flatten the correlation matrix using the function **flat_cm()** we created earlier.

```
# 5.5 Apply Flatten function here
rs<-rcorr(as.matrix(df_cea[, c(4:16)]))
flat_cm(rs$r, rs$P)
```

```
##                          row                    column          cor
## 1             railyards_crpm              airport_crpm  0.050096306
## 2             railyards_crpm             lightduty_crpm  0.172332257
## 3               airport_crpm             lightduty_crpm  0.237628510
## 4             railyards_crpm             heavyduty_crpm  0.108300827
## 5               airport_crpm             heavyduty_crpm  0.266389871
## 6             lightduty_crpm             heavyduty_crpm  0.734609525
## 7             railyards_crpm              cmv_loco_crpm  0.121881816
## 8               airport_crpm              cmv_loco_crpm  0.259299573
## 9             lightduty_crpm              cmv_loco_crpm  0.836562500
## 10            heavyduty_crpm              cmv_loco_crpm  0.701181851
## 11            railyards_crpm np_10m_releaseheight_crpm  0.149935390
## 12              airport_crpm np_10m_releaseheight_crpm  0.008782237
## 13            lightduty_crpm np_10m_releaseheight_crpm  0.325073521
## 14            heavyduty_crpm np_10m_releaseheight_crpm  0.222543753
## 15             cmv_loco_crpm np_10m_releaseheight_crpm  0.242199876
## 16            railyards_crpm np_low_releaseheight_crpm  0.115216319
## 17              airport_crpm np_low_releaseheight_crpm  0.109726017
## 18            lightduty_crpm np_low_releaseheight_crpm  0.709023638
## 19            heavyduty_crpm np_low_releaseheight_crpm  0.524414270
## 20             cmv_loco_crpm np_low_releaseheight_crpm  0.689076122
```

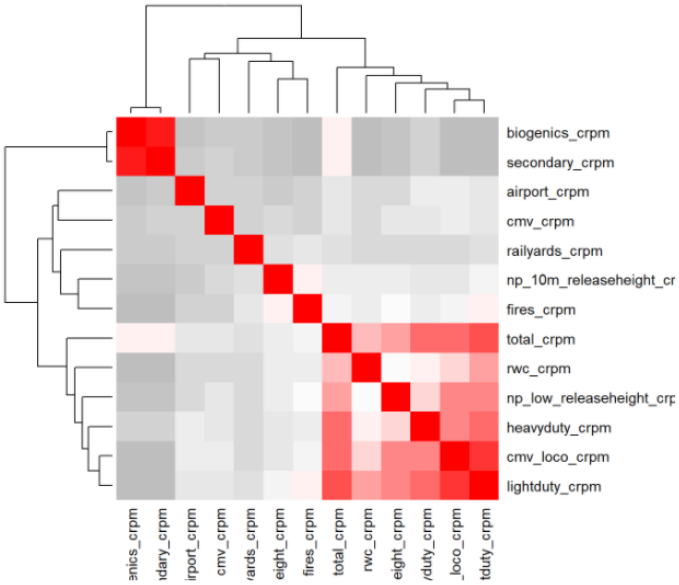
We focus on the output where one of the variables is total_crpm for which the correlation coefficient is observed, along with a p-value. Seems like the correlations are statistically significant since p-value is always less than 0.05.

```
## 66               fires_crpm            secondary_crpm -0.137815720
## 67            railyards_crpm                total_crpm  0.177917764
## 68              airport_crpm                total_crpm  0.208017606
## 69            lightduty_crpm                total_crpm  0.781906506
## 70            heavyduty_crpm                total_crpm  0.715169598
## 71             cmv_loco_crpm                total_crpm  0.718495915
## 72 np_10m_releaseheight_crpm                total_crpm  0.300491574
## 73 np_low_releaseheight_crpm                total_crpm  0.605575374
## 74                   rwc_crpm                total_crpm  0.549438143
## 75                   cmv_crpm                total_crpm  0.225394056
## 76            biogenics_crpm                total_crpm  0.438772691
## 77               fires_crpm                total_crpm  0.344282481
## 78            secondary_crpm                total_crpm  0.466994218
```

```
## 66 0.000000e+00
## 67 0.000000e+00
## 68 0.000000e+00
## 69 0.000000e+00
## 70 0.000000e+00
## 71 0.000000e+00
## 72 0.000000e+00
## 73 0.000000e+00
## 74 0.000000e+00
## 75 0.000000e+00
## 76 0.000000e+00
## 77 0.000000e+00
## 78 0.000000e+00
```

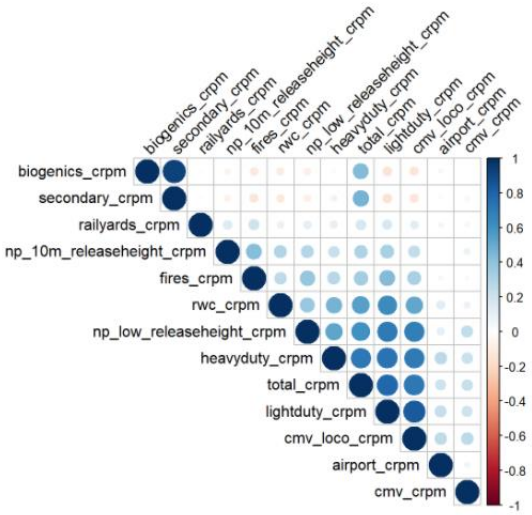
6. Correlation Plot 1 – Using a heatmap.
   a. This plot shows us the 5 main variables that have an impact on total_crpm.
      i. Lightduty_crpm
      ii. Cmv_loco_crpm
      iii. Heavyduty_crpm
      iv. Np_low_releaseheight_crpm
      v. Rwc_crpm

```
# 5.6 - Correlation Matrix Analysis using Heatmap
col <- colorRampPalette(c("grey", "white", "red"))(20)
heatmap(x = df_cea_cor, col = col, symm = TRUE)
```



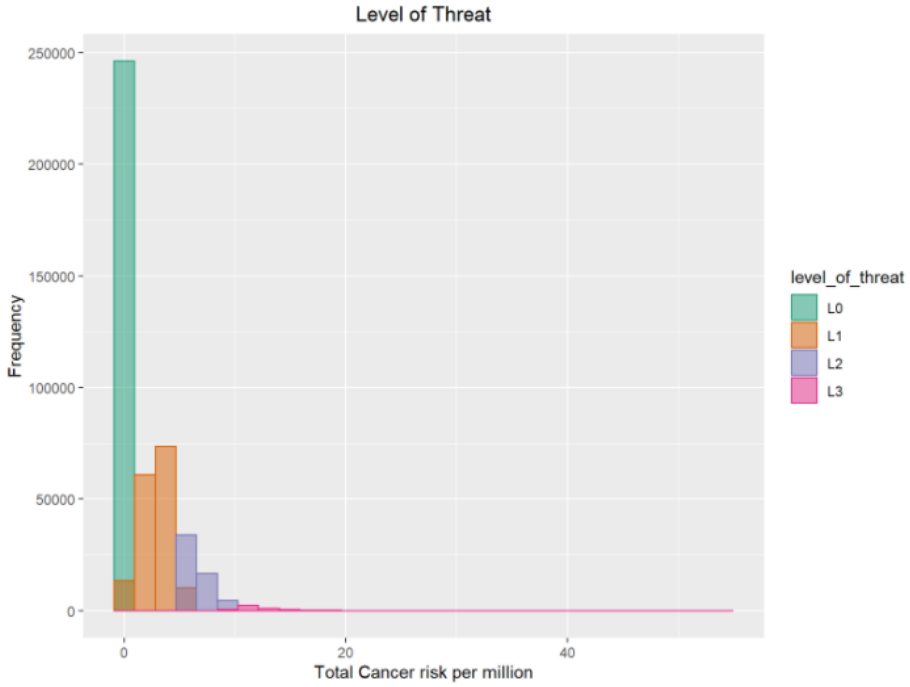
7. Correlation Plot 2 – Correlogram
   a. The same relationships can be observed as shown in the heatmap.

```
# 5.7 - Correlogram
corrplot(df_cea_cor, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)
```



8. Using ggplot, we could observe the distribution of total risk of cancer per million by the Level of Threat variable that we created earlier. This was created during the data wrangling phase.

```
# 5.8 - Level of Threat
ggplot(df_cea, aes(x=total_crpm, fill=level_of_threat, color=level_of_threat)) +
  geom_histogram( alpha=0.5, position="identity")+
  scale_color_brewer(palette="Dark2")+
  scale_fill_brewer(palette="Dark2")+
  labs(title="Level of Threat",x="Total Cancer risk per million", y = "Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
```



1.5.3 Data Pre-processing steps

1. Subset only the required predictor variables and the response variable into a new data frame.

```
# Subset to only predictor and response variables.
df_cea_model <- df_cea[, 4:17]
names(df_cea_model)
```

```
##  [1] "railyards_crpm"            "airport_crpm"
##  [3] "lightduty_crpm"            "heavyduty_crpm"
##  [5] "cmv_loco_crpm"             "np_10m_releaseheight_crpm"
##  [7] "np_low_releaseheight_crpm" "rwc_crpm"
##  [9] "cmv_crpm"                  "biogenics_crpm"
## [11] "fires_crpm"                "secondary_crpm"
## [13] "total_crpm"                "level_of_threat"
```

2. Export the subset file to a .csv version in order to proceed with the analysis using SAS Enterprise Miner.

```
# Write the .csv file for SAS task
write.csv(df_cea_model,"D:/University/ASDM/Developments/Task 1 - Classification/Cancer_Risk_USA_SAS.csv", row.names = TRUE)
```

3. Setup the random number generator using set.seed() and apply the split using Sampling technique into 80% and 20% of the data respectively.

```
# Random number generator
set.seed(28)

# Use Sampling function to split the training vs. test datasets.
cea_split <- sample(2, nrow(df_cea_model),replace=TRUE, prob=c(0.8,0.2))
head(cea_split)
```

```
## [1] 1 1 1 2 1 1
```

4. Split the data into training and testing dataframes.

```
# Create train & test datasets
train_cea <- df_cea[cea_split==1,]
test_cea <- df_cea[cea_split==2,]

# Inspect the datasets & split
dim(train_cea)
```

```
## [1] 371031     17
```

```
dim(test_cea)
```

```
## [1] 93044    17
```

5. As part of the validation, inspect the response variable frequency and overall dimensionality.

```
as.data.frame(table(train_cea$level_of_threat))
```

```
##   Var1   Freq
## 1   L0 196599
## 2   L1 126298
## 3   L2  44193
## 4   L3   3941
```

```
as.data.frame(table(test_cea$level_of_threat))
```

```
##   Var1  Freq
## 1   L0 49472
## 2   L1 31630
## 3   L2 10916
## 4   L3  1026
```

6. Also, create a function **model_predict()** that would perform the visualization of the plot, prediction, confusion matrix creation, accuracy rate % and error rate % calculation, along with precision %, recall % and f1 score % calculations. This would eliminate redundancy in our model and optimize it overall.

```
# Function to be used to predict the class and calculate accuracy
model_predict <- function(df, tree, v_title ) {

  # 1. Plot the tree
  fancyRpartPlot(tree, main = v_title)

  # 2. Cancer Risk Prediction
  risk_prediction <- predict(tree, df, type = 'class')

  # 3. Confusion matrix
  cmat <- table(df$level_of_threat, risk_prediction)
  print('Confusion Matrix is created')

  # 4. Accuracy % & Error %
  acc <- sum(diag(cmat)) * 100 / sum(cmat)
  print(paste('Accuracy :', round(acc,2), '%'))
  print(paste('Error :', round(100 - acc,2), '%'))

  # 5. Precision, Recall & F1 Score
  v_diag <- diag(cmat) # number of correctly classified instances per class
  rowsums <- apply(cmat, 1, sum) # number of instances per class
  colsums <- apply(cmat, 2, sum) # number of predictions per class
  v_precision <- v_diag / colsums
  v_recall <- v_diag / rowsums
  v_f1 <- 2 * v_precision * v_recall / (v_precision + v_recall)
  df_prf1 <- data.frame(round(v_precision*100,2), round(v_recall*100,2), round(v_f1*100,2))
  colnames(df_prf1) <- c('Precision (%)', 'Recall (%)', 'F1 Score (%)')
  df_prf1
}
```

1.5.4 Assumptions (if any)

- L0, L1, L2, L3 are essentially low to high levels of risks involved when a group of people are exposed to toxic pollutants in a county's tract.

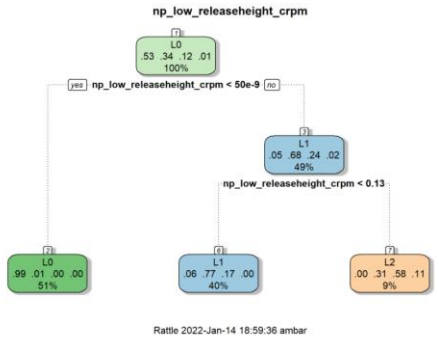
1.6 Task: Classification

1.6.1 Data Exploration and Attribute Visualization in R

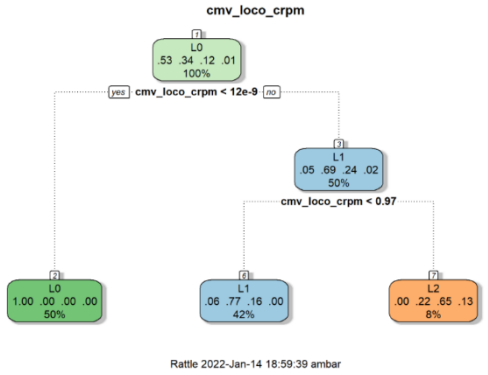
1.6.1.1 Model Building in R

1. We use rpart() to create the decision tree model. Initially, we create dummy trees based on the strong correlations.
    a. Np_low_releaseheight_crpm
    b. Cmv_loco_crpm
    c. Lightduty_crpm
    d. Heavyduty_crpm

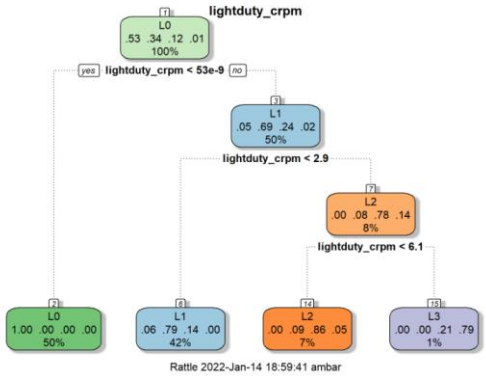
```
# 1. Understand the variables
tree1 <- rpart(level_of_threat ~ np_low_releaseheight_crpm, data=train_cea, method="class")
fancyRpartPlot(tree1, main = "np_low_releaseheight_crpm")
```



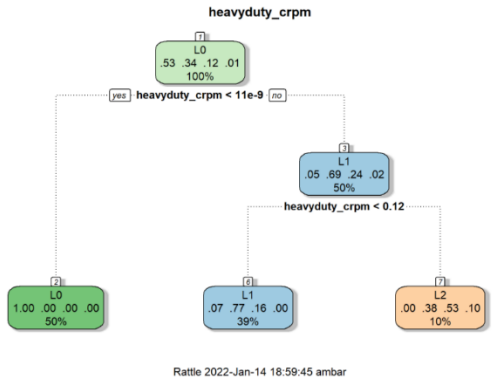

```
tree2 <- rpart(level_of_threat ~ cmv_loco_crpm, data=train_cea, method="class")
fancyRpartPlot(tree2, main = "cmv_loco_crpm")
```




```
tree3 <- rpart(level_of_threat ~ lightduty_crpm, data=train_cea, method="class")
fancyRpartPlot(tree3, main = "lightduty_crpm")
```

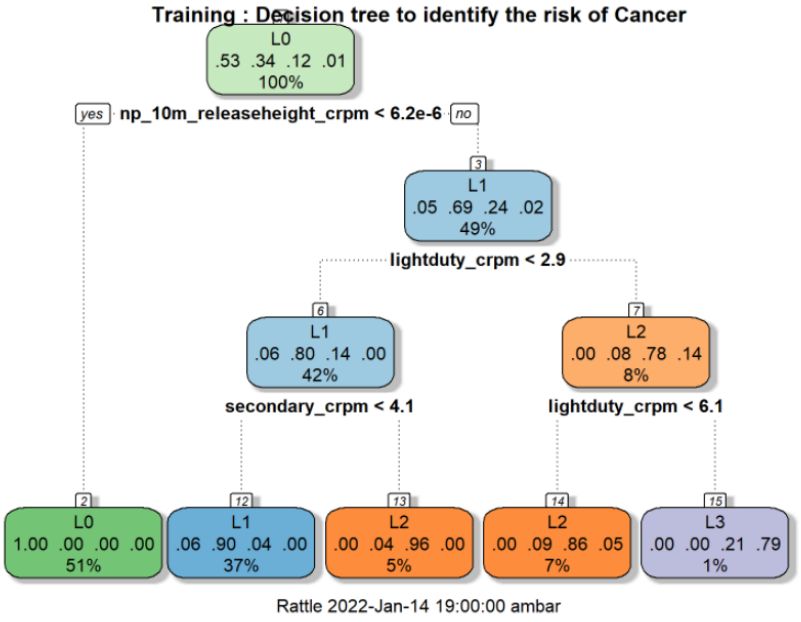



```
tree4 <- rpart(level_of_threat ~ heavyduty_crpm, data=train_cea, method="class")
fancyRpartPlot(tree4, main = "heavyduty_crpm")
```



2. Train the model using the variables.

```
# 2. Train the model
rpart_tree <- rpart(level_of_threat ~ railyards_crpm+
                                      airport_crpm +
                                      rwc_crpm +
                                      cmv_crpm +
                                      biogenics_crpm +
                                      fires_crpm +
                                      secondary_crpm +
                                      np_10m_releaseheight_crpm +
                                      np_low_releaseheight_crpm +
                                      cmv_loco_crpm +
                                      lightduty_crpm +
                                      heavyduty_crpm,
                    data=train_cea, method="class")
```



3. Evaluate the model and predict using the **model_predict()** function we created.

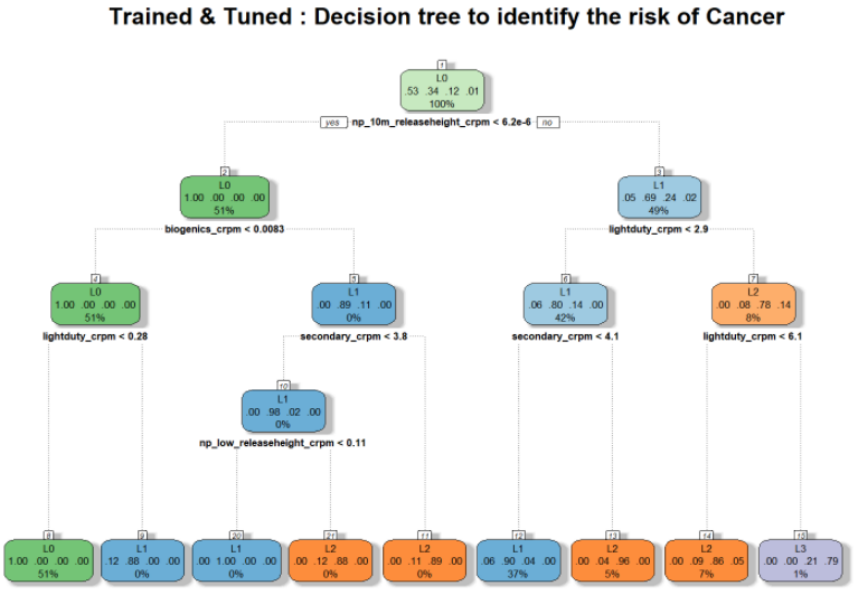
```
# 3. Evaluate the trained model
model_predict(train_cea, rpart_tree, 'Training : Decision tree to identify the risk of Cancer')
```

```
## [1] "Confusion Matrix is created"
## [1] "Accuracy : 94.84 %"
## [1] "Error : 5.16 %"
```

```
##    Precision (%) Recall (%) F1 Score (%)
## L0         99.64      95.62         97.59
## L1         90.02      97.17         93.46
## L2         90.24      87.04         88.61
## L3         79.48      68.82         73.77
```

4. Accuracy Tuning is performed to determine the best fit. We introduce a control variable that would establish the best fit for the decision tree model.

```
# 4. Accuracy Tuning - (not needed if > 78~80%) - to identify more nodes
ctrl <- rpart.control(minsplit = 4,
    minbucket = round(5 / 3),
    maxdepth = 4,
    cp = 0)
tune_fit_train <- rpart(level_of_threat ~ railyards_crpm+
                                    airport_crpm +
                                    rwc_crpm +
                                    cmv_crpm +
                                    biogenics_crpm +
                                    fires_crpm +
                                    secondary_crpm +
                                    np_10m_releaseheight_crpm +
                                    np_low_releaseheight_crpm +
                                    cmv_loco_crpm +
                                    lightduty_crpm +
                                    heavyduty_crpm,
                        data=train_cea, method="class", control = ctrl)

model_predict(train_cea, tune_fit_train, 'Trained & Tuned : Decision tree to identify the risk of Cancer')
```



Rattle 2022-Jan-14 19:00:15 ambar

```
## [1] "Confusion Matrix is created"
## [1] "Accuracy : 95 %"
## [1] "Error : 5 %"
```

```
##    Precision (%) Recall (%) F1 Score (%)
## L0         99.96      95.61        97.74
## L1         90.05      97.60        93.68
## L2         90.24      87.16        88.67
## L3         79.48      68.82        73.77
```

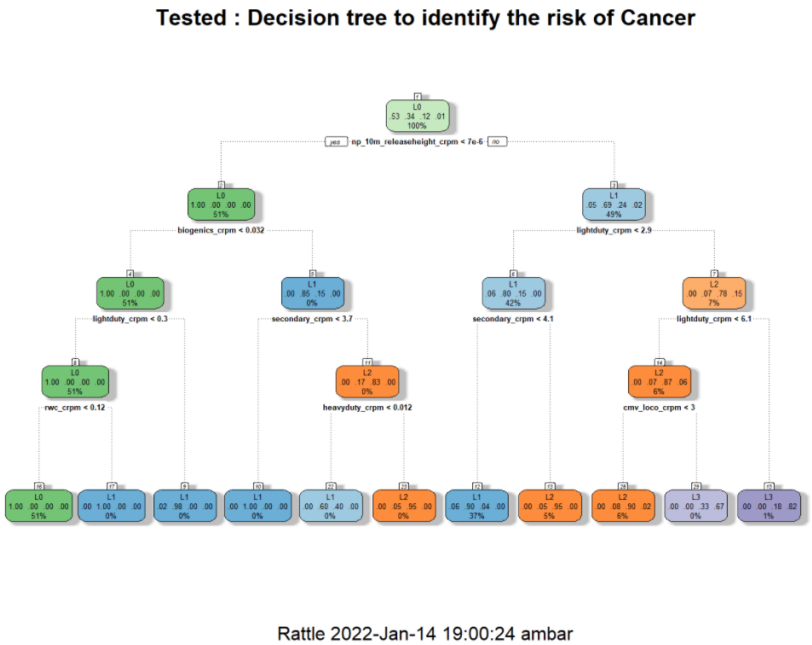
1.6.1.2 Model Assessment in R

1. Model assessment is done using the Test dataset.

```
# 1. Test the model
tune_fit_test <- rpart(level_of_threat ~ railyards_crpm+
                                         airport_crpm +
                                         rwc_crpm +
                                         cmv_crpm +
                                         biogenics_crpm +
                                         fires_crpm +
                                         secondary_crpm +
                                         np_10m_releaseheight_crpm +
                                         np_low_releaseheight_crpm +
                                         cmv_loco_crpm +
                                         lightduty_crpm +
                                         heavyduty_crpm,
                       data=test_cea, method="class", control = ctrl)

# 2. Evaluate the tested model
model_predict(test_cea, tune_fit_test, 'Tested : Decision tree to identify the risk of Cancer')
```

1.6.1.3 Results visualisation in R

1. Results are visualized for the testing dataset as shown below.



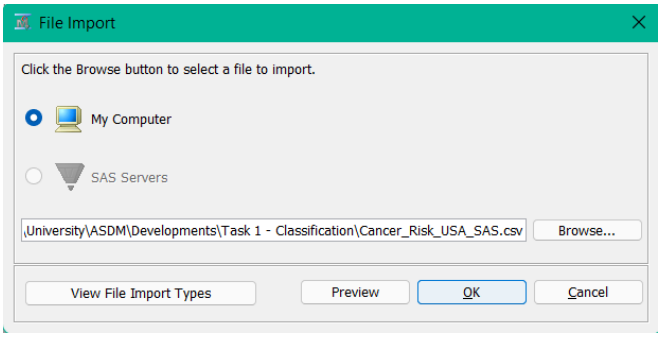

```
## [1] "Confusion Matrix is created"
## [1] "Accuracy : 95.09 %"
## [1] "Error : 4.91 %"
```

```
##    Precision (%) Recall (%) F1 Score (%)
## L0         99.95      95.63        97.74
## L1         89.80      97.87        93.66
## L2         92.26      85.35        88.67
## L3         78.24      86.55        82.18
```

1.6.2 Data Exploration and Attribute Visualization in SAS EM

1.6.2.1 Model Building in SAS EM

1. A model has been built within SAS Enterprise Miner using the exported .csv file after the cleaning has been completed in R.
2. A new project is created, and we import the .csv file into SAS EM.



3. Assign the Score role as 'Train' and the number of guessing rows is 500.

Property	Value
General	
Node ID	FIMPORT
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Import File	D:\University\ASDM\Developments\Task 1 - C
Maximum Rows to I	1000000
Maximum Columns	10000
Delimiter	,
Name Row	Yes
Number of Rows to	0
Guessing Rows	500
File Location	Local
File Type	csv
Advanced Advisor	No
Rerun	No
Score	
Role	Train
Report	
Summarize	No
Status	
Create Time	30/12/21 01:29
Run ID	0e29fb05-4e52-442d-8ad0-90f6d9b91f27
Last Error	
Last Status	Complete

4. Apply the following roles for the variables. Level of threat is selected as our target/response variable, and other predictor/input variables are selected.

Name	Role	Level	Report
airport_crpm	**Input**	**Interval**	**No**
biogenics_crpm	**Input**	**Interval**	**No**
cmv_crpm	**Input**	**Interval**	**No**
cmv_loco_crpm	**Input**	**Interval**	**No**
fires_crpm	**Input**	**Interval**	**No**
heavyduty_crpm	**Input**	**Interval**	**No**
level_of_threat	**Target**	**Nominal**	**No**
lightduty_crpm	**Input**	**Interval**	**No**
np_10m_releaseheight_crpm	**Input**	**Interval**	**No**
np_low_releaseheight_crpm	**Input**	**Interval**	**No**
railyards_crpm	**Input**	**Interval**	**No**
rwc_crpm	**Input**	**Interval**	**No**
secondary_crpm	**Input**	**Interval**	**No**
total_crpm	**Rejected**	**Interval**	**No**
VAR1	**Rejected**	**Nominal**	**No**

5. Later we partition the data into train, validate and test of 60 %, 30 % and 10 % of the original data.

Property	Value
General	
Node ID	Part
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Output Type	Data
Partitioning Method	Default
Random Seed	12345
Data Set Allocations	
Training	60.0
Validation	30.0
Test	10.0
Report	
Interval Targets	Yes
Class Targets	Yes
Status	
Create Time	30/12/21 01:46
Run ID	ca06f9bb-7348-4eac-a86b-833cdaaeab42
Last Error	
Last Status	Complete
Last Run Time	30/12/21 01:49
Run Duration	0 Hr. 0 Min. 2.66 Sec.
Grid Host	
User-Added Node	No

6. Apply statExplore for exploring the data using correlations and statistical summaries using fit statistics.

Property	Value
General	
Node ID	Stat
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Data	
Number of Observations	ALL
Validation	No
Test	No
Standard Reports	
Interval Distributions	Yes
Class Distributions	Yes
Level Summary	Yes
Use Segment Variables	No
Cross-Tabulation	
Variable Selection	
Hide Rejected Variables	Yes
Number of Selected Variables	1000
Chi-Square Statistics	
Chi-Square	Yes
Interval Variables	No
Number of Bins	5
Correlation Statistics	
Correlations	Yes
Pearson Correlations	Yes
Spearman Correlations	No
Status	
Create Time	30/12/21 01:50
Run ID	f048508f-6b74-4117-9759-b0efdbd7c89
Last Error	
Last Status	Complete
Last Run Time	30/12/21 01:53
Run Duration	0 Hr. 0 Min. 6.66 Sec.
Grid Host	
User-Added Node	No

7. Apply control point

Property	Value
General	
Node ID	CNTRL
Imported Data	
Exported Data	
Status	
Create Time	
Run ID	
Last Error	
Last Status	
Last Run Time	
Run Duration	
Grid Host	
User-Added Node	No

8. Apply the Decision tree using the following settings.

.. Property	Value
General	
Node ID	Tree
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Interactive	
Import Tree Model	No
Tree Model Data Set	
Use Frozen Tree	No
Use Multiple Targets	No
Splitting Rule	
Interval Target Criterion	ProbF
Nominal Target Criterion	ProbChisq
Ordinal Target Criterion	Entropy
Significance Level	0.2
Missing Values	Use in search
Use Input Once	No
Maximum Branch	2
Maximum Depth	4
Minimum Categorical Size	3
Node	
Leaf Size	5
Number of Rules	5
Number of Surrogate Rules	0
Split Size	.
Split Search	
Use Decisions	No
Use Priors	No
Exhaustive	5000
Node Sample	20000
Subtree	
Method	Assessment
Number of Leaves	1
Assessment Measure	Decision
Assessment Fraction	0.25
Cross Validation	
Perform Cross Validation	No
Number of Subsets	10
Number of Repeats	1

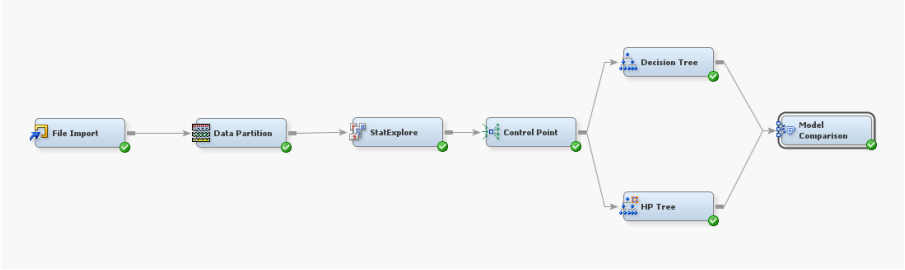
9. Simultaneously, apply a HP tree as well.

.. Property	Value
General	
Node ID	HPTree
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Splitting Rule	
Interval Target Criterion	Variance
Nominal Target Criterion	Entropy
Interval Bins	100
Minimum Distance	0.01
Significance Level	0.2
Bonferroni	No
Missing Values	Largest
Use Input Once	No
Maximum Branch	2
Maximum Depth	4
Minimum Categorical Size	3
Node	
Leaf Size	3
Surrogate Rules	0
Validation	
Create Validation	No
Validation	0.15
Partition Seed	12345
Split Search	
Exhaustive Search Comparis	500000
Fast Search Comparisons	1000000
Subtree	
Subtree Method	Cost-Complexity
Selection Method	Automatic
Confidence	0.25
Nominal Target Assessment	Entropy
Minimum Subtree	No
Assessment Threshold Valu	1.0
Number of Leaves	1
Cross Validation Folds	10
Cross Validation Seed	12345
Score	

10. Apply Model comparison to check the results and compare.

Property	Value
General	
Node ID	MdlComp
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Assessment Reports	
Number of Bins	20
ROC Chart	Yes
Recompute	No
Model Selection	
Selection Data	Default
Selection Statistic	Default
HP Selection Statistic	Default
SAS Viya Selection Statistic	
Selection Table	Train
Selection Depth	10
Score	
Selection Editor	
Report	
Selected Model	
Target	level_of_threat
Model Node	HPTree
Model Description	HP Tree
Selection Criteria	Valid: Misclassification Rate
Status	
Create Time	30/12/21 02:25
Run ID	d6703f9d-9160-46b6-aea5-d736c2595f
Last Error	
Last Status	Complete
Last Run Time	30/12/21 02:33
Run Duration	0 Hr. 0 Min. 9.33 Sec.
Grid Host	
User-Added Node	No

11. The final model looks like this.



1.6.2.2 Model Assessment in SAS EM

1. Summarizing and looking at the observations for initial validations.

Variable Summary

Role	Measurement Level	Frequency Count
INPUT	INTERVAL	12
REJECTED	INTERVAL	1
REJECTED	NOMINAL	1
TARGET	NOMINAL	1

The CONTENTS Procedure

Data Set Name	EMWS1.FIMPORT_DATA	Observations	464075
Member Type	DATA	Variables	15
Engine	V9	Indexes	0
Created	30/12/2021 01:49:16	Observation Length	120
Last Modified	30/12/2021 01:49:16	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	WINDOWS_32		
Encoding	wlatin1 Western (Windows)		

2. Summary Statistics for partitioned data

Summary Statistics for Class Targets

Data=DATA

Variable	Numeric Value	Formatted Value	Frequency Count	Percent	Label
level_of_threat	.	L0	246071	53.0240	
level_of_threat	.	L1	157928	34.0307	
level_of_threat	.	L2	55109	11.8750	
level_of_threat	.	L3	4967	1.0703	

Data=TEST

Variable	Numeric Value	Formatted Value	Frequency Count	Percent	Label
level_of_threat	.	L0	24608	53.0219	
level_of_threat	.	L1	15793	34.0286	
level_of_threat	.	L2	5512	11.8765	
level_of_threat	.	L3	498	1.0730	

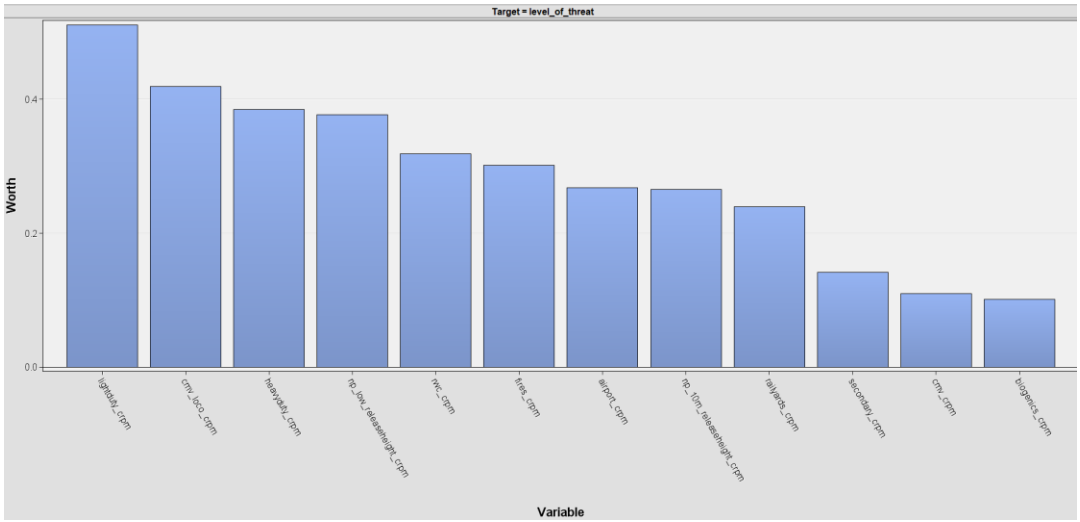
Data=TRAIN

Variable	Numeric Value	Formatted Value	Frequency Count	Percent	Label
level_of_threat	.	L0	147642	53.0241	
level_of_threat	.	L1	94757	34.0310	
level_of_threat	.	L2	33064	11.8746	
level_of_threat	.	L3	2980	1.0702	

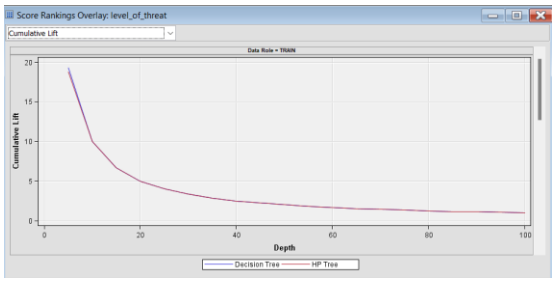
Data=VALIDATE

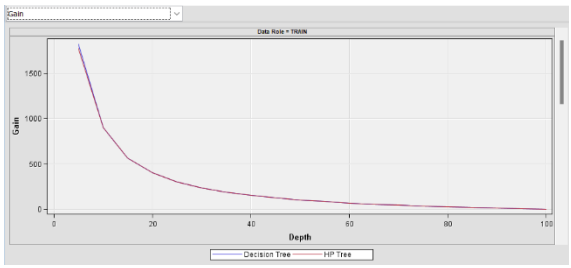
Variable	Numeric Value	Formatted Value	Frequency Count	Percent	Label
level_of_threat	.	L0	73821	53.0243	
level_of_threat	.	L1	47378	34.0308	
level_of_threat	.	L2	16533	11.8754	
level_of_threat	.	L3	1489	1.0695	

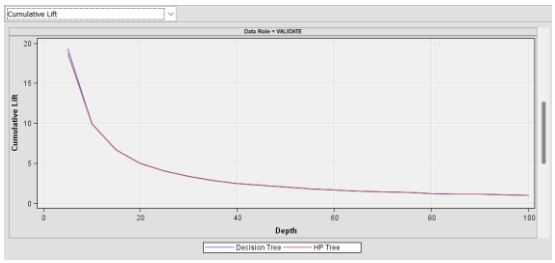
3. Variable importance evaluation – lightduty_crpm is the most important and biogenics_crpm is the least important.

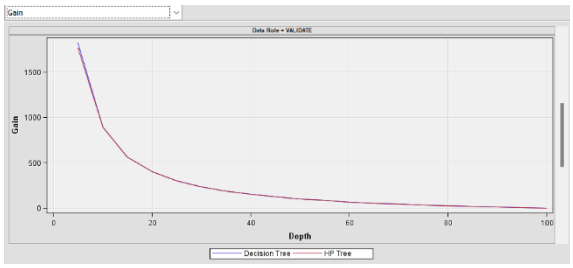


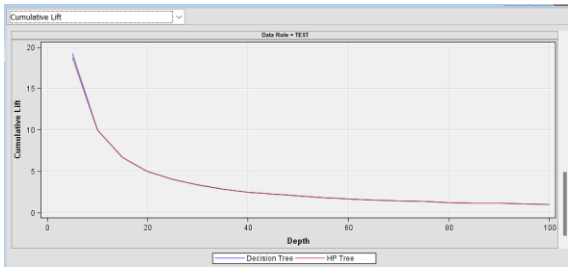
4. Model Comparison – Decision Tree vs. HP tree

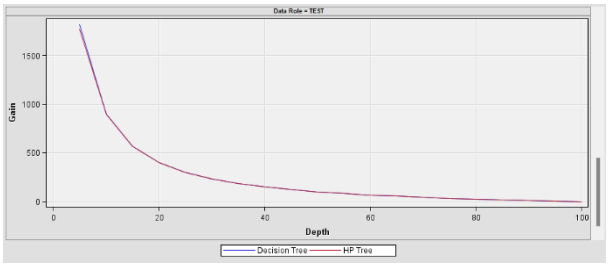













```
*------------------------------------------------------------*
User:                 ambar
Date:                 14 January 2022
Time:                 22:39:37
*------------------------------------------------------------*
* Training Output
*------------------------------------------------------------*
```

Variable Summary

Role	Measurement Level	Frequency Count
TARGET	NOMINAL	1

Obs	TARGET	TARGETLABEL	_AUR_	_GINI_	KS	_KS_PROB_ CUTOFF	_KS_BIN_	BINNED_KS_ PROB_ CUTOFF
1	level_of_threat		0.995	0.99	0.925	0.01	0.923	0.447

Obs	TARGET	TARGETLABEL	_VAUR_	_VGINI_	VKS	_VKS_ PROB_ CUTOFF_	_VKS_ BIN_	_VBINNED_ KS_PROB_ CUTOFF_
1	level_of_threat		0.995	0.99	0.925	0.02	0.924	0.447

Obs	TARGET	TARGETLABEL	_TAUR_	_TGINI_	TKS	_TKS_ PROB_ CUTOFF_	_TKS_ BIN_	_TBINNED_ KS_PROB_ CUTOFF_
1	level_of_threat		0.995	0.991	0.927	0.01	0.921	0.447

5. Fit Statistics – HP Tree is better according to SAS in comparison to Decision tree

Fit Statistics

Selected Model	Predecessor Node	Model Node	Model Description	Target Label	Target Variable	Selection Criterion: Valid: Misclassification Rate	Train: Sum of Frequencies	Train: Misclassification Rate	Train: Maximum Absolute Error	Train: Sum of Squared Errors	Train: Average Squared Error	Train: Root Average Squared Error	Train: Divisor for ASE	Train: Total Degrees of Freedom
Y	HPTree	HPTree	HP Tree		level of threat	0.040741	278443	0.041007	0.999855	19014.22	0.017072	0.13066	1113772	.
	Tree	Tree	Decision Tree		level of threat	0.081927	278443	0.08285	0.999979	38157.75	0.03426	0.185094	1113772	835329

Valid: Sum of Frequencies	Valid: Misclassification Rate	Valid: Maximum Absolute Error	Valid: Sum of Squared Errors	Valid: Average Squared Error	Valid: Root Average Squared Error	Valid: Divisor for VASE	Test: Sum of Frequencies	Test: Sum of Weights Times Freqs	Test: Misclassification Rate	Test: Maximum Absolute Error	Test: Sum of Squared Errors	Test: Average Squared Error	Test: Root Average Squared Error	Test: Divisor for TASE
139221	0.040741	0.999855	9430.63	0.016935	0.130133	556884	46411	.	0.04124	0.998528	3164.659	0.017047	0.130564	185644
139221	0.081927	0.999979	18950.44	0.034029	0.184471	556884	46411	185644	0.083127	0.999979	6390.09	0.034421	0.18553	185644

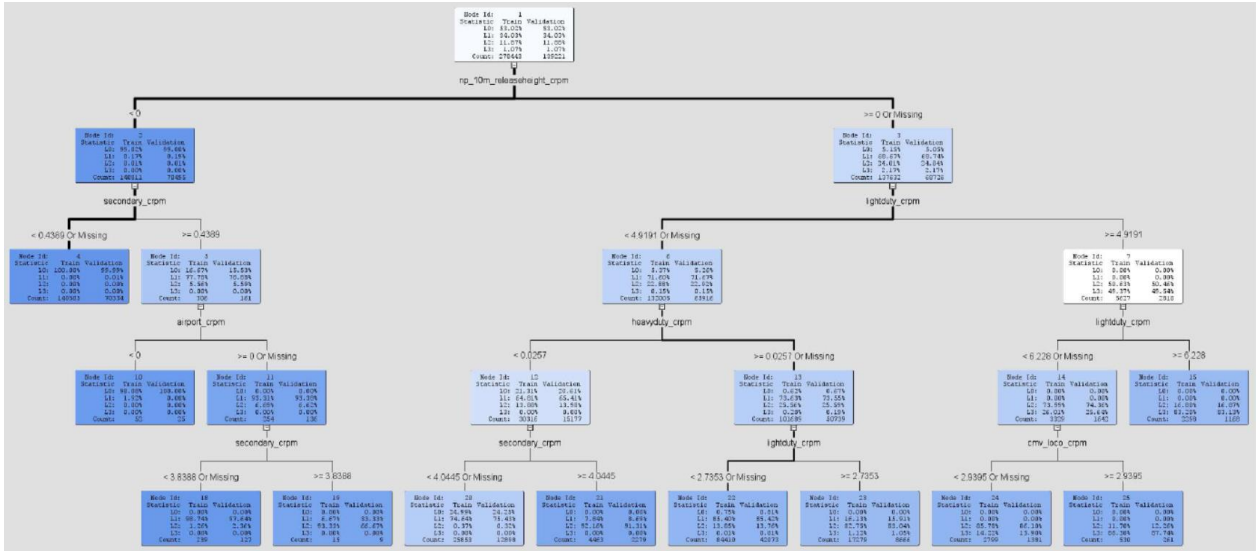
Train: Frequency of Classified Cases	Train: Number of Wrong Classifications	Valid: Frequency of Classified Cases	Valid: Number of Wrong Classifications	Test: Frequency of Classified Cases	Test: Number of Wrong Classifications	Train: Roc Index	Train: Gini Coefficient	Train: Kolmogorov-Smirnov Statistic	Train: Kolmogorov-Smirnov Probability Cutoff	Train: Bin-Based Two-Way Kolmogorov-Smirnov Statistic	Bin-Based Two-Way Kolmogorov-Smirnov Probability Cutoff	Valid: Roc Index	Valid: Gini Coefficient	Valid: Kolmogorov-Smirnov Statistic	Valid: Kolmogorov-Smirnov Probability Cutoff	Valid: Bin-Based Two-Way Kolmogorov-Smirnov Statistic	Valid: Bin-Based Two-Way Kolmogorov-Smirnov Probability Cutoff
278443	11418	139221	5672	46411	1914	0.993	0.986	0.93	0.01	0.907	0.012	0.993	0.986	0.93	0.01	0.906	0.012
.	.	.	.	.	.	0.995	0.99	0.925	0.01	0.923	0.447	0.995	0.99	0.925	0.02	0.924	0.447

Test: Roc Index	Test: Gini Coefficient	Test: Kolmogorov-Smirnov Statistic	Test: Kolmogorov-Smirnov Probability Cutoff	Test: Bin-Based Two-Way Kolmogorov-Smirnov Statistic	Test: Bin-Based Two-Way Kolmogorov-Smirnov Probability Cutoff	Train: Gain	Valid: Gain	Train: Lift	Train: Cumulative Lift	Valid: Lift	Valid: Cumulative Lift	Train: Percent Response	Train: Cumulative Percent Response	Valid: Percent Response	Valid: Cumulative Percent Response	Train: Percent Captured Response	Train: Cumulative Percent Captured Response	Valid: Percent Captured Response	Valid: Cumulative Percent Captured Response
0.993	0.986	0.933	0.01	0.91	0.012	897.1457	896.164	1.193986	9.971457	1.204166	9.96164	1.277848	10.67182	1.287883	10.6542	5.969866	99.71708	6.020789	99.62284
0.995	0.991	0.927	0.01	0.921	0.447	897.4278	896.1404	0.679646	9.974278	0.641369	9.961404	0.727383	10.67484	0.685959	10.65395	3.398196	99.74529	3.206824	99.62048

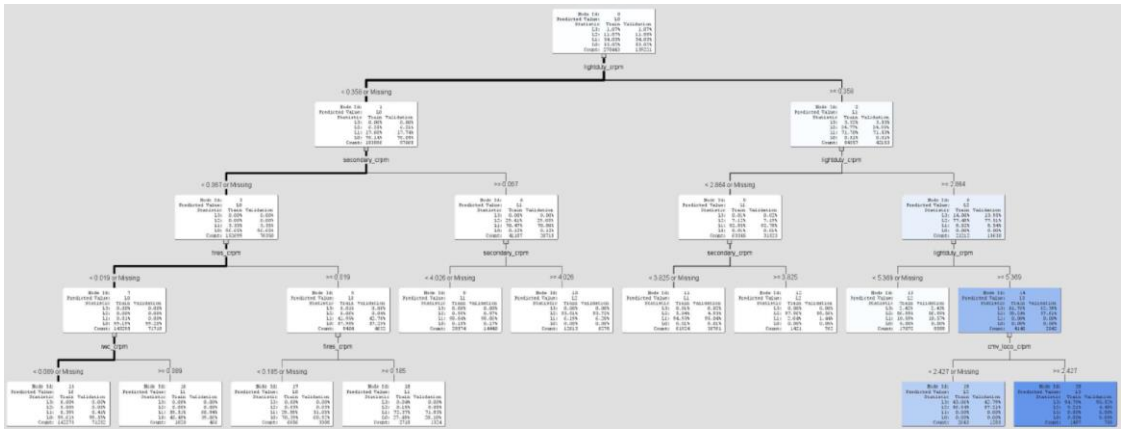
Test: Gain	Test: Lift	Test: Percent Response	Test: Percent Captured Response	Test: Cumulative Lift	Test: Cumulative Percent Response	Test: Cumulative Percent Captured Response
899.8061	1.277989	1.37131	6.391186	9.998061	10.72813	100
899.8061	0.782466	0.839603	3.913088	9.998061	10.72813	100

1.6.2.3 Results visualisation in SAS EM

1. Decision Tree - are a machine learning technique for making predictors; they are built by repeatedly splitting data into smaller and smaller clusters. The tress are trained by passing data down from a root node to leaves.



2. HP Tree - enables you to create and visualize a tree model and determine input variable importance.



1.7 Results analysis and discussion

1.7.1 Result comparison between R and SAS EM

- If the risk of cancer from non-point sources with low release height is greater than or equal to 0.13, we could see 9% of the risks could result in L2.
- If the risk of cancer from non-road locomotive emissions is greater than or equal to 0.97, we could see 8% of the risks could result in L2.
- If the risk of cancer from light duty vehicle emissions is within the range of 3.0 to 6.1, we could see 7% of the risks could result in L2 and if it exceeds 6.1, there is a 1% chance that the risks could results in L3.
- If the risk of cancer from heavy duty vehicle emissions is greater than or equal to 0.97, we could see that 10% of the risks could result in L2.
- The L3 risks are primarily depending upon the cancer risks due to light duty vehicle emissions, which needs to be kept in check first.
- In SAS, we see the fit diagnostics and cumulative lift and score rankings for the model. Since the data is split into 60 % for training, 30% for validation and 10% for testing, we wouldn't perhaps get the exact outputs as R, however, we could observe the relationships are linear and the outcomes are similar during the model evaluation.

1.7.2 Critical findings

- Within the confusion matrix, we see the accuracy as 94.84% and error to be 5.16% before the accuracy tuning. Up on accuracy tuning for the training dataset, we see the accuracy as 95% and error as 5%. Ultimately, for testing dataset, we see the accuracy as 95.09% and error as 4.91%. Precision, recall and F1 scores are all above 78% for testing, and more than 68% for training which is a relatively good score to evaluate the goodness of fit of the model, and hence this model could be recommended.
- Based on SAS EM model comparison, HP tree is more efficient than Decision tree which seems to be producing more accurate predictions along with variable importance.

1.8 Conclusion

In conclusion, we successfully performed the classification task using R and SAS EM based on CRISP-DM methodology, gained valuable insights, compared the design and model evaluation aspects within SAS and R, performed goodness of fit, improved the model with accuracy tuning using a control variable or parametrizing input arguments within SAS decision / HP trees. We can say that it is statistically proven that light duty vehicle emissions contribute to L3 level cancer risks among the people of USA, and thus they must be prioritized to be curbed or reduced for the welfare of the society.

2. Association Rules Mining – Concomitant Pollutants in CA, USA

2.1 Abstract

This module of the project aims to support and discover the insights from the research excerpts of National Air Toxicity Assessment (NATA) executed by the US Environmental Protection Agency (EPA) in 2011 and 2014. Using R and SAS Enterprise Miner tools, we perform market basket analysis on the average of the total pollutant concentrations in a tract of a county within the state of California with the goal to identify a subsequent second or third pollutant presence upon an existing pollutant in a county. To proceed with the data mining process, we could use any of the data mining algorithms like SEMMA or CRISP-DM. We used CRISP-DM methodology to perform data mining for the association rules mining task. An R markdown file has been used to prepare the association – market basket analysis with the R Shiny presentation rendering an HTML output. The business or operational understanding and data understanding is performed as part of the requirements gathering. Later we import the dataset and clean it as part of data preparation. To optimize the functionality and reduce the redundancy, we use R functions enabling to break down or decompose a problem into smaller chunks. In addition, the code can be reproducible and reusable, and it was prepared in a systematic, organised, robust and an efficient manner. 'arules' and 'arulesviz' libraries were used to perform this data mining task. We first train the model and discover the fitting, and if necessary, we perform tuning operation to improve the results. To identify best results, we use the parameters like support, confidence, lift. Plots derived from ggplot2 were used wherever necessary to showcase the quality and aesthetics within the graphs. We later utilise SAS Enterprise Miner as a secondary data mining tool where we produce process flow diagrams and parameterize the tasks and compare the results with R.

2.2 Introduction

In 2011 and 2014, the US government agency of the United States Environmental Protection Agency (EPA) assessed the national air toxicity and released a dataset to the public, and this study is titled as the National Air Toxicity Assessment (NATA). EPA developed NATA as a screening tool for state, local and tribal air agencies. NATA's results help these agencies identify which pollutants, emission sources and places they may wish to study further to better understand any possible risks to public health from air toxics. There is now enough evidence that pollutants like acetaldehyde, benzene, cyanide, particulate matter components of diesel engine emissions (namely, diesel PM), toluene, and 1,3-butadiene have been proved to be the root cause for cancer across a wide scale of patients.

Air quality specialists use NATA results to learn which air toxics and emission source types may raise health risks in certain places. They can then study these places in more detail, focusing where the risks to people may be highest. NATA uses a 4-step methodology to develop the assessment:

1. Compile a national emissions inventory of outdoor air toxics sources.
2. Estimate ambient concentrations of air toxics across the United States.
3. Estimate population exposures across the United States.
4. Determine potential public health risks from breathing air toxics.

In this task, we are going to try associating the pollutants within the state of California, USA to create a model that can identify a concomitant pattern.

2.2.1 Brief background of the task

Below is the table showing the variables used from the dataset to create the association rules.

S.No	Variable Name	Type	Brief Description
1	Pollutant Name	*Identifier*	Name of the toxic chemical released into the air

2	Total_conc	*Predictor*	Total average concentration of a given chemical from all source types (air concentrations in µg/m3)

2.2.2 Formulation of the research question

The pollutants are in a single column where are the total concentration of those pollutants is in another column, hence we pivot this data into various columns by each census tract within the existing counties of the state of California. The cause of one pollutant can give rise to another pollutant due to various reasons like elemental conversions, competitive markets, crony capitalism, lack of awareness or even a political bias within a given county.

2.2.3 Justification: Why did I choose this topic/dataset?

According to the American Association for Cancer Research, new study suggests that air pollution is also associated with increased risk of mortality for several other types of cancer, including breast, liver, and pancreatic cancer.

Following heart disease, cancer is the second leading cause of death in the United States and around the world. In 2018, an estimated 9.5 million people died of cancer worldwide. That's about 26,000 people each day and 1 out of every 6 deaths. About 600,000 cancer deaths happen in the U.S. each year and about 80,000 in Canada. The rest happen in countries all around the world. About 7 out of every 10 deaths from the disease happen in low- or middle-income countries.

Cancers develop when something goes wrong in the DNA of a cell. Studying the DNA of people who develop cancer, and of those who don't, can be key in identifying people with a particularly high risk. It also helps in the search for new drugs and in choosing the best treatments for patients.

2.3 Aim and Objective of the task

Over the years, there have been various toxic pollutants which have caused Cancer among the people across the counties of California, USA. The objective of this task is to be able to identify if there is high risk of multiple carcinogens in association with each other in each of the tract in California, USA.

2.4 Brief Literature Review

Association Rule Mining is commonly used for market basket analysis to determine the likely combinations of items that will appeal to a consumer based on prior records. It has also been used to create predictive association rules for classification problems. The information collected using Association Rule Discovery technique also helps the companies in making decisions, forecasting sales, determining frauds etc. Also, it is widely used in medicine, biology, and several business areas such as telecommunication networks, market and risk management, inventory control etc. As stated above, Association rule mining aims to identify interesting correlations, frequent patterns, associations, or casual structures among a set of items in a transactional database or various data repositories.

The 3 important measures for association rules mining are support and confidence.

1. Support – Support is an indication of how frequently the items appear in the data.
2. Confidence – Confidence indicates the number of times the if-then statements are found true.
3. Lift - the ratio of the confidence of the rule and the expected confidence of the rule.

Apriori is an algorithm for frequent item set mining and association rule learning over relational databases. The Apriori algorithm takes advantage of the fact that any subset of a frequent itemset is also a frequent itemset. The algorithm can, therefore, reduce the number of candidates being considered by only exploring the item sets whose support count is greater than the minimum support count. Market basket analysis is a data mining technique used by retailers to increase sales by better understanding customer purchasing patterns.

2.5 Explanation and preparation of datasets

2.5.1 Description of the dataset

- **Data Source** - Link
- **Data File (in .xlsx format)** - Link (Size – 193 MB)

- The dataset contains various exposure concentrations, hazard indices, average cancer risk per million, and the population - all grouped by state, EPA region, county, tract, FIPS, and the pollutant. To deal with performance issues with R, we will take only the state of California.

- **Categorical Variables** -
  - State: State in the United States
  - EPA Region: EPA has ten regional offices, each of which is responsible for the execution of its programs within several states and territories.
  - County: A county is a political and administrative division of a state, providing certain local governmental services.
  - FIPS: FIPS (Federal Information Processing Standards) are a set of standards that describe document processing, encryption algorithms and other information technology standards for use within non-military government agencies and by government contractors and vendors who work with the agencies
  - Tract - Numeric code designating census tract from U.S. Census Bureau. Census tracts are Land areas defined by the U.S. Census Bureau. Tracts can vary in size but each typically contains about 4,000 residents. Census tracts are usually smaller than 2 square miles in cities but are much larger in rural areas.
  - Pollutant Name: Name of chemical

- **Continuous Variables** –
  - Population - Number of people in given census tract
  - Total_conc – Total average concentration of a given chemical from all source types (air concentrations in $\mu g/m^3$)

- **Environment setup :– The following libraries Installed and activated**
  - dplyr : Data manipulations
  - tidyverse : Data science tasks
  - readxl : to Import the .xlsx file
  - skimr : Statistical summary
  - corrplot : Correlation matrix
  - arulesViz : Assoc Rules - Visualization
  - arules : Association Rules
  - ggplot2 : Plotting graphs
  - RColorBrewer : Colour palette

Steps performed in R:

1. Setup the working directory using setwd(<filepath>).
2. Install the readxl package to import the dataset into R using read_excel() function, and view the top 6 rows of the dataset.

```
## Importing / Reading the data into "df_cea" data frame
df_cea_raw <- read_excel("ARM Dataset.xlsx", sheet = 1)

# Inspect the raw data
head(df_cea_raw)
```

3. Let us view the bottom 6 rows of the dataset.

```
tail(df_cea_raw)
```

```
## # A tibble: 6 x 5
##   State County       Tract Pollutant.Name Total.Conc
##   <chr> <chr>        <dbl> <chr>                <dbl>
## 1 WY    Sweetwater       0 TOLUENE              0.375
## 2 WY    Teton            0 TOLUENE              0.141
## 3 WY    Uinta            0 TOLUENE              0.426
## 4 WY    Washakie         0 TOLUENE              0.303
## 5 WY    Weston           0 TOLUENE              0.140
## 6 WY    Entire state     0 TOLUENE              0.611
```

4. Identify the column names

```
names(df_cea_raw)
```

```
## [1] "State"          "County"         "Tract"          "Pollutant.Name"
## [5] "Total.Conc"
```

5. Summarize the dataset to view the concentrations of pollutants in California.

```
summary(df_cea_raw)
```

```
##     State              County              Tract           Pollutant.Name
##  Length:464075      Length:464075      Min.   :0.000e+00   Length:464075
##  Class :character   Class :character   1st Qu.:1.208e+10   Class :character
##  Mode  :character   Mode  :character   Median :2.616e+10   Mode  :character
##                                        Mean   :2.720e+10
##                                        3rd Qu.:4.102e+10
##                                        Max.   :7.803e+10
##
##    Total.Conc
##  Min.   : 0.00000
##  1st Qu.: 0.04734
##  Median : 0.54590
##  Mean   : 1.02659
##  3rd Qu.: 1.52100
##  Max.   :39.66581
```

6. Let us check the structure of the dataset.

```
str(df_cea_raw)
```

```
## tibble [464,075 x 5] (S3: tbl_df/tbl/data.frame)
##  $ State         : chr [1:464075] "AK" "AK" "AK" "AK" ...
##  $ County        : chr [1:464075] "Aleutians East Borough" "Aleutians West Census Area" "Aleutian
us Area" "Anchorage Municipality" ...
##  $ Tract         : num [1:464075] 2.01e+09 2.02e+09 2.02e+09 2.02e+09 2.02e+09 ...
##  $ Pollutant.Name: chr [1:464075] "ACETALDEHYDE" "ACETALDEHYDE" "ACETALDEHYDE" "ACETALDEHYDE" ...
##  $ Total.Conc    : num [1:464075] 0.00174 0.00016 0.00461 0.02869 0.07384 ...
```

7. Let us check the dimensionality.

```
dim(df_cea_raw)
```

```
## [1] 464075      5
```

8. Change the names to simplify tasks.

```
# Renaming the columns as part of simplification
names(df_cea_raw)[names(df_cea_raw) == 'State'] <-  'state'
names(df_cea_raw)[names(df_cea_raw) == 'County'] <-  'county'
names(df_cea_raw)[names(df_cea_raw) == 'Tract'] <-  'tract'
names(df_cea_raw)[names(df_cea_raw) == 'Pollutant.Name'] <-  'pollutant'
names(df_cea_raw)[names(df_cea_raw) == 'Total.Conc'] <-  'total_conc'
```

9. Data Preparation steps –
    - Let us subset to California state only,

- Aggregate the data using averages of total concentration of pollutants,
- Apply string manipulations,
- Pivot the data to show 6 new columns (pollutants) each having various air concentration levels.
- Convert each tract into a categorical variable as factor
- Remove the raw dataset which is now redundant.

```
# 4.1 Subset to state CA
df_cea <- select(filter(df_cea_raw,
                        state == "CA" &
                        tract != 0 & pollutant != '' )
                 ,c(tract, pollutant, total_conc))

# 4.2 Aggregate data using averages
df_cea_agg <- df_cea %>%
              group_by(tract, pollutant) %>%
              summarise_at(vars(c(1)), list(avg = mean))

# 4.3 String manipulations within Pollutant
df_cea_agg$pollutant <- str_to_title(df_cea_agg$pollutant)

# 4.4 Pivot pollutants and total conc columns
df_cea_piv <- df_cea_agg %>%
  pivot_wider(names_from = pollutant,
                      values_from = avg)

# 4.5 Convert tract into a categorical variable
df_cea_piv$tract <- as.factor(df_cea_piv$tract)

# 4.5 Drop the raw dataframe
remove(df_cea_raw)
```

10. Inspect the modified dataset –

```
# Inspect the cleansing
# Describing the data subset
head(df_cea_piv)
```

```
## # A tibble: 6 x 7
## # Groups:   tract [6]
##   tract      `1,3-Butadiene` Acetaldehyde Benzene `Cyanide Compound~ `Diesel Pm`
##   <fct>                <dbl>        <dbl>   <dbl>              <dbl>       <dbl>
## 1 6001400100          0.0516         1.12   0.480            0.00290       0.793
## 2 6001400200          0.105          1.34   0.965            0.00326       2.37
## 3 6001400300          0.104          1.34   0.956            0.00331       2.33
## 4 6001400400          0.0999         1.31   0.915            0.00350       1.55
## 5 6001400500          0.0902         1.27   0.838            0.00373       1.34
## 6 6001400600          0.106          1.38   0.953            0.00359       2.01
## # ... with 1 more variable: Toluene <dbl>
```

```
tail(df_cea_piv)
```

```
## # A tibble: 6 x 7
## # Groups:   tract [6]
##   tract      `1,3-Butadiene` Acetaldehyde Benzene `Cyanide Compound~ `Diesel Pm`
##   <fct>                <dbl>        <dbl>   <dbl>              <dbl>       <dbl>
## 1 6115040700          0.0350         2.19   0.414            0.00266       0.332
## 2 6115040800          0.0255         2.17   0.332            0.000995      0.255
## 3 6115040901          0.0216         2.19   0.273            0.000960      0.188
## 4 6115040902          0.0212         2.16   0.283            0.00128       0.173
## 5 6115041000          0.0187         2.31   0.253            0.000696      0.173
## 6 6115041100          0.0143         2.23   0.170            0.000257      0.0696
## # ... with 1 more variable: Toluene <dbl>
```

```
names(df_cea_piv)
```

```
## [1] "tract"              "1,3-Butadiene"     "Acetaldehyde"
## [4] "Benzene"            "Cyanide Compounds" "Diesel Pm"
## [7] "Toluene"
```

```
dim(df_cea_piv)
```

```
## [1] 8177    7
```

11. Summary of each pollutant and structure of the dataset is as shown below

```
summary(df_cea_piv)
```

```
##         tract      1,3-Butadiene      Acetaldehyde      Benzene
##  6001400100:   1   Min.   :0.00000   Min.   :0.000   Min.   :0.0000
##  6001400200:   1   1st Qu.:0.03514   1st Qu.:1.377   1st Qu.:0.4690
##  6001400300:   1   Median :0.05926   Median :1.726   Median :0.6846
##  6001400400:   1   Mean   :0.06361   Mean   :1.682   Mean   :0.6973
##  6001400500:   1   3rd Qu.:0.08328   3rd Qu.:1.973   3rd Qu.:0.8495
##  6001400600:   1   Max.   :0.48383   Max.   :3.127   Max.   :4.4989
##  (Other)    :8171   NA's   :120       NA's   :120     NA's   :120
##  Cyanide Compounds     Diesel Pm          Toluene
##  Min.   :0.0000000   Min.   :0.01124   Min.   : 0.000
##  1st Qu.:0.0000007   1st Qu.:0.49677   1st Qu.: 1.509
##  Median :0.0002866   Median :0.88782   Median : 2.574
##  Mean   :0.0041891   Mean   :1.00194   Mean   : 2.728
##  3rd Qu.:0.0041889   3rd Qu.:1.35671   3rd Qu.: 3.687
##  Max.   :0.3525916   Max.   :8.83122   Max.   :16.180
##  NA's   :1           NA's   :154       NA's   :120
```

```
str(df_cea_piv)
```

```
## grouped_df [8,177 x 7] (S3: grouped_df/tbl_df/tbl/data.frame)
##  $ tract            : Factor w/ 8177 levels "6001400100","6001400200",..: 1 2 3 4 5 6 7
##  $ 1,3-Butadiene    : num [1:8177] 0.0516 0.1051 0.1036 0.0999 0.0902 ...
##  $ Acetaldehyde     : num [1:8177] 1.12 1.34 1.34 1.31 1.27 ...
##  $ Benzene          : num [1:8177] 0.48 0.965 0.956 0.915 0.838 ...
##  $ Cyanide Compounds: num [1:8177] 0.0029 0.00326 0.00331 0.0035 0.00373 ...
##  $ Diesel Pm        : num [1:8177] 0.793 2.373 2.328 1.555 1.335 ...
##  $ Toluene          : num [1:8177] 1.68 3.57 3.59 3.39 3.24 ...
##  - attr(*, "groups")= tibble [8,177 x 2] (S3: tbl_df/tbl/data.frame)
##   ..$ tract: Factor w/ 8177 levels "6001400100","6001400200",..: 1 2 3 4 5 6 7 8 9 10 ...
##   ..$ .rows: list<int> [1:8177]
```

12. Replace NAs with 0s

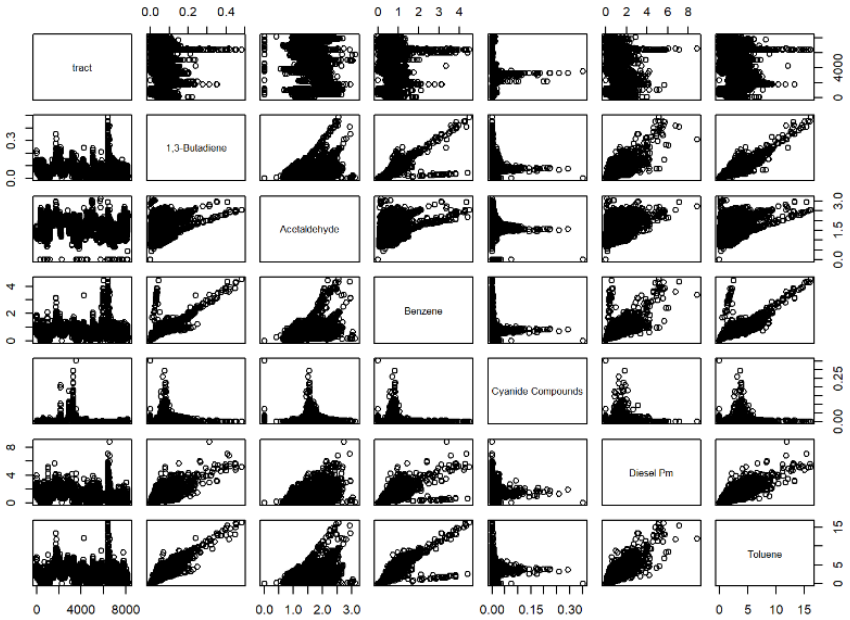
```
# Replace NAs with 0
df_cea_piv[is.na(df_cea_piv)] <- 0
summary(df_cea_piv)
```

```
##         tract      1,3-Butadiene      Acetaldehyde      Benzene
##  6001400100:   1   Min.   :0.00000   Min.   :0.000   Min.   :0.0000
##  6001400200:   1   1st Qu.:0.03399   1st Qu.:1.359   1st Qu.:0.4572
##  6001400300:   1   Median :0.05856   Median :1.717   Median :0.6795
##  6001400400:   1   Mean   :0.06268   Mean   :1.657   Mean   :0.6870
##  6001400500:   1   3rd Qu.:0.08272   3rd Qu.:1.969   3rd Qu.:0.8465
##  6001400600:   1   Max.   :0.48383   Max.   :3.127   Max.   :4.4989
##  (Other)    :8171
##  Cyanide Compounds     Diesel Pm          Toluene
##  Min.   :0.0000000   Min.   :0.0000   Min.   : 0.000
##  1st Qu.:0.0000006   1st Qu.:0.4738   1st Qu.: 1.457
##  Median :0.0002861   Median :0.8701   Median : 2.543
##  Mean   :0.0041886   Mean   :0.9831   Mean   : 2.688
##  3rd Qu.:0.0041845   3rd Qu.:1.3477   3rd Qu.: 3.668
##  Max.   :0.3525916   Max.   :8.8312   Max.   :16.180
##
```

2.5.2 Identify independent dependent variables (if any)

1. Correlation analysis using pairs()

```
# Pairs plot - Correlation plot
df_cea_piv$tract <- as.factor(df_cea_piv$tract)
pairs(df_cea_piv)
```



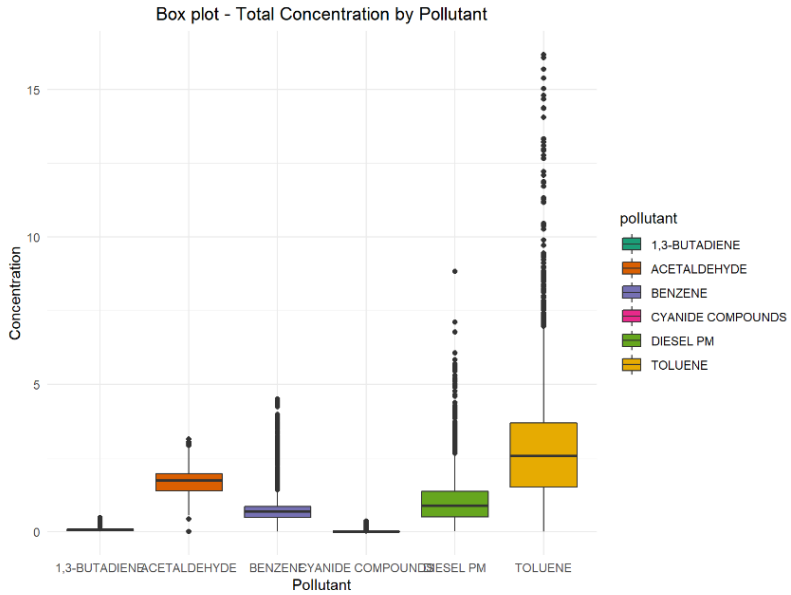
2. Correlation analysis – Correlation plot

```
rownames(df_cea_piv) <- df_cea_piv$tract

corrmatrix <- cor(df_cea_piv[,2:7])
corrplot(corrmatrix, method = 'number')
```

3. Creating a box plot to identify total distributions of concentration by Pollutant

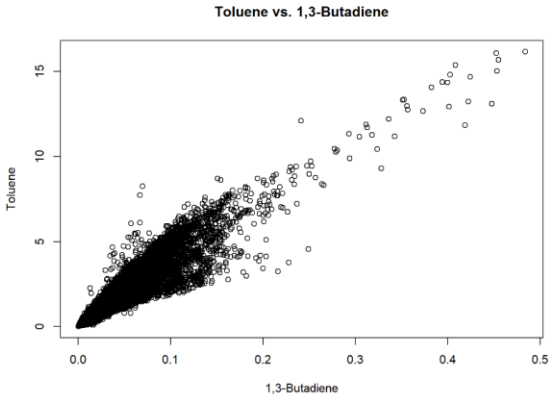
2. Box plot to check pollutants

```
# 5.3 - Boxplot - Total Concentration by Pollutant
ggplot(df_cea, aes(x=pollutant, y=total_conc, fill=pollutant)) +
  geom_boxplot()+
  labs(title="Box plot - Total Concentration by Pollutant",x="Pollutant", y = "Concentration")+ scale_fill_brewer(palette="Dark2") + theme_minimal
  theme(plot.title = element_text(hjust = 0.5))
```



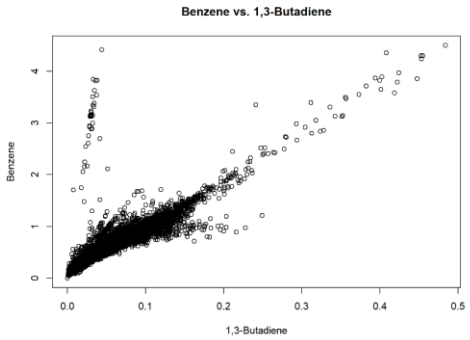
4. Plotting based on the observations –
    - Toulene vs. 1,3-Butadiene

```
# Toluene vs. 1,3-Butadiene
plot(Toluene ~ `1,3-Butadiene`, data = df_cea_piv, main = "Toluene vs. 1,3-Butadiene")
```



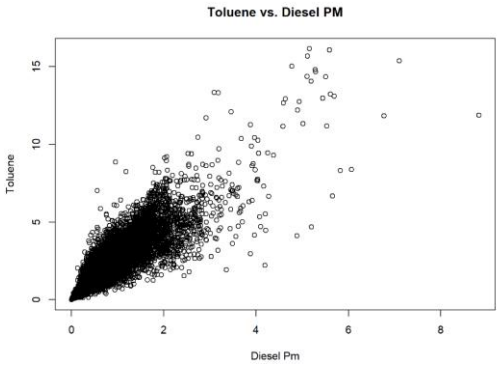
- Benzene vs. 1,3-Butadiene

```
# Benzene vs. 1,3-Butadiene
plot(Benzene ~ `1,3-Butadiene`, data = df_cea_piv, main = "Benzene vs. 1,3-Butadiene")
```



- Toulene vs. Diesel PM

```
# Toluene vs. Diesel PM
plot(Toluene ~ `Diesel Pm`, data = df_cea_piv, main = "Toluene vs. Diesel PM")
```

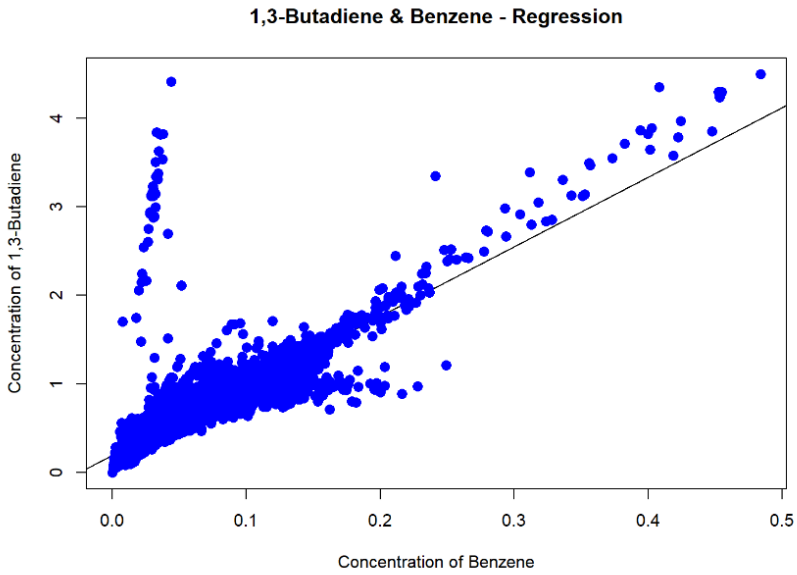


5. Apply Linear Regression on 1,3-Butadiene & Benzene

```
# Apply the lm() function.
relation <- lm(df_cea_piv$`1,3-Butadiene` ~ df_cea_piv$Benzene)

print(summary(relation))
```

```
##
## Call:
## lm(formula = df_cea_piv$`1,3-Butadiene` ~ df_cea_piv$Benzene)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.35588 -0.00783 -0.00088  0.00725  0.13917
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.0004434  0.0005072   0.874    0.382
## df_cea_piv$Benzene 0.0905868  0.0006383 141.913   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02305 on 8175 degrees of freedom
## Multiple R-squared:  0.7113, Adjusted R-squared:  0.7112
## F-statistic: 2.014e+04 on 1 and 8175 DF,  p-value: < 2.2e-16
```

6. Create the plot for linear regression –



2.5.3 Data Pre-processing steps

1. Create benchmark variables which is the average of the column (pollutant), based on which high or low concentration is designed.

```
# Create Benchmark Variables to classify the variables
# Anything greater than avg, can be considered as High or else Low conc
toluene_50 <- mean(df_cea_piv$Toluene)
butadiene_50 <- mean(df_cea_piv$`1,3-Butadiene`)
acetaldehyde_50 <- mean(df_cea_piv$Acetaldehyde)
benzene_50 <- mean(df_cea_piv$Benzene)
cyanide_50 <- mean(df_cea_piv$`Cyanide Compounds`)
diesel_50 <- mean(df_cea_piv$`Diesel Pm`)

# Convert continuous values into dichotomous variables (High/Low)
df_cea_piv <- df_cea_piv %>% mutate(Toluene = ifelse(Toluene > toluene_50, "H", "L"))
df_cea_piv <- df_cea_piv %>% mutate(`1,3-Butadiene` = ifelse(`1,3-Butadiene` > butadiene_50, "H", "L"))
df_cea_piv <- df_cea_piv %>% mutate(Acetaldehyde = ifelse(Acetaldehyde > acetaldehyde_50, "H", "L"))
df_cea_piv <- df_cea_piv %>% mutate(Benzene = ifelse(Benzene > benzene_50, "H", "L"))
df_cea_piv <- df_cea_piv %>% mutate(`Cyanide Compounds` = ifelse(`Cyanide Compounds` > cyanide_50, "H", "L"))
df_cea_piv <- df_cea_piv %>% mutate(`Diesel Pm` = ifelse(`Diesel Pm` > diesel_50, "H", "L"))

# Convert to factor
df_cea_piv$Toluene <- as.factor(df_cea_piv$Toluene)
df_cea_piv$`1,3-Butadiene` <- as.factor(df_cea_piv$`1,3-Butadiene`)
df_cea_piv$Acetaldehyde <- as.factor(df_cea_piv$Acetaldehyde)
df_cea_piv$Benzene <- as.factor(df_cea_piv$Benzene)
df_cea_piv$`Cyanide Compounds` <- as.factor(df_cea_piv$`Cyanide Compounds`)
df_cea_piv$`Diesel Pm` <- as.factor(df_cea_piv$`Diesel Pm`)

# Check frequency
as.data.frame(table(df_cea_piv$Toluene))
```

2. Inspect the frequency for each of the pollutant – observe the dichotomous nature.

```
# Check frequency
as.data.frame(table(df_cea_piv$Toluene))
```

```
##   Var1 Freq
## 1    H 3811
## 2    L 4366
```

```
as.data.frame(table(df_cea_piv$`1,3-Butadiene`))
```

```
##   Var1 Freq
## 1    H 3714
## 2    L 4463
```

```
as.data.frame(table(df_cea_piv$Acetaldehyde))
```

```
##   Var1 Freq
## 1    H 4445
## 2    L 3732
```

```
as.data.frame(table(df_cea_piv$Benzene))
```

```
##   Var1 Freq
## 1    H 4002
## 2    L 4175
```

```
as.data.frame(table(df_cea_piv$`Cyanide Compounds`))
```

```
##   Var1 Freq
## 1    H 2044
## 2    L 6133
```

```
as.data.frame(table(df_cea_piv$`Diesel Pm`))
```

```
##   Var1 Freq
## 1    H 3580
## 2    L 4597
```

3. Skim the dataset now

```
# Move these dichotonomous variables into a DF
df_cea_arm <- df_cea_piv[2:7]
skim(df_cea_arm)
```

Data summary

Name	df_cea_arm
Number of rows	8177
Number of columns	6
Column type frequency:	
factor	6
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
1,3-Butadiene	0	1	FALSE	2	L: 4463, H: 3714
Acetaldehyde	0	1	FALSE	2	H: 4445, L: 3732
Benzene	0	1	FALSE	2	L: 4175, H: 4002
Cyanide Compounds	0	1	FALSE	2	L: 6133, H: 2044
Diesel Pm	0	1	FALSE	2	L: 4597, H: 3580
Toluene	0	1	FALSE	2	L: 4366, H: 3811

4. Analyzing the custom-defined concentration levels –

2. Analyzing the custom-defined concentration levels

```
# Concentration_Tracts
#colSums() function computes the sums of columns.
high <- colSums(df_cea_arm == "H")
high
```

```
##      1,3-Butadiene       Acetaldehyde            Benzene Cyanide Compounds
##               3714               4445               4002               2044
##         Diesel Pm            Toluene
##               3580               3811
```

```
low <- colSums(df_cea_arm == "L")
low
```

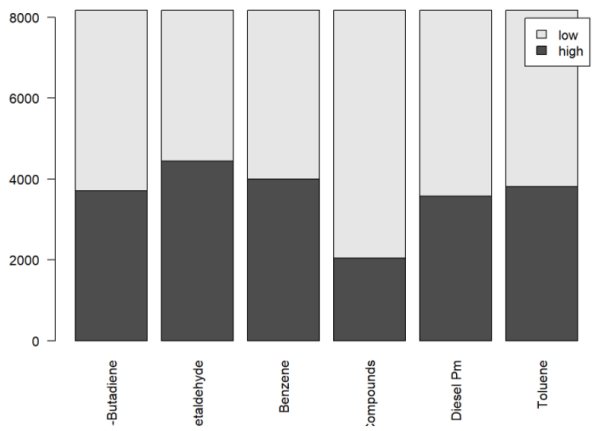
```
##      1,3-Butadiene       Acetaldehyde            Benzene Cyanide Compounds
##               4463               3732               4175               6133
##         Diesel Pm            Toluene
##               4597               4366
```

```
df_conc <- rbind(high,low)
df_conc
```

```
##      1,3-Butadiene Acetaldehyde Benzene Cyanide Compounds Diesel Pm Toluene
## high          3714         4445    4002               2044      3580    3811
## low           4463         3732    4175               6133      4597    4366
```

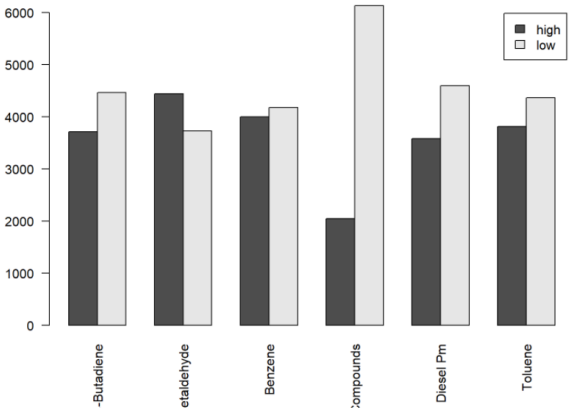
5. Create a stacked bar to observe the high + low pollutant concentration (100%)

```
barplot(df_conc,legend=rownames(df_conc),las=2) #Plot 1
```



6. Create a clustered bar chart to compare high and low pollutant concentrations .

```
barplot(df_conc, beside=T,legend=rownames(df_conc),las=2)# Plot 2
```



7. Unpivot & save the cleaned dataset (.csv) to work in SAS later

3. Unpivot & Saving the cleaned dataset as a .csv (for SAS)

```
# 3.1 - Unpivot the Pollutants and risks
df_cea_unpiv <- df_cea_piv %>%
  pivot_longer(!tract, names_to = "Pollutant", values_to = "Risk Status")

df_cea_unpiv$Potent_Risk <- ifelse(df_cea_unpiv$`Risk Status` == 'L',
                                   "", df_cea_unpiv$Pollutant)

head(df_cea_unpiv)
```

```
## # A tibble: 6 x 4
## # Groups:   tract [1]
##    tract      Pollutant         `Risk Status` Potent_Risk
##    <fct>      <chr>             <fct>         <chr>
## 1 6001400100 1,3-Butadiene     L             ""
## 2 6001400100 Acetaldehyde      L             ""
## 3 6001400100 Benzene           L             ""
## 4 6001400100 Cyanide Compounds L             ""
## 5 6001400100 Diesel Pm         L             ""
## 6 6001400100 Toluene           L             ""
```

8. Last few records

```
tail(df_cea_unpiv)
```

```
## # A tibble: 6 x 4
## # Groups:   tract [1]
##   tract      Pollutant         `Risk Status` Potent_Risk
##   <fct>      <chr>             <fct>         <chr>
## 1 6115041100 1,3-Butadiene     L             ""
## 2 6115041100 Acetaldehyde      H             "Acetaldehyde"
## 3 6115041100 Benzene           L             ""
## 4 6115041100 Cyanide Compounds L             ""
## 5 6115041100 Diesel Pm         L             ""
## 6 6115041100 Toluene           L             ""
```

9. Export everything to a .csv file

```
# 3.2 - Write to a .csv file
write.csv(df_cea_unpiv, "CancerRisk_Pollutants.csv", row.names=TRUE)
```

10. Validate the top few rows for Association Rules Mining

```
head(df_cea_arm)
```

```
## # A tibble: 6 x 6
##   `1,3-Butadiene` Acetaldehyde Benzene `Cyanide Compounds` `Diesel Pm` Toluene
##   <fct>           <fct>        <fct>   <fct>               <fct>       <fct>
## 1 L               L            L       L                   L           L
## 2 H               L            H       L                   H           H
## 3 H               L            H       L                   H           H
## 4 H               L            H       L                   H           H
## 5 H               L            H       L                   H           H
## 6 H               L            H       L                   H           H
```

11. Validate the dimensionality for Association Rules Mining

```
dim(df_cea_arm)
```

```
## [1] 8177    6
```

2.5.4 Assumptions (if any)

- It is assumed that if the total concentration of a pollutant in a given tract of a county is greater than the average value of all the tracts within California, then that tract is said to have High levels of toxic pollutants, and similarly if the total concentration of a pollutant in a given tract of a county is less than or equal to the average value of all the tracts within California, then that tract is said to have High levels of toxic pollutants.

2.6 Task: Association Rules

2.6.1 Data Exploration and Attribute Visualization in R

2.6.1.1 Model Building in R

1. Apply Apriori Algorithm – get all the rules (2453)

```
all_rules <- apriori(df_cea_arm,
                     parameter =list(minlen=2,maxlen=3,conf = 0.70, supp = 0.30))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.7    0.1    1 none FALSE            TRUE       5     0.3      2
##  maxlen target  ext
##       3  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 2453
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[12 item(s), 8177 transaction(s)] done [0.00s].
## sorting and recoding items ... [11 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [76 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

2. Summary of all rules

```
summary(all_rules)
```

```
## set of 76 rules
##
## rule length distribution (lhs + rhs):sizes
##  2  3
## 32 44
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   2.000   3.000   2.579   3.000   3.000
##
## summary of quality measures:
##     support         confidence        coverage           lift
##  Min.   :0.3199   Min.   :0.7011   Min.   :0.3447   Min.   :1.129
##  1st Qu.:0.3741   1st Qu.:0.8375   1st Qu.:0.4378   1st Qu.:1.490
##  Median :0.4182   Median :0.8790   Median :0.4677   Median :1.672
##  Mean   :0.4135   Mean   :0.8774   Mean   :0.4730   Mean   :1.634
##  3rd Qu.:0.4499   3rd Qu.:0.9337   3rd Qu.:0.5138   3rd Qu.:1.821
##  Max.   :0.5259   Max.   :0.9684   Max.   :0.7500   Max.   :2.054
##      count
##  Min.   :2616
##  1st Qu.:3059
##  Median :3420
##  Mean   :3381
##  3rd Qu.:3679
##  Max.   :4300
##
## mining info:
##         data ntransactions support confidence
##  df_cea_arm          8177     0.3        0.7
```

3. Inspect the rules – top 12

```
inspect(all_rules)
```

```
##      lhs                          rhs                      support   confidence coverage  lift     count
## [1]  {Diesel Pm=H}             => {1,3-Butadiene=H}      0.3447475 0.7874302  0.4378134 1.733661 2819
## [2]  {1,3-Butadiene=H}         => {Diesel Pm=H}          0.3447475 0.7590199  0.4542008 1.733661 2819
## [3]  {Diesel Pm=H}             => {Toluene=H}            0.3728751 0.8516760  0.4378134 1.827382 3049
## [4]  {Toluene=H}               => {Diesel Pm=H}          0.3728751 0.8000525  0.4660633 1.827382 3049
## [5]  {Diesel Pm=H}             => {Benzene=H}            0.3551425 0.8111732  0.4378134 1.657412 2904
## [6]  {Benzene=H}               => {Diesel Pm=H}          0.3551425 0.7256372  0.4894215 1.657412 2904
## [7]  {Diesel Pm=H}             => {Acetaldehyde=H}       0.3266479 0.7460894  0.4378134 1.372502 2671
## [8]  {1,3-Butadiene=H}         => {Toluene=H}            0.3866944 0.8513732  0.4542008 1.826733 3162
## [9]  {Toluene=H}               => {1,3-Butadiene=H}      0.3866944 0.8297035  0.4660633 1.826733 3162
## [10] {1,3-Butadiene=H}         => {Benzene=H}            0.4200807 0.9248788  0.4542008 1.889739 3435
## [11] {Benzene=H}               => {1,3-Butadiene=H}      0.4200807 0.8583208  0.4894215 1.889739 3435
## [12] {Acetaldehyde=L}          => {Diesel Pm=L}          0.3452366 0.7564309  0.4564021 1.345516 2823
## [13] {Acetaldehyde=L}          => {Cyanide Compounds=L} 0.3864498
```

4. Identify the rules where each of the pollutant's concentration is high with a 70% confidence interval and inspect the rules

- Acetaldehyde

```
rules <- apriori(df_cea_arm,
                 parameter =list(minlen=2,maxlen=3,conf = 0.70),
                 appearance = list(rhs=c("Acetaldehyde=H"),default="lhs"))
```

```
inspect(rules)
```

```
##      lhs                     rhs               support confidence  coverage     lift count
## [1]  {Cyanide Compounds=H} => {Acetaldehyde=H} 0.1800171  0.7201566 0.2499694 1.324796  1472
## [2]  {Diesel Pm=H}         => {Acetaldehyde=H} 0.3266479  0.7460894 0.4378134 1.372502  2671
## [3]  {Cyanide Compounds=H,
##       Diesel Pm=H}         => {Acetaldehyde=H} 0.1690106  0.7910704 0.2136480 1.455249  1382
## [4]  {1,3-Butadiene=H,
##       Cyanide Compounds=H} => {Acetaldehyde=H} 0.1709673  0.7508056 0.2277119 1.381178  1398
## [5]  {Cyanide Compounds=H,
##       Toluene=H}           => {Acetaldehyde=H} 0.1720680  0.7659227 0.2246545 1.408988  1407
## [6]  {Benzene=H,
##       Cyanide Compounds=H} => {Acetaldehyde=H} 0.1703559  0.7769102 0.2192736 1.429200  1393
## [7]  {1,3-Butadiene=H,
##       Diesel Pm=H}         => {Acetaldehyde=H} 0.2467898  0.7158567 0.3447475 1.316886  2018
## [8]  {Diesel Pm=H,
##       Toluene=H}           => {Acetaldehyde=H} 0.2818882  0.7559856 0.3728751 1.390707  2305
## [9]  {Benzene=H,
##       Diesel Pm=H}         => {Acetaldehyde=H} 0.2656231  0.7479339 0.3551425 1.375895  2172
## [10] {Cyanide Compounds=L,
##       Diesel Pm=H}         => {Acetaldehyde=H} 0.1576373  0.7032188 0.2241653 1.293638  1289
```

- Diesel PM

```
rules <- apriori(df_cea_arm,
                 parameter =list(minlen=2,maxlen=3,conf = 0.70),
                 appearance = list(rhs=c("Diesel Pm=H"),default="lhs"))
```

```
inspect(rules)
```

```
##      lhs                                    rhs           support
## [1]  {Cyanide Compounds=H}                  => {Diesel Pm=H} 0.2136480
## [2]  {1,3-Butadiene=H}                      => {Diesel Pm=H} 0.3447475
## [3]  {Toluene=H}                            => {Diesel Pm=H} 0.3728751
## [4]  {Benzene=H}                            => {Diesel Pm=H} 0.3551425
## [5]  {1,3-Butadiene=H,Cyanide Compounds=H}  => {Diesel Pm=H} 0.2066773
## [6]  {Cyanide Compounds=H,Toluene=H}        => {Diesel Pm=H} 0.2074110
## [7]  {Benzene=H,Cyanide Compounds=H}        => {Diesel Pm=H} 0.2005626
## [8]  {Acetaldehyde=H,Cyanide Compounds=H}   => {Diesel Pm=H} 0.1690106
## [9]  {1,3-Butadiene=H,Toluene=H}            => {Diesel Pm=H} 0.3279932
## [10] {1,3-Butadiene=H,Benzene=H}            => {Diesel Pm=H} 0.3293384
## [11] {1,3-Butadiene=H,Acetaldehyde=H}       => {Diesel Pm=H} 0.2467898
## [12] {Benzene=H,Toluene=H}                  => {Diesel Pm=H} 0.3399780
## [13] {Acetaldehyde=H,Toluene=H}             => {Diesel Pm=H} 0.2818882
## [14] {Acetaldehyde=H,Benzene=H}             => {Diesel Pm=H} 0.2656231
##      confidence coverage  lift     count
## [1]  0.8546967  0.2499694 1.952194 1747
## [2]  0.7590199  0.4542008 1.733661 2819
## [3]  0.8000525  0.4660633 1.827382 3049
## [4]  0.7256372  0.4894215 1.657412 2904
## [5]  0.9076262  0.2277119 2.073089 1690
## [6]  0.9232444  0.2246545 2.108762 1696
## [7]  0.9146682  0.2192736 2.089174 1640
## [8]  0.9388587  0.1800171 2.144427 1382
## [9]  0.8481973  0.3866944 1.937349 2682
## [10] 0.7839884  0.4200807 1.790691 2693
## [11] 0.8380399  0.2944845 1.914149 2018
## [12] 0.8183691  0.4154335 1.869219 2780
## [13] 0.8824655  0.3194326 2.015620 2305
## [14] 0.8162345  0.3254250 1.864343 2172
```

- 1,3-Butadiene

```
rules <- apriori(df_cea_arm,
                 parameter =list(minlen=2,maxlen=3,conf = 0.70),
                 appearance = list(rhs=c("1,3-Butadiene=H"),default="lhs"))
```

```
inspect(rules)
```

```
##      lhs                     rhs                 support confidence  coverage     lift count
## [1]  {Cyanide Compounds=H} => {1,3-Butadiene=H} 0.2277119  0.9109589 0.2499694 2.005630  1862
## [2]  {Diesel Pm=H}         => {1,3-Butadiene=H} 0.3447475  0.7874302 0.4378134 1.733661  2819
## [3]  {Toluene=H}           => {1,3-Butadiene=H} 0.3866944  0.8297035 0.4660633 1.826733  3162
## [4]  {Benzene=H}           => {1,3-Butadiene=H} 0.4200807  0.8583208 0.4894215 1.889739  3435
## [5]  {Cyanide Compounds=H,
##       Diesel Pm=H}         => {1,3-Butadiene=H} 0.2066773  0.9673726 0.2136480 2.129835  1690
## [6]  {Cyanide Compounds=H,
##       Toluene=H}           => {1,3-Butadiene=H} 0.2200073  0.9793141 0.2246545 2.156126  1799
## [7]  {Benzene=H,
##       Cyanide Compounds=H} => {1,3-Butadiene=H} 0.2176837  0.9927496 0.2192736 2.185706  1780
## [8]  {Acetaldehyde=H,
##       Cyanide Compounds=H} => {1,3-Butadiene=H} 0.1709673  0.9497283 0.1800171 2.090988  1398
## [9]  {Diesel Pm=H,
##       Toluene=H}           => {1,3-Butadiene=H} 0.3279932  0.8796327 0.3728751 1.936660  2682
## [10] {Benzene=H,
##       Diesel Pm=H}         => {1,3-Butadiene=H} 0.3293384  0.9273416 0.3551425 2.041700  2693
## [11] {Acetaldehyde=H,
##       Diesel Pm=H}         => {1,3-Butadiene=H} 0.2467898  0.7555223 0.3266479 1.663410  2018
## [12] {Acetaldehyde=L,
##       Toluene=H}           => {1,3-Butadiene=H} 0.1260854  0.8598832 0.1466308 1.893179  1031
## [13] {Acetaldehyde=L,
##       Benzene=H}           => {1,3-Butadiene=H} 0.1353797  0.8255034 0.1639966 1.817485  1107
## [14] {Benzene=H,
##       Toluene=H}           => {1,3-Butadiene=H} 0.3744650  0.9013836 0.4154335 1.984549  3062
## [15] {Acetaldehyde=H,
##       Toluene=H}           => {1,3-Butadiene=H} 0.2606090  0.8158499 0.3194326 1.796232  2131
## [16] {Acetaldehyde=H,
##       Benzene=H}           => {1,3-Butadiene=H} 0.2847010  0.8748591 0.3254250 1.926150  2328
## [17] {Benzene=H,
##       Cyanide Compounds=L} => {1,3-Butadiene=H} 0.2023970  0.7492078 0.2701480 1.649508  1655
```

- Cyanide Compounds

```
rules <- apriori(df_cea_arm,
                 parameter =list(minlen=2,maxlen=3,conf = 0.50),
                 appearance = list(rhs=c("Cyanide Compounds=H"),default="lhs"))
```

```
inspect(rules)
```

```
##      lhs                 rhs                     support confidence  coverage     lift count
## [1]  {1,3-Butadiene=H} => {Cyanide Compounds=H} 0.2277119  0.5013463 0.4542008 2.005630  1862
## [2]  {1,3-Butadiene=H,
##       Diesel Pm=H}     => {Cyanide Compounds=H} 0.2066773  0.5995034 0.3447475 2.398307  1690
## [3]  {Diesel Pm=H,
##       Toluene=H}       => {Cyanide Compounds=H} 0.2074110  0.5562480 0.3728751 2.225264  1696
## [4]  {Benzene=H,
##       Diesel Pm=H}     => {Cyanide Compounds=H} 0.2005626  0.5647383 0.3551425 2.259229  1640
## [5]  {Acetaldehyde=H,
##       Diesel Pm=H}     => {Cyanide Compounds=H} 0.1690106  0.5174092 0.3266479 2.069890  1382
## [6]  {1,3-Butadiene=H,
##       Toluene=H}       => {Cyanide Compounds=H} 0.2200073  0.5689437 0.3866944 2.276053  1799
## [7]  {1,3-Butadiene=H,
##       Benzene=H}       => {Cyanide Compounds=H} 0.2176837  0.5181951 0.4200807 2.073034  1780
## [8]  {1,3-Butadiene=H,
##       Acetaldehyde=H}  => {Cyanide Compounds=H} 0.1709673  0.5805648 0.2944845 2.322543  1398
## [9]  {Benzene=H,
##       Toluene=H}       => {Cyanide Compounds=H} 0.2141372  0.5154548 0.4154335 2.062071  1751
## [10] {Acetaldehyde=H,
##       Toluene=H}       => {Cyanide Compounds=H} 0.1720680  0.5386677 0.3194326 2.154934  1407
## [11] {Acetaldehyde=H,
##       Benzene=H}       => {Cyanide Compounds=H} 0.1703559  0.5234874 0.3254250 2.094206  1393
```

- Benzene –

```
rules <- apriori(df_cea_arm,
                 parameter =list(minlen=2,maxlen=3,conf = 0.70),
                 appearance = list(rhs=c("Benzene=H"),default="lhs"))
```

```
inspect(rules)
```

```
##      lhs                                      rhs          support   confidence
## [1]  {Cyanide Compounds=H}                 => {Benzene=H} 0.2192736 0.8772016
## [2]  {Diesel Pm=H}                         => {Benzene=H} 0.3551425 0.8111732
## [3]  {1,3-Butadiene=H}                     => {Benzene=H} 0.4200807 0.9248788
## [4]  {Toluene=H}                           => {Benzene=H} 0.4154335 0.8913671
## [5]  {Cyanide Compounds=H,Diesel Pm=H}     => {Benzene=H} 0.2005626 0.9387521
## [6]  {1,3-Butadiene=H,Cyanide Compounds=H} => {Benzene=H} 0.2176837 0.9559613
## [7]  {Cyanide Compounds=H,Toluene=H}       => {Benzene=H} 0.2141372 0.9531845
## [8]  {Acetaldehyde=H,Cyanide Compounds=H}  => {Benzene=H} 0.1703559 0.9463315
## [9]  {1,3-Butadiene=H,Diesel Pm=H}         => {Benzene=H} 0.3293384 0.9553033
## [10] {Diesel Pm=H,Toluene=H}               => {Benzene=H} 0.3399780 0.9117744
## [11] {Acetaldehyde=H,Diesel Pm=H}          => {Benzene=H} 0.2656231 0.8131786
## [12] {1,3-Butadiene=H,Acetaldehyde=L}      => {Benzene=H} 0.1353797 0.8476263
## [13] {1,3-Butadiene=H,Toluene=H}           => {Benzene=H} 0.3744650 0.9683744
## [14] {1,3-Butadiene=H,Acetaldehyde=H}      => {Benzene=H} 0.2847010 0.9667774
## [15] {1,3-Butadiene=H,Cyanide Compounds=L} => {Benzene=H} 0.2023970 0.8936285
## [16] {Acetaldehyde=L,Toluene=H}            => {Benzene=H} 0.1312217 0.8949124
## [17] {Acetaldehyde=H,Toluene=H}            => {Benzene=H} 0.2842118 0.8897397
## [18] {Cyanide Compounds=L,Toluene=H}       => {Benzene=H} 0.2012963 0.8338399
##      coverage  lift     count
## [1]  0.2499694 1.792323 1793
## [2]  0.4378134 1.657412 2904
## [3]  0.4542008 1.889739 3435
## [4]  0.4660633 1.821267 3397
## [5]  0.2136480 1.918085 1640
## [6]  0.2277119 1.953247 1780
## [7]  0.2246545 1.947574 1751
## [8]  0.1800171 1.933571 1393
## [9]  0.3447475 1.951903 2693
## [10] 0.3728751 1.862963 2780
## [11] 0.3266479 1.661510 2172
## [12] 0.1597163 1.731894 1107
## [13] 0.3866944 1.978610 3062
## [14] 0.2944845 1.975347 2328
## [15] 0.2264889 1.825887 1655
## [16] 0.1466308 1.828510 1073
## [17] 0.3194326 1.817941 2324
## [18] 0.2414088 1.703725 1646
```

- Toluene

```
rules <- apriori(df_cea_arm,
                 appearance = list(rhs=c("Toluene=H"),default="lhs"))
```

```
inspect(rules)
```

```
##      lhs                                rhs           support   confidence  coverage     lift count
## [1]  {Cyanide Compounds=H}           => {Toluene=H} 0.2246545  0.8987280 0.2499694 1.928339  1837
## [2]  {Diesel Pm=H}                   => {Toluene=H} 0.3728751  0.8516760 0.4378134 1.827382  3049
## [3]  {1,3-Butadiene=H}               => {Toluene=H} 0.3866944  0.8513732 0.4542008 1.826733  3162
## [4]  {Benzene=H}                     => {Toluene=H} 0.4154335  0.8488256 0.4894215 1.821267  3397
## [5]  {Cyanide Compounds=H,
##       Diesel Pm=H}                   => {Toluene=H} 0.2074110  0.9708071 0.2136480 2.082994  1696
## [6]  {1,3-Butadiene=H,
##       Cyanide Compounds=H}           => {Toluene=H} 0.2200073  0.9661654 0.2277119 2.073035  1799
## [7]  {Benzene=H,
##       Cyanide Compounds=H}           => {Toluene=H} 0.2141372  0.9765756 0.2192736 2.095371  1751
## [8]  {Acetaldehyde=H,
##       Cyanide Compounds=H}           => {Toluene=H} 0.1720680  0.9558424 0.1800171 2.050885  1407
## [9]  {1,3-Butadiene=H,
##       Diesel Pm=H}                   => {Toluene=H} 0.3279932  0.9514012 0.3447475 2.041356  2682
## [10] {Benzene=H,
##       Diesel Pm=H}                   => {Toluene=H} 0.3399780  0.9573003 0.3551425 2.054013  2780
## [11] {Acetaldehyde=H,
##       Diesel Pm=H}                   => {Toluene=H} 0.2818882  0.8629727 0.3266479 1.851621  2305
## [12] {1,3-Butadiene=H,
##       Benzene=H}                     => {Toluene=H} 0.3744650  0.8914119 0.4200807 1.912641  3062
## [13] {1,3-Butadiene=H,
##       Acetaldehyde=H}                => {Toluene=H} 0.2606090  0.8849668 0.2944845 1.898812  2131
## [14] {Acetaldehyde=L,
##       Benzene=H}                     => {Toluene=H} 0.1312217  0.8001491 0.1639966 1.716825  1073
## [15] {Acetaldehyde=H,
##       Benzene=H}                     => {Toluene=H} 0.2842118  0.8733559 0.3254250 1.873900  2324
## [16] {1,3-Butadiene=H,
##       Cyanide Compounds=H,
##       Diesel Pm=H}                   => {Toluene=H} 0.2043537  0.9887574 0.2066773 2.121509  1671
## [17] {Benzene=H,
##       Cyanide Compounds=H,
##       Diesel Pm=H}                   => {Toluene=H} 0.1986058  0.9902439 0.2005626 2.124698  1624
## [18] {Acetaldehyde=H,
##       Cyanide Compounds=H,
##       Diesel Pm=H}                   => {Toluene=H} 0.1682769  0.9956585 0.1690106 2.136316  1376
```

- Applying combination – 1,3-Butadiene and Diesel PM

```
rules3 <- apriori(df_cea_arm,
parameter = list(minlen=2,maxlen=3, conf = 0.70),
appearance = list(rhs=c("1,3-Butadiene=H","Diesel Pm=H" ), lhs = c("Acetaldehyde=H","Benzene=H","Toluene=H","Cyanide Compounds=H") , default="none"))
```

```
inspect(rules3)
```

```
##      lhs                      rhs                    support confidence  coverage     lift count
## [1]  {Cyanide Compounds=H} => {Diesel Pm=H}     0.2136480  0.8546967 0.2499694 1.952194  1747
## [2]  {Cyanide Compounds=H} => {1,3-Butadiene=H} 0.2277119  0.9109589 0.2499694 2.005630  1862
## [3]  {Toluene=H}           => {Diesel Pm=H}     0.3728751  0.8000525 0.4660633 1.827382  3049
## [4]  {Benzene=H}           => {Diesel Pm=H}     0.3551425  0.7256372 0.4894215 1.657412  2904
## [5]  {Toluene=H}           => {1,3-Butadiene=H} 0.3866944  0.8297035 0.4660633 1.826733  3162
## [6]  {Benzene=H}           => {1,3-Butadiene=H} 0.4200807  0.8583208 0.4894215 1.889739  3435
## [7]  {Acetaldehyde=H,
##       Cyanide Compounds=H} => {Diesel Pm=H}     0.1690106  0.9388587 0.1800171 2.144427  1382
## [8]  {Acetaldehyde=H,
##       Cyanide Compounds=H} => {1,3-Butadiene=H} 0.1709673  0.9497283 0.1800171 2.090988  1398
## [9]  {Cyanide Compounds=H,
##       Toluene=H}           => {Diesel Pm=H}     0.2074110  0.9232444 0.2246545 2.108762  1696
## [10] {Benzene=H,
##       Cyanide Compounds=H} => {Diesel Pm=H}     0.2005626  0.9146682 0.2192736 2.089174  1640
## [11] {Cyanide Compounds=H,
##       Toluene=H}           => {1,3-Butadiene=H} 0.2200073  0.9793141 0.2246545 2.156126  1799
## [12] {Benzene=H,
##       Cyanide Compounds=H} => {1,3-Butadiene=H} 0.2176837  0.9927496 0.2192736 2.185706  1780
## [13] {Acetaldehyde=H,
##       Toluene=H}           => {Diesel Pm=H}     0.2818882  0.8824655 0.3194326 2.015620  2305
## [14] {Acetaldehyde=H,
##       Benzene=H}           => {Diesel Pm=H}     0.2656231  0.8162345 0.3254250 1.864343  2172
## [15] {Acetaldehyde=H,
##       Toluene=H}           => {1,3-Butadiene=H} 0.2606090  0.8158499 0.3194326 1.796232  2131
## [16] {Acetaldehyde=H,
##       Benzene=H}           => {1,3-Butadiene=H} 0.2847010  0.8748591 0.3254250 1.926150  2328
## [17] {Benzene=H,
##       Toluene=H}           => {Diesel Pm=H}     0.3399780  0.8183691 0.4154335 1.869219  2780
## [18] {Benzene=H,
##       Toluene=H}           => {1,3-Butadiene=H} 0.3744650  0.9013836 0.4154335 1.984549  3062
```

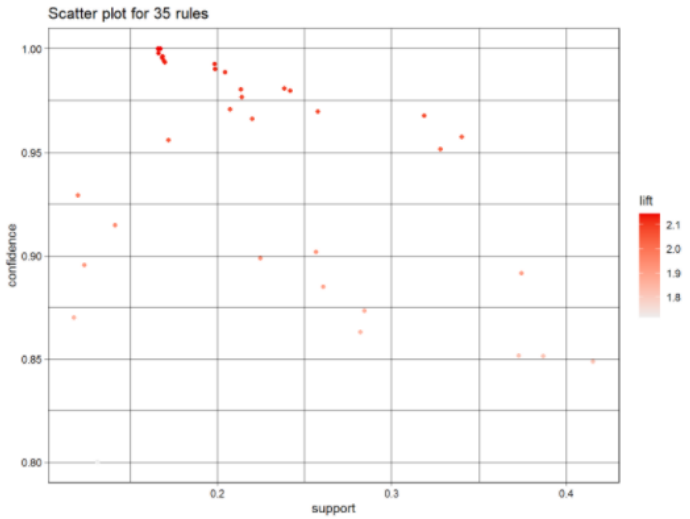
2.6.1.2 Model Assessment in R

1. For Model assessment – we could first apply the same methodology within ruleExplorer() and observe

2.6.1.3 Results visualisation in R

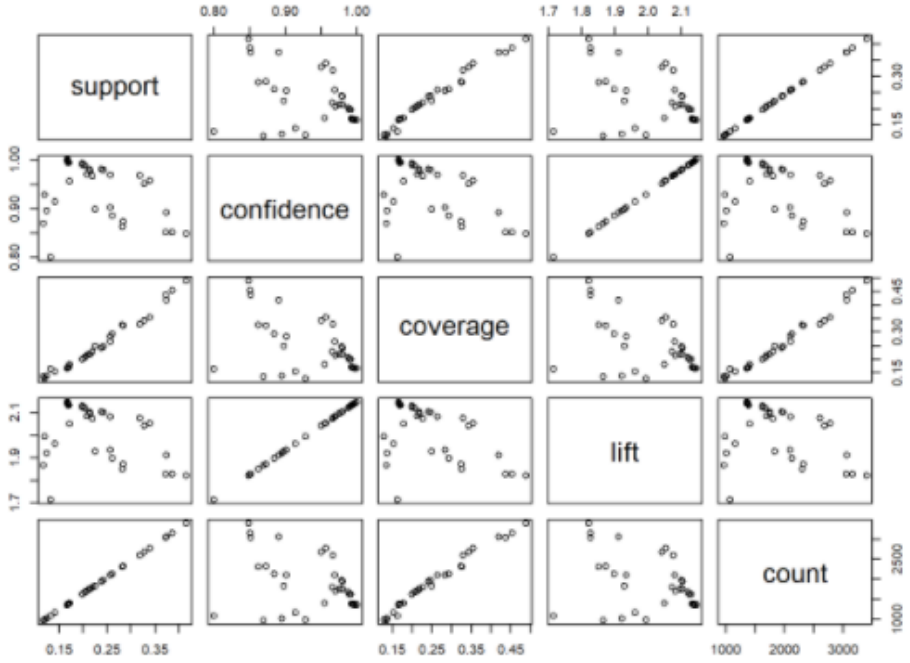
1. Plot the rules

```
# arulesViz - plots

plot(rules)
```

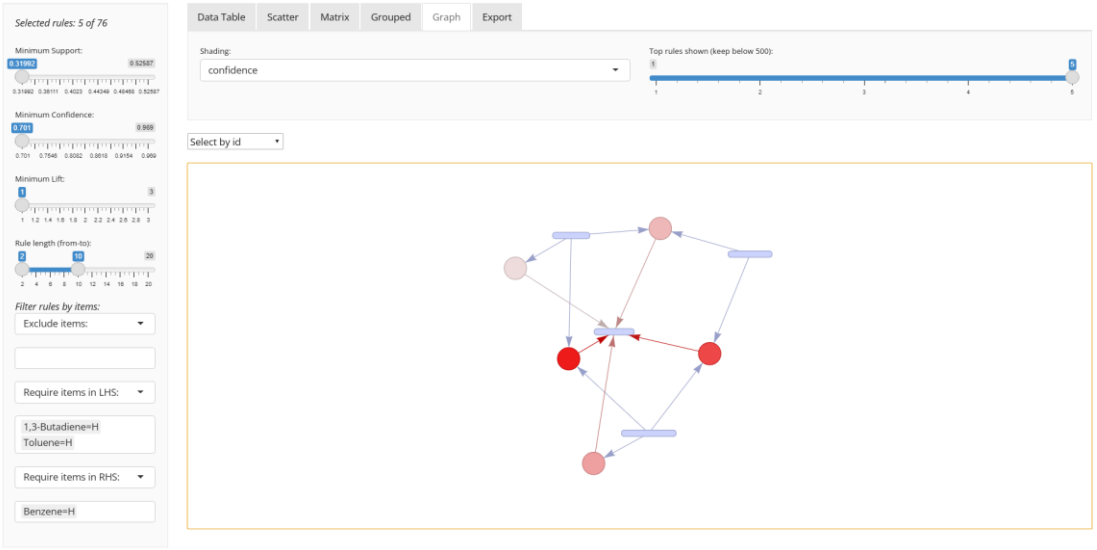


2. Plot the rules@quality

```
plot(rules@quality)
```



2. Association Rule Explorer findings



2.6.2 Data Exploration and Attribute Visualization in SAS EM

2.6.2.1 Model Building in SAS EM

1. File import into SAS EM. Score role is transaction.

Property	Value
General	
Node ID	FIMPORT
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Import File	D:\University\ASDM\Develo
Maximum Rows to Import	1000000
Maximum Columns to Import	10000
Delimiter	,
Name Row	Yes
Number of Rows to Skip	0
Guessing Rows	500
File Location	Local
File Type	csv
Advanced Advisor	No
Rerun	No
Score	
Role	Transaction
Report	
Summarize	No
Status	
Create Time	30/12/21 16:23
Run ID	3f0191ed-be39-4fda-82f0-01e
Last Error	
Last Status	Complete
Last Run Time	30/12/21 16:24
Run Duration	0 Hr. 0 Min. 3.02 Sec.
Grid Host	
User-Added Node	No

2. Model the variables – ID and Target variables.

Name	Role	Level	Report	Order
Pollutant	**Rejected**	**Nominal**	**No**	
Potent_Risk	**Target**	**Nominal**	**No**	
Risk_Status	**Rejected**	**Nominal**	**No**	
VAR1	**Rejected**	**Nominal**	**No**	
tract	**ID**	**Nominal**	**No**	**Ascending**

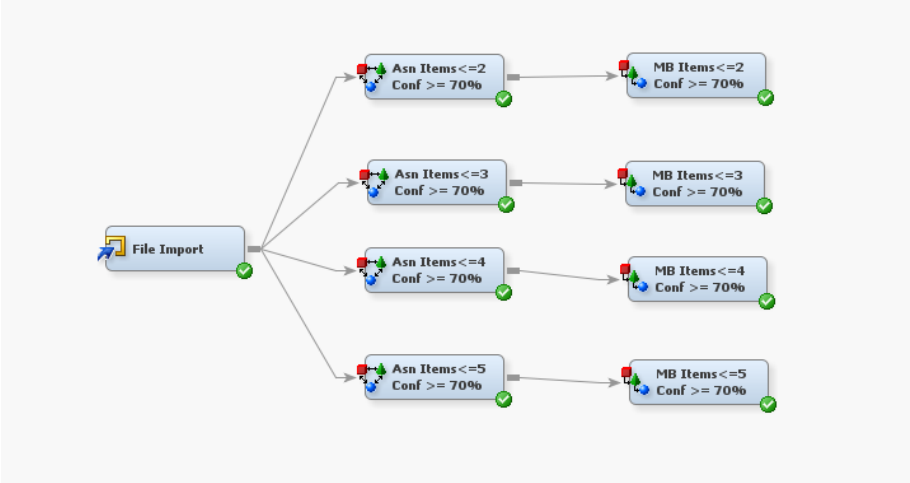
3. Apply association –

.. Property	Value
General	
Node ID	Assoc
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Maximum Number of Items to	100000
Rules	
Association	
Maximum Items	2
Minimum Confidence Level	70
Support Type	Percent
Support Count	.
Support Percentage	5.0
Sequence	
Chain Count	3
Consolidate Time	0.0
Maximum Transaction Duratio	0.0
Support Type	Percent
Support Count	.
Support Percentage	2.0
Rules	
Number to Keep	200
Sort Criterion	Default
Number to Transpose	200
Export Rule by ID	No
Recommendation	No
Status	
Create Time	30/12/21 16:24
Run ID	3b00612a-60ac-413a-8f3b-3b
Last Error	
Last Status	Complete
Last Run Time	30/12/21 18:51
Run Duration	0 Hr. 0 Min. 2.38 Sec.
Grid Host	
User-Added Node	No

4. Market Basket –

.. Property	Value
General	
Node ID	MRKBSKT
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Normalize	No
Constraints	
Maximum Items	2
Minimum Confidence Level	70
Minimum Lift	.
Minimum Support Lift	.
Support Type	Percent
Support Count	5
Support Percentage	2.0
Hierarchy	
Dimension Data Set	
Mapping	
Basket Size Options	
Minimum Size	1
Maximum Size	1000
Rules	
Maximum Number of Rules	100000
Number to Keep	1000
Sort Criterion	Support
Score	
Rules	
Export Rule by ID	No
Transpose Selection	Automatic
Number to Transpose	200
Rules	
Recommendation	No
Status	
Create Time	30/12/21 18:57
Run ID	6e9656a1-6dd8-4cc8-9901-
Last Error	
Last Status	Complete
Last Run Time	30/12/21 19:01
Run Duration	0 Hr. 0 Min. 2.51 Sec.
Grid Host	

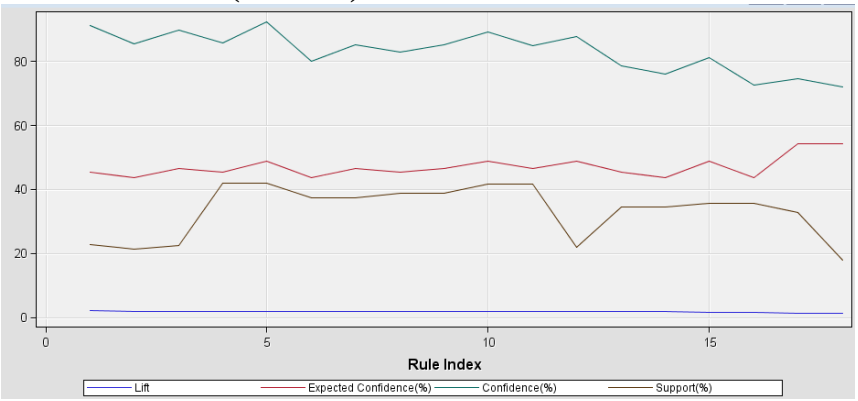
5. This is how the process flow diagram looks

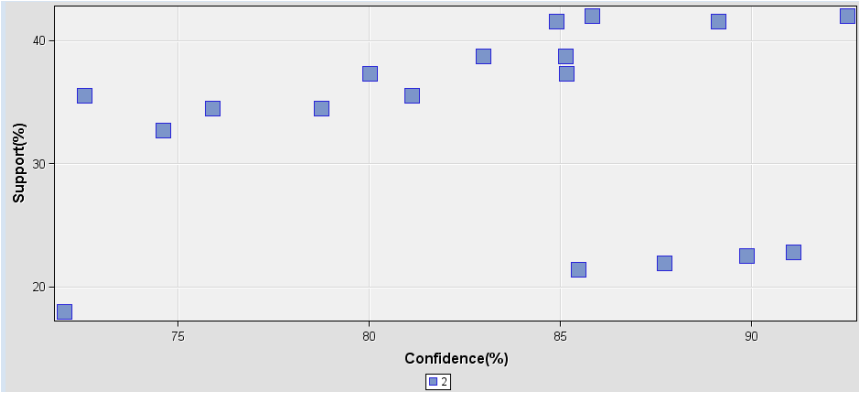


2.6.2.2 Model Assessment in SAS EM

2.6.2.3 Results visualisation in SAS EM

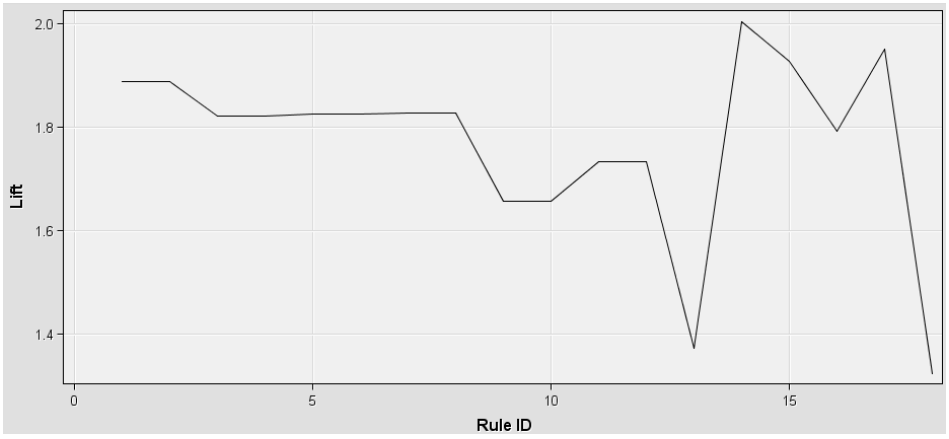
1. Statistics Line Plot (level = 1)



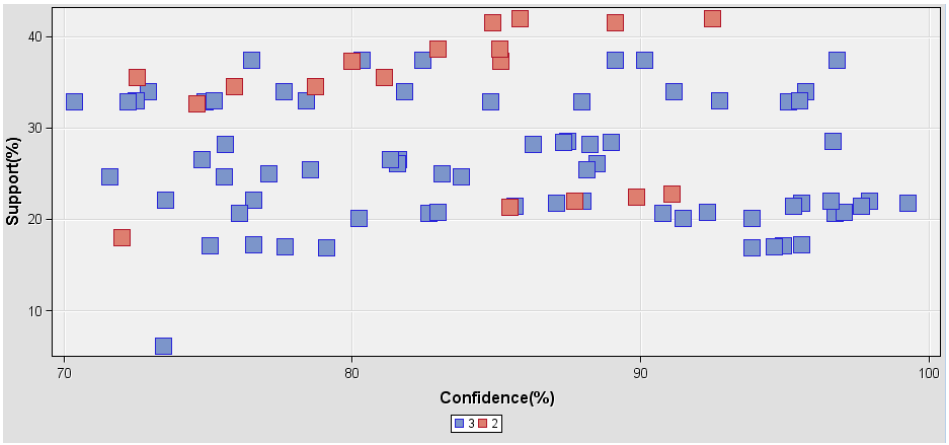


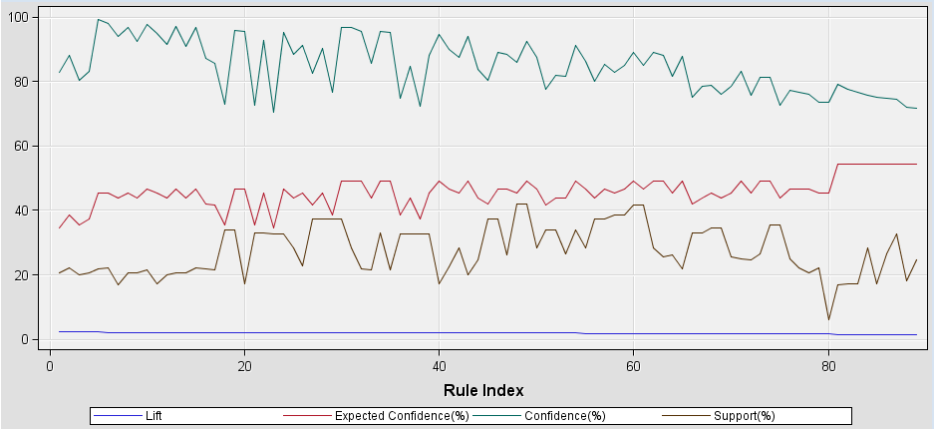
Item Name	Transaction Count	Support(%)
	6819	83.3924
1,3-Butadiene	3714	45.4201
Benzene	4002	48.9422
Diesel Pm	3580	43.7813
Toluene	3811	46.6063
Acetaldehyde	4445	54.3598
Cyanide Compoun	2044	24.9969

Confidence (%)	Support (%)	Lift	Transaction Count	Rule
91.10	22.77	2.01	1862.0	Cyanide Compoun ==> 1,3-Butadiene
85.47	21.36	1.95	1747.0	Cyanide Compoun ==> Diesel Pm
89.87	22.47	1.93	1837.0	Cyanide Compoun ==> Toluene
85.83	42.01	1.89	3435.0	Benzene ==> 1,3-Butadiene
92.49	42.01	1.89	3435.0	1,3-Butadiene ==> Benzene
80.01	37.29	1.83	3049.0	Toluene ==> Diesel Pm
85.17	37.29	1.83	3049.0	Diesel Pm ==> Toluene
82.97	38.67	1.83	3162.0	Toluene ==> 1,3-Butadiene
85.14	38.67	1.83	3162.0	1,3-Butadiene ==> Toluene
89.14	41.54	1.82	3397.0	Toluene ==> Benzene
84.88	41.54	1.82	3397.0	Benzene ==> Toluene
87.72	21.93	1.79	1793.0	Cyanide Compoun ==> Benzene
78.74	34.47	1.73	2819.0	Diesel Pm ==> 1,3-Butadiene
75.90	34.47	1.73	2819.0	1,3-Butadiene ==> Diesel Pm
81.12	35.51	1.66	2904.0	Diesel Pm ==> Benzene
72.56	35.51	1.66	2904.0	Benzene ==> Diesel Pm
74.61	32.66	1.37	2671.0	Diesel Pm ==> Acetaldehyde
72.02	18.00	1.32	1472.0	Cyanide Compoun ==> Acetaldehyde

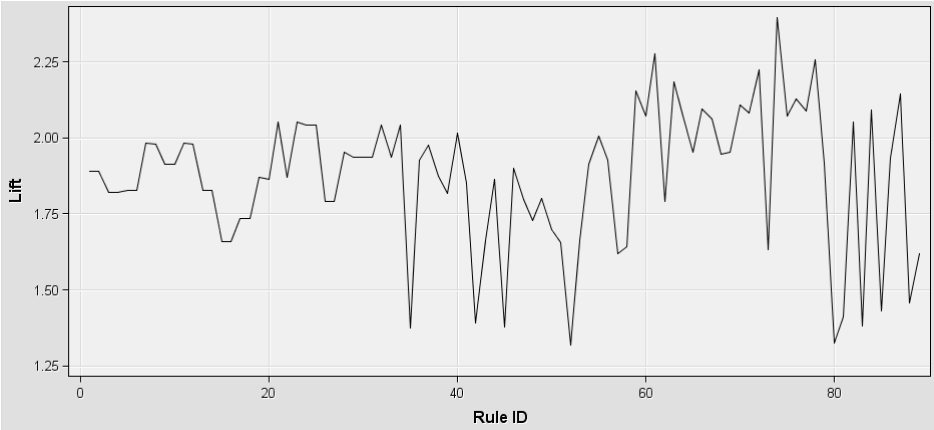


2. Level 2 -



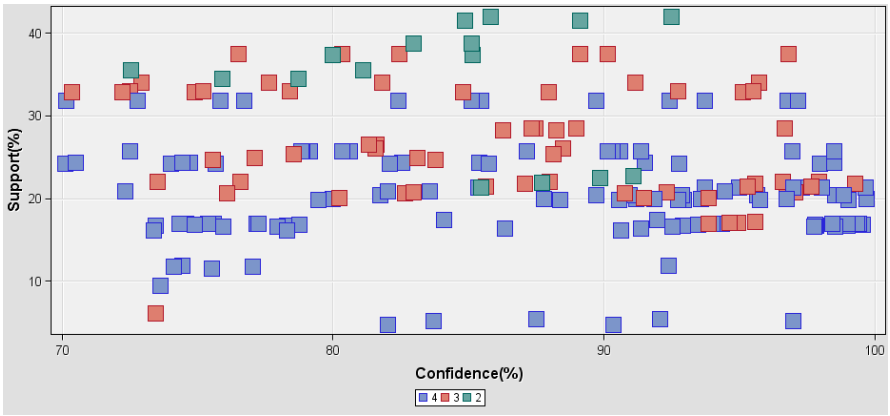


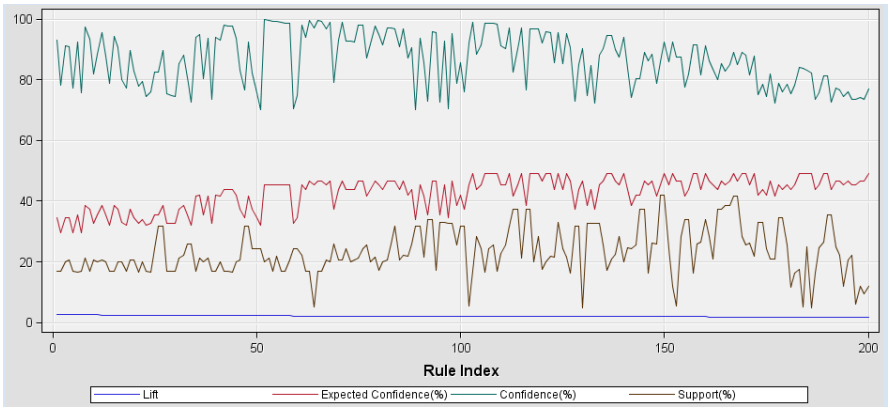
Expected Confidence (%)	Confidence (%)	Support (%)	Lift	Transaction Count	Rule
34.47	82.68	20.67	2.40	1690.0	Cyanide Compoun ==> Diesel Pm & 1,3-Butadiene
38.67	88.01	22.00	2.28	1799.0	Cyanide Compoun ==> Toluene & 1,3-Butadiene
35.51	80.23	20.06	2.26	1640.0	Cyanide Compoun ==> Diesel Pm & Benzene
37.29	82.97	20.74	2.23	1696.0	Cyanide Compoun ==> Toluene & Diesel Pm
45.42	99.27	21.77	2.19	1780.0	Cyanide Compoun & Benzene ==> 1,3-Butadiene
45.42	97.93	22.00	2.16	1799.0	Toluene & Cyanide Compoun ==> 1,3-Butadiene
43.78	93.89	16.90	2.14	1382.0	Cyanide Compoun & Acetaldehyde ==> Diesel Pm
45.42	96.74	20.67	2.13	1690.0	Diesel Pm & Cyanide Compoun ==> 1,3-Butadiene
43.78	92.32	20.74	2.11	1696.0	Toluene & Cyanide Compoun ==> Diesel Pm
46.61	97.66	21.41	2.10	1751.0	Cyanide Compoun & Benzene ==> Toluene
45.42	94.97	17.10	2.09	1398.0	Cyanide Compoun & Acetaldehyde ==> 1,3-Butadiene
43.78	91.47	20.06	2.09	1640.0	Cyanide Compoun & Benzene ==> Diesel Pm
46.61	97.08	20.74	2.08	1696.0	Diesel Pm & Cyanide Compoun ==> Toluene
43.78	90.76	20.67	2.07	1690.0	Cyanide Compoun & 1,3-Butadiene ==> Diesel Pm
46.61	96.62	22.00	2.07	1799.0	Cyanide Compoun & 1,3-Butadiene ==> Toluene
42.01	87.08	21.77	2.07	1780.0	Cyanide Compoun ==> Benzene & 1,3-Butadiene
41.54	85.67	21.41	2.06	1751.0	Cyanide Compoun ==> Toluene & Benzene
35.51	72.95	34.00	2.05	2780.0	Toluene ==> Diesel Pm & Benzene
46.61	95.73	34.00	2.05	2780.0	Diesel Pm & Benzene ==> Toluene
46.61	95.58	17.21	2.05	1407.0	Cyanide Compoun & Acetaldehyde ==> Toluene



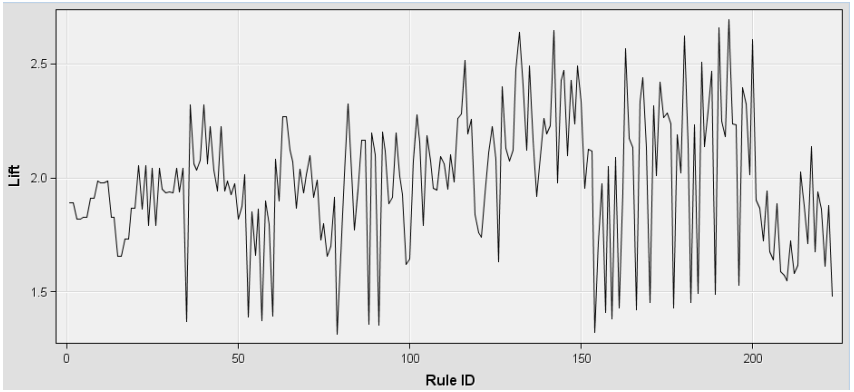
Item Name	Transaction Count	Support(%)
	6819	83.3924
1,3-Butadiene	3714	45.4201
Benzene	4002	48.9422
Diesel Pm	3580	43.7813
Toluene	3811	46.6063
Acetaldehyde	4445	54.3598
Cyanide Compoun	2044	24.9969

3. Level 3 –



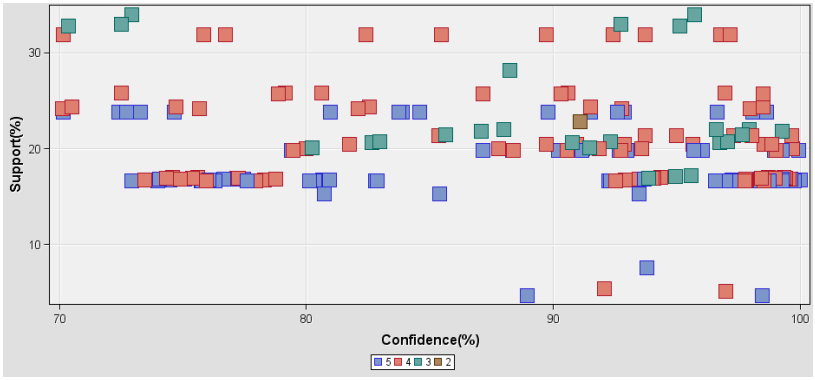


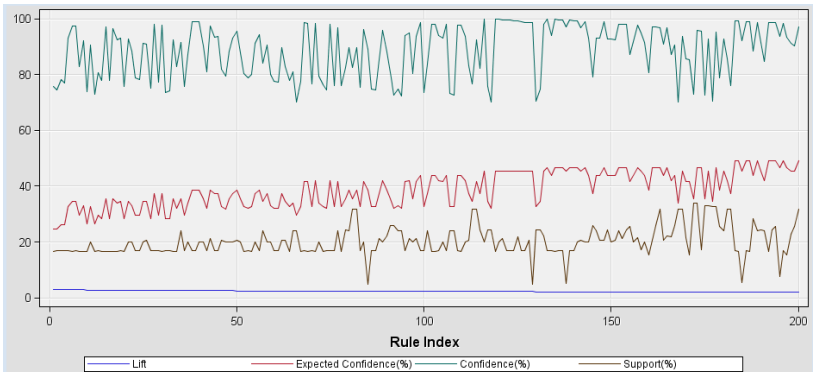
Confidence (%)	Support (%)	Lift	Transaction Count	Rule
92.93	16.73	2.70	1368.0	Cyanide Compoun & Acetaldehyde ==> Diesel Pm & 1,3-Butadiene
78.31	16.73	2.66	1368.0	Diesel Pm & Cyanide Compoun ==> Acetaldehyde & 1,3-Butadiene
91.19	20.00	2.65	1635.0	Cyanide Compoun & Benzene ==> Diesel Pm & 1,3-Butadiene
90.96	20.44	2.64	1671.0	Toluene & Cyanide Compoun ==> Diesel Pm & 1,3-Butadiene
77.19	16.93	2.62	1384.0	Cyanide Compoun & Benzene ==> Acetaldehyde & 1,3-Butadiene
92.53	16.66	2.61	1362.0	Cyanide Compoun & Acetaldehyde ==> Diesel Pm & Benzene
75.61	16.99	2.57	1389.0	Toluene & Cyanide Compoun ==> Acetaldehyde & 1,3-Butadiene
97.32	21.34	2.52	1745.0	Cyanide Compoun & Benzene ==> Toluene & 1,3-Butadiene
93.48	16.83	2.51	1376.0	Cyanide Compoun & Acetaldehyde ==> Toluene & Diesel Pm
81.75	20.44	2.49	1671.0	Cyanide Compoun ==> Toluene & Diesel Pm & 1,3-Butadiene
88.41	19.86	2.49	1624.0	Toluene & Cyanide Compoun ==> Diesel Pm & Benzene
95.65	20.44	2.47	1671.0	Diesel Pm & Cyanide Compoun ==> Toluene & 1,3-Butadiene
87.81	20.00	2.47	1635.0	Cyanide Compoun & 1,3-Butadiene ==> Diesel Pm & Benzene
78.76	16.83	2.47	1376.0	Diesel Pm & Cyanide Compoun ==> Toluene & Acetaldehyde
94.36	16.99	2.44	1389.0	Cyanide Compoun & Acetaldehyde ==> Toluene & 1,3-Butadiene
90.57	19.86	2.43	1624.0	Cyanide Compoun & Benzene ==> Toluene & Diesel Pm
79.99	20.00	2.43	1635.0	Cyanide Compoun ==> Diesel Pm & Benzene & 1,3-Butadiene
77.24	16.94	2.42	1385.0	Cyanide Compoun & Benzene ==> Toluene & Acetaldehyde
89.74	20.44	2.41	1671.0	Cyanide Compoun & 1,3-Butadiene ==> Toluene & Diesel Pm
82.68	20.67	2.40	1690.0	Cyanide Compoun ==> Diesel Pm & 1,3-Butadiene



Item Name	Transaction Count	Support(%)
	6819	83.3924
1,3-Butadiene	3714	45.4201
Benzene	4002	48.9422
Diesel Pm	3580	43.7813
Toluene	3811	46.6063
Acetaldehyde	4445	54.3598
Cyanide Compoun	2044	24.9969

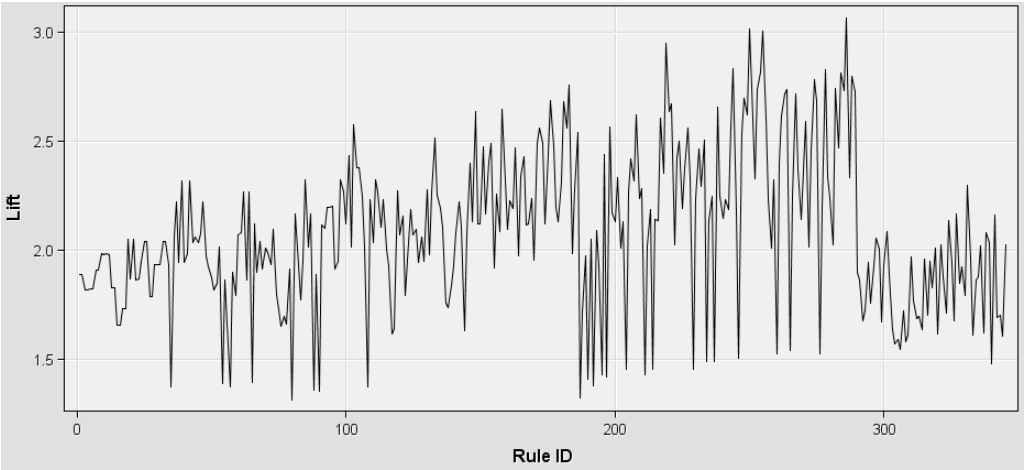
4. Level 4 –





Confidence (%)	Support (%)	Lift	Transaction Count	Rule
75.74	16.61	3.07	1358.0	Cyanide Compoun & Benzene ==> Diesel Pm & Acetaldehyde & 1,3-Butadiene
74.47	16.73	3.02	1368.0	Toluene & Cyanide Compoun ==> Diesel Pm & Acetaldehyde & 1,3-Butadiene
78.31	16.73	3.00	1368.0	Diesel Pm & Cyanide Compoun ==> Toluene & Acetaldehyde & 1,3-Butadiene
76.91	16.86	2.95	1379.0	Cyanide Compoun & Benzene ==> Toluene & Acetaldehyde & 1,3-Butadiene
92.93	16.73	2.83	1368.0	Cyanide Compoun & Acetaldehyde ==> Toluene & Diesel Pm & 1,3-Butadiene
97.49	16.61	2.83	1358.0	Cyanide Compoun & Benzene & Acetaldehyde ==> Diesel Pm & 1,3-Butadiene
97.23	16.73	2.82	1368.0	Toluene & Cyanide Compoun & Acetaldehyde ==> Diesel Pm & 1,3-Butadiene
82.80	16.61	2.81	1358.0	Diesel Pm & Cyanide Compoun & Benzene ==> Acetaldehyde & 1,3-Butadiene
92.26	16.61	2.80	1358.0	Cyanide Compoun & Acetaldehyde ==> Diesel Pm & Benzene & 1,3-Butadiene
73.98	16.62	2.79	1359.0	Toluene & Cyanide Compoun ==> Diesel Pm & Benzene & Acetaldehyde
90.52	19.85	2.76	1623.0	Cyanide Compoun & Benzene ==> Toluene & Diesel Pm & 1,3-Butadiene
72.93	16.61	2.75	1358.0	Cyanide Compoun & 1,3-Butadiene ==> Diesel Pm & Benzene & Acetaldehyde
80.66	16.73	2.74	1368.0	Toluene & Diesel Pm & Cyanide Compoun ==> Acetaldehyde & 1,3-Butadiene
77.79	16.62	2.74	1359.0	Diesel Pm & Cyanide Compoun ==> Toluene & Benzene & Acetaldehyde
97.14	16.61	2.74	1358.0	Cyanide Compoun & Acetaldehyde & 1,3-Butadiene ==> Diesel Pm & Benzene
77.73	16.61	2.73	1358.0	Diesel Pm & Cyanide Compoun ==> Benzene & Acetaldehyde & 1,3-Butadiene
96.59	16.62	2.72	1359.0	Toluene & Cyanide Compoun & Acetaldehyde ==> Diesel Pm & Benzene
92.32	16.62	2.72	1359.0	Cyanide Compoun & Acetaldehyde ==> Toluene & Diesel Pm & Benzene
92.93	16.73	2.70	1368.0	Cyanide Compoun & Acetaldehyde ==> Diesel Pm & 1,3-Butadiene
75.79	16.62	2.69	1359.0	Cyanide Compoun & Benzene ==> Toluene & Diesel Pm & Acetaldehyde

Item Name	Transaction Count	Support(%)
	6819	83.3924
1,3-Butadiene	3714	45.4201
Benzene	4002	48.9422
Diesel Pm	3580	43.7813
Toluene	3811	46.6063
Acetaldehyde	4445	54.3598
Cyanide Compoun	2044	24.9969



2.7 Results analysis and discussion

2.7.1 Result comparison between R and SAS EM

- In R, we observed that Toluene has the highest distribution of average concentration values across the tracts.
- In addition, we see that Toluene and 1,3-Butadiene has strong correlation, along with Benzene and 1,3-Butadiene, Toluene and Diesel PM.
- In simple linear regression between 1,3-Butadiene and Benzene, we see a strong positive line with a very small residual error of 2.3 % across 8175 degrees of freedom (tracts). Also, R-squared value is 71.13% and adjusted R-squared value is 71.12% respectively with a significantly negligible p-value of 2.2e-16 explaining the statistical significance for the regression performed.
- We used the power of Shiny output within the report for rule Explorer. Alternatively, we could use UI and server level components & parameters as well to produce Shiny designs using R.
- Based on the assumption of averages, the high concentration or low concentration levels are prescribed, and both these levels are equivalent and distributed normally.
- In SAS, we experimented with multiple levels of data inputs when the number of combinational rules is less than or equal to 2, 3, 4 etc in multiple instances and confidence as greater than 70 %, and we see equal outputs that we have seen in R.
- So, we can confidently assert that association rules in SAS EM are working as per the design and producing desirable outcomes.

2.7.2 Critical findings

- For high concentrations of 1,3-Butadiene, we observe consequential presence of high concentrations of Benzene as well with 42% support and 92.4% confidence with 1.88 lift.
- For high concentrations of Toluene, we observe consequential presence of high concentrations of Benzene as well with 41.5% support and 89.1% confidence with 1.82 lift.

2.8 Conclusion

In conclusion, high amounts of 1,3-Butadiene and Toluene can independently be associated with high concentrations of Benzene in the atmosphere too as they have strong associations based on the apriori rules created. We proved both R and SAS are working similarly and producing the same outputs with exactly equal values of support , confidence, and lift.

3. Clustering – High risk vs. Low risk of cancer by county in CA

3.1 Abstract

This module of the project aims to support and discover the insights from the research excerpts of National Air Toxicity Assessment (NATA) executed by the US Environmental Protection Agency (EPA) in 2011 and 2014. In this task, we predict the high-risk vs. low-risk zones/counties within the Californian state based on averages of cancer risks per a million people in a county's tract using R and SAS Enterprise Miner tools. To proceed with the Data mining process, we could use any of the Data mining algorithms like SEMMA or CRISP-DM. We used CRISP-DM methodology to perform Data mining for this clustering task. An R markdown file has been used to prepare the k-means cluster model with the slidy presentation rendering an HTML output. The business or operational understanding and data understanding is first performed as part of the requirements gathering. Later we import the dataset and clean it as part of data preparation. To optimize the functionality and reduce the redundancy, we use R functions enabling to break down or decompose a problem into smaller chunks. In addition, the code can be reproducible and reusable, and it was prepared in a systematic, organised, robust and an efficient manner. 'cluster' and 'factoextra' libraries were used to perform this data mining task. We first understand the model and discover the fitting and perform any data wrangling if necessary. To identify the goodness of fit within clustering, we utilise the Silhouette scores or Dunn matrix with the underlying optimum number of clusters for the model. Plots derived from ggplot2 were used wherever necessary to showcase the quality and aesthetics of the graphs. We later utilise SAS Enterprise Miner as a secondary data mining tool where we produce process flow diagrams and parameterize the tasks and compare the results with R.

3.2 Introduction

In 2011 and 2014, the US government agency of the United States Environmental Protection Agency (EPA) assessed the national air toxicity and released a dataset to the public, and this study is titled as the National Air Toxicity Assessment (NATA). EPA developed NATA as a screening tool for state, local and tribal air agencies. NATA's results help these agencies identify which pollutants, emission sources and places they may wish to study further to better understand any possible risks to public health from air toxics. There is now enough evidence that pollutants like acetaldehyde, benzene, cyanide, particulate matter components of diesel engine emissions (namely, diesel PM), toluene, and 1,3-butadiene have been proved to be the root cause for cancer across a wide scale of patients.

Air quality specialists use NATA results to learn which air toxics and emission source types may raise health risks in certain places. They can then study these places in more detail, focusing where the risks to people may be highest. NATA uses a 4-step methodology to develop the assessment:

1. Compile a national emissions inventory of outdoor air toxics sources.
2. Estimate ambient concentrations of air toxics across the United States.
3. Estimate population exposures across the United States.
4. Determine potential public health risks from breathing air toxics.

In this task, we will cluster the counties based on the underlying amount of risk of cancer and create a k-means cluster that could predict the high-risk vs. low-risk zones within the Californian state of USA.

3.2.1 Brief background of the task

Below is the table showing the predictor variables along with their descriptions. Kindly note that all the variables are essentially the average risk of cancer per million due to various causes.

S.No	Variable Name	Type	Brief Description
1	Total_crpm	*Predictor*	Total average cancer risk of a given chemical from all source types
2	railyards_crpm	*Predictor*	Average cancer risk from point sources and railyards

3	airport_crpm	*Predictor*	Average cancer risk from airport sources
4	rwc_crpm	*Predictor*	Average cancer risk from residential wood combustion sources
5	cmv_crpm	*Predictor*	Average cancer risk from commercial marine vessels
6	biogenics_crpm	*Predictor*	Average cancer risk from biogenic sources
7	fires_crpm	*Predictor*	Average cancer risk from fires
8	secondary_crpm	*Predictor*	Average cancer risk due to secondary formation, which is a process by which chemicals are transformed in the air into other chemicals
9	np_10m_releaseheight_crpm	*Predictor*	Average cancer risk from non-point sources with 10m release height
10	np_low_releaseheight_crpm	*Predictor*	Average cancer risk from non-point sources with low release height
11	cmv_loco_crpm	*Predictor*	Average cancer risk from non-road sources (e.g., airplanes, trains, lawn mowers, construction vehicles, farm machinery).
12	lightduty_crpm	*Predictor*	Average cancer risk from on-road light duty mobile sources
13	heavyduty_crpm	*Predictor*	Average cancer risk from on-road heavy duty mobile sources

3.2.2 Formulation of the research question

Using the predictor variables above, can we group the data into multiple clusters, where in we can identify high risk versus low risk within the amount of cancer that is spread across the state of California due to toxic pollutants.

3.2.3 Justification: Why did I choose this topic/dataset?

According to the American Association for Cancer Research, new study suggests that air pollution is also associated with increased risk of mortality for several other types of cancer, including breast, liver, and pancreatic cancer.

Following heart disease, cancer is the second leading cause of death in the United States and around the world. In 2018, an estimated 9.5 million people died of cancer worldwide. That's about 26,000 people each day and 1 out of every 6 deaths. About 600,000 cancer deaths happen in the U.S. each year and about 80,000 in Canada. The rest happen in countries all around the world. About 7 out of every 10 deaths from the disease happen in low- or middle-income countries.

Cancers develop when something goes wrong in the DNA of a cell. Studying the DNA of people who develop cancer, and of those who don't, can be key in identifying people with a particularly high risk. It also helps in the search for new drugs and in choosing the best treatments for patients.

3.3 Aim and Objective of the task

Over the years, there have been various toxic pollutants which have caused Cancer among the people across the counties of California, USA. The objective of this task based on cancer is to be able to cluster the high-risk counties vs. low-risk counties or identify if there could be a third tier (moderate).

3.4 Brief Literature Review

Clustering is an unsupervised machine-learning-based data mining technique used to place data elements into related groups. Clustering data in databases is an important task in real applications of data mining and knowledge discovery. It is the process of partitioning a data set into clusters so that similar objects are put into the same cluster while dissimilar objects are put into different clusters. Farhat Roohi provides a framework for neuro-fuzzy cluster analysis. Neural networks and the fuzzy set theory has emerged as a great breakthrough in the field of clustering, which is a process of grouping data items based on a measure of similarity. Neuro-fuzzy system as it is a self-learning system and generates patterns and rules automatically. Pooja Sikka proposed SVM technique. The new technique is named as K-Means Clustering Based SVM (KMCB-SVM).

K-means clustering is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.

The complexity of SVM depends on no. of input variables and support vectors. The proposed model will apply a clustering algorithm that scans entire data set only to provide the high-quality samples that will carry statistical data. This will provide finer description closer to boundary and farther to boundary. KMCB-SVM would be used for classifying large data sets of relatively low dimensions in large warehouses.

3.5 Explanation and preparation of datasets

3.5.1 Description of the dataset

- **Data Source** - Link
- **Data File (in .xlsx format)** - Link (Size – 193 MB)

- The dataset contains various exposure concentrations, hazard indices, average cancer risk per million, and the population - all grouped by state, EPA region, county, tract, FIPS, and the pollutant. To deal with performance issues with R, we will take only the state of California.

- **Categorical Variables** -
  - State: State in the United States
  - EPA Region: EPA has ten regional offices, each of which is responsible for the execution of its programs within several states and territories.
  - County: A county is a political and administrative division of a state, providing certain local governmental services.
  - FIPS: FIPS (Federal Information Processing Standards) are a set of standards that describe document processing, encryption algorithms and other information technology standards for use within non-military government agencies and by government contractors and vendors who work with the agencies
  - Tract - Numeric code designating census tract from U.S. Census Bureau. Census tracts are Land areas defined by the U.S. Census Bureau. Tracts can vary in size but each typically contains about 4,000 residents. Census tracts are usually smaller than 2 square miles in cities but are much larger in rural areas.
  - Pollutant Name: Name of chemical

- **Continuous Variables** –
  - Population - Number of people in given census tract
  - Total_crpm - Total average cancer risk of a given chemical from all source types
  - railyards_crpm - Average cancer risk from point sources and railyards
  - airport_crpm - Average cancer risk from airport sources
  - rwc_crpm - Average cancer risk from residential wood combustion sources
  - cmv_crpm - Average cancer risk from commercial marine vessels
  - biogenics_crpm - Average cancer risk from biogenic sources
  - fires_crpm - Average cancer risk from fires
  - secondary_crpm - Average cancer risk due to secondary formation, which is a process by which chemicals are transformed in the air into other chemicals
  - np_10m_releaseheight_crpm - Average cancer risk from non-point sources with 10m release height
  - np_low_releaseheight_crpm - Average cancer risk from non-point sources with low release height
  - cmv_loco_crpm - Average cancer risk from non-road sources (e.g., airplanes, trains, lawn mowers, construction vehicles, farm machinery).
  - lightduty_crpm - Average cancer risk from on-road light duty mobile sources
  - heavyduty_crpm - Average cancer risk from on-road heavy duty mobile sources

- **Environment setup :– The following libraries Installed and activated**
  - dplyr : Data manipulations
  - tidyverse : Data science tasks
  - readxl : to Import the .xlsx file
  - skimr : Statistical summary
  - corrplot : Correlation matrix
  - cluster : Clustering
  - factoextra : Clustering & Evaluation
  - ggplot2 : Plotting graphs
  - RColorBrewer : Colour palette

Steps performed in R:

1. Setup the working directory using setwd(<filepath>).
2. Install the readxl package to import the dataset into R using read_excel() function and view the top 6 rows of the dataset.

```
## Importing / Reading the data into "df_cea_raw" data frame
df_cea_raw <- read_excel("ARM Dataset.xlsx", sheet = 1)


# Inspect the raw data
head(df_cea_raw)
```

3. Looking at the top 6 rows

```
## # A tibble: 6 x 17
##   State County Pollutant.Name Point..includes~ Airport.Cancer.~ OR.Lightduty..i~
##   <chr> <chr>  <chr>                    <dbl>            <dbl>            <dbl>
## 1 AK    Aleut~ ACETALDEHYDE         0                0.000995        0.0000719
## 2 AK    Aleut~ ACETALDEHYDE         0.00000248       0.000119        0.0000206
## 3 AK    Aleut~ ACETALDEHYDE         0                0.00561         0.000159
## 4 AK    Ancho~ ACETALDEHYDE         0.0000288        0.00465         0.0429
## 5 AK    Ancho~ ACETALDEHYDE         0.0000607        0.00267         0.0850
## 6 AK    Ancho~ ACETALDEHYDE         0.000135         0.00270         0.386
## # ... with 11 more variables: OR.Heavyduty.Cancer.Risk..per.million. <dbl>,
## #   NR..no.airports..CMV..locomotives..Cancer.Risk..per.million. <dbl>,
## #   NP.10m.ReleaseHeight.Cancer.Risk..per.million. <dbl>,
## #   NP.Low.ReleaseHeight.Cancer.Risk..per.million. <dbl>,
## #   ResidentialWoodCombustion..RWC..Cancer.Risk..per.million. <dbl>,
## #   NR.CommercialMarineVessel..CMV..Cancer.Risk..per.million. <dbl>,
## #   Biogenics.Cancer.Risk..per.million. <dbl>, ...
```

4. Looking at the bottom 6 rows

```
tail(df_cea_raw)
```

```
## # A tibble: 6 x 17
##   State County Pollutant.Name Point..includes~ Airport.Cancer.~ OR.Lightduty..i~
##   <chr> <chr>  <chr>                    <dbl>            <dbl>            <dbl>
## 1 WY    Sweet~ TOLUENE                      0                0                0
## 2 WY    Teton  TOLUENE                      0                0                0
## 3 WY    Uinta  TOLUENE                      0                0                0
## 4 WY    Washa~ TOLUENE                      0                0                0
## 5 WY    Weston TOLUENE                      0                0                0
## 6 WY    Entir~ TOLUENE                      0                0                0
## # ... with 11 more variables: OR.Heavyduty.Cancer.Risk..per.million. <dbl>,
## #   NR..no.airports..CMV..locomotives..Cancer.Risk..per.million. <dbl>,
## #   NP.10m.ReleaseHeight.Cancer.Risk..per.million. <dbl>,
## #   NP.Low.ReleaseHeight.Cancer.Risk..per.million. <dbl>,
## #   ResidentialWoodCombustion..RWC..Cancer.Risk..per.million. <dbl>,
## #   NR.CommercialMarineVessel..CMV..Cancer.Risk..per.million. <dbl>,
## #   Biogenics.Cancer.Risk..per.million. <dbl>, ...
```

5. Checking the variable names

```
names(df_cea_raw)
```

```
##  [1] "State"
##  [2] "County"
##  [3] "Pollutant.Name"
##  [4] "Point..includes.railyards..Cancer.Risk..per.million."
##  [5] "Airport.Cancer.Risk..per.million."
##  [6] "OR.Lightduty..includes.refueling..Cancer.Risk..per.million."
##  [7] "OR.Heavyduty.Cancer.Risk..per.million."
##  [8] "NR..no.airports..CMV..locomotives..Cancer.Risk..per.million."
##  [9] "NP.10m.ReleaseHeight.Cancer.Risk..per.million."
## [10] "NP.Low.ReleaseHeight.Cancer.Risk..per.million."
## [11] "ResidentialWoodCombustion..RWC..Cancer.Risk..per.million."
## [12] "NR.CommercialMarineVessel..CMV..Cancer.Risk..per.million."
## [13] "Biogenics.Cancer.Risk..per.million."
## [14] "Fires..ag..prescribed..and.wild..Cancer.Risk..per.million."
## [15] "Secondary.Cancer.Risk..per.million."
## [16] "Background.Cancer.Risk..per.million."
## [17] "Total.Cancer.Risk..per.million."
```

6. Checking the structure –

```
str(df_cea_raw)
```

```
## tibble [464,075 x 17] (S3: tbl_df/tbl/data.frame)
##  $ State                                                : chr [1:464075] "AK" "AK" "AK" "AK" ...
##  $ County                                               : chr [1:464075] "Aleutians East Borough" "Aleutians West Census Area" "Aleutians
##  $ Pollutant.Name                                       : chr [1:464075] "ACETALDEHYDE" "ACETALDEHYDE" "ACETALDEHYDE" "ACETALDEHYDE" ...
##  $ Point..includes.railyards..Cancer.Risk..per.million. : num [1:464075] 0.00 2.48e-06 0.00 2.88e-05 6.07e-05 ...
##  $ Airport.Cancer.Risk..per.million.                    : num [1:464075] 0.000995 0.000119 0.005605 0.00465 0.002675 ...
##  $ OR.Lightduty..includes.refueling..Cancer.Risk..per.million. : num [1:464075] 7.19e-05 2.06e-05 1.59e-04 4.29e-02 8.50e-02 ...
##  $ OR.Heavyduty.Cancer.Risk..per.million.               : num [1:464075] 6.81e-06 1.05e-05 1.43e-04 1.46e-02 3.79e-02 ...
##  $ NR..no.airports..CMV..locomotives..Cancer.Risk..per.million.: num [1:464075] 7.86e-05 1.41e-05 8.50e-05 4.14e-02 5.09e-02 ...
##  $ NP.10m.ReleaseHeight.Cancer.Risk..per.million.       : num [1:464075] 6.81e-05 1.60e-05 1.39e-04 2.17e-03 1.84e-03 ...
##  $ NP.Low.ReleaseHeight.Cancer.Risk..per.million.       : num [1:464075] 9.97e-07 1.29e-07 1.42e-06 6.16e-04 9.77e-04 ...
##  $ ResidentialWoodCombustion..RWC..Cancer.Risk..per.million. : num [1:464075] 1.92e-05 1.05e-05 2.77e-07 7.50e-03 2.19e-02 ...
##  $ NR.CommercialMarineVessel..CMV..Cancer.Risk..per.million. : num [1:464075] 7.75e-07 4.71e-06 5.11e-03 1.01e-04 1.00e-04 ...
##  $ Biogenics.Cancer.Risk..per.million.                  : num [1:464075] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Fires..ag..prescribed..and.wild..Cancer.Risk..per.million. : num [1:464075] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Secondary.Cancer.Risk..per.million.                  : num [1:464075] 3.51 3.51 3.51 3.51 3.51 ...
##  $ Background.Cancer.Risk..per.million.                 : num [1:464075] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Total.Cancer.Risk..per.million.                      : num [1:464075] 3.51 3.51 3.52 3.63 3.71 ...
```

7. Checking the summary –

```
summary(df_cea_raw)
```

```
##     State              County          Pollutant.Name
##  Length:464075      Length:464075      Length:464075
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##  Point..includes.railyards..Cancer.Risk..per.million.
##  Min.    : 0.000000
##  1st Qu.: 0.000000
##  Median : 0.000000
##  Mean    : 0.012345
##  3rd Qu.: 0.002291
##  Max.    :14.437665
##  Airport.Cancer.Risk..per.million.
##  Min.    :0.000000
##  1st Qu.:0.000000
##  Median :0.000000
##  Mean    :0.007321
##  3rd Qu.:0.003851
##  Max.    :3.187052
##  OR.Lightduty..includes.refueling..Cancer.Risk..per.million.
##  Min.    : 0.0000
##  1st Qu.: 0.0000
##  Median : 0.0000
##  Mean    : 0.6690
##  3rd Qu.: 0.6826
##  Max.    :39.1931
```

8. Check for the dimensionality

```
dim(df_cea_raw)
```

```
## [1] 464075     17
```

9. Renaming the columns as part of simplification

```
# Renaming the columns as part of simplification
names(df_cea_raw)[names(df_cea_raw) == 'State'] <-  'state'
names(df_cea_raw)[names(df_cea_raw) == 'County'] <-  'county'
names(df_cea_raw)[names(df_cea_raw) == 'Pollutant.Name'] <-  'pollutant'
names(df_cea_raw)[names(df_cea_raw) == 'Point..includes.railyards..Cancer.Risk..per.million.'] <-  'railyards_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Airport.Cancer.Risk..per.million.'] <-  'airport_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'OR.Lightduty..includes.refueling..Cancer.Risk..per.million.'] <-  'lightduty_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'OR.Heavyduty.Cancer.Risk..per.million.'] <-  'heavyduty_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'NR..no.airports..CMV..locomotives..Cancer.Risk..per.million.'] <-  'cmv_loco_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'NP.10m.ReleaseHeight.Cancer.Risk..per.million.'] <-  'np_10m_releaseheight_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'NP.Low.ReleaseHeight.Cancer.Risk..per.million.'] <-  'np_low_releaseheight_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'ResidentialWoodCombustion..RWC..Cancer.Risk..per.million.'] <-  'rwc_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'NR.CommercialMarineVessel..CMV..Cancer.Risk..per.million.'] <-  'cmv_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Biogenics.Cancer.Risk..per.million.'] <-  'biogenics_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Fires..ag..prescribed..and.wild..Cancer.Risk..per.million.'] <-  'fires_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Secondary.Cancer.Risk..per.million.'] <-  'secondary_crpm'
names(df_cea_raw)[names(df_cea_raw) == 'Total.Cancer.Risk..per.million.'] <-  'total_crpm'
```

10. Subset the table to a single state – California and aggregate the cancer risk data per category using averages per county. Later, remove the raw dataframe to eliminate redundancy.

```
# 4.1 Subset to state CA
df_cea <- select(filter(df_cea_raw, state == "CA")
                 ,c(state, county, #pollutant,
                    railyards_crpm,
                    airport_crpm,
                    lightduty_crpm,
                    heavyduty_crpm,
                    cmv_loco_crpm,
                    np_10m_releaseheight_crpm,
                    np_low_releaseheight_crpm,
                    rwc_crpm,
                    cmv_crpm,
                    biogenics_crpm,
                    fires_crpm,
                    secondary_crpm,
                    total_crpm  ))

# 4.2 Aggregate data using averages
df_cea_agg <- df_cea %>%
              group_by(state, county) %>%
              summarise_at(vars(c(1:13)), list(avg = mean))

# 4.3 Drop the raw dataframe
remove(df_cea_raw)
```

11. Inspecting the subset – top 6 rows

```
# Describing the data subset
head(df_cea_agg)
```

```
## # A tibble: 6 x 15
## # Groups:   state [1]
##   state county    railyards_crpm_avg airport_crpm_avg lightduty_crpm_avg
##   <chr> <chr>                  <dbl>            <dbl>              <dbl>
## 1 CA    Alameda          0.0181             0.0114               0.760
## 2 CA    Alpine           0.000000926        0.0000301            0.0280
## 3 CA    Amador           0.000152           0.00161              0.124
## 4 CA    Butte            0.00299            0.00270              0.375
## 5 CA    Calaveras        0.000352           0.000432             0.0767
## 6 CA    Colusa           0.00536            0.00226              0.0528
## # ... with 10 more variables: heavyduty_crpm_avg <dbl>,
## #   cmv_loco_crpm_avg <dbl>, np_10m_releaseheight_crpm_avg <dbl>,
## #   np_low_releaseheight_crpm_avg <dbl>, rwc_crpm_avg <dbl>,
## #   cmv_crpm_avg <dbl>, biogenics_crpm_avg <dbl>, fires_crpm_avg <dbl>,
## #   secondary_crpm_avg <dbl>, total_crpm_avg <dbl>
```

12. Bottom 6 rows –

```
tail(df_cea_agg)
```

```
## # A tibble: 6 x 15
## # Groups:   state [1]
##   state county   railyards_crpm_avg airport_crpm_avg lightduty_crpm_avg
##   <chr> <chr>                 <dbl>            <dbl>              <dbl>
## 1 CA    Trinity           0.000483          0.00198             0.0273
## 2 CA    Tulare            0.0106            0.00267             0.466
## 3 CA    Tuolumne          0.000716          0.00256             0.133
## 4 CA    Ventura           0.00402           0.00302             0.398
## 5 CA    Yolo              0.00360           0.00520             0.455
## 6 CA    Yuba              0.00638           0.00374             0.290
## # ... with 10 more variables: heavyduty_crpm_avg <dbl>,
## #   cmv_loco_crpm_avg <dbl>, np_10m_releaseheight_crpm_avg <dbl>,
## #   np_low_releaseheight_crpm_avg <dbl>, rwc_crpm_avg <dbl>,
## #   cmv_crpm_avg <dbl>, biogenics_crpm_avg <dbl>, fires_crpm_avg <dbl>,
## #   secondary_crpm_avg <dbl>, total_crpm_avg <dbl>
```

13. Variable names –

```
names(df_cea_agg)
```

```
##  [1] "state"                          "county"
##  [3] "railyards_crpm_avg"             "airport_crpm_avg"
##  [5] "lightduty_crpm_avg"             "heavyduty_crpm_avg"
##  [7] "cmv_loco_crpm_avg"              "np_10m_releaseheight_crpm_avg"
##  [9] "np_low_releaseheight_crpm_avg"  "rwc_crpm_avg"
## [11] "cmv_crpm_avg"                   "biogenics_crpm_avg"
## [13] "fires_crpm_avg"                 "secondary_crpm_avg"
## [15] "total_crpm_avg"
```

14. Structure –

```
str(df_cea_agg)
```

```
## grouped_df [59 x 15] (S3: grouped_df/tbl_df/tbl/data.frame)
##  $ state                       : chr [1:59] "CA" "CA" "CA" "CA" ...
##  $ county                      : chr [1:59] "Alameda" "Alpine" "Amador" "Butte" ...
##  $ railyards_crpm_avg          : num [1:59] 1.81e-02 9.26e-07 1.52e-04 2.99e-03 3.52e-04 ...
##  $ airport_crpm_avg            : num [1:59] 1.14e-02 3.01e-05 1.61e-03 2.70e-03 4.32e-04 ...
##  $ lightduty_crpm_avg          : num [1:59] 0.76 0.028 0.1243 0.3752 0.0767 ...
##  $ heavyduty_crpm_avg          : num [1:59] 0.05733 0.00088 0.00507 0.01827 0.00334 ...
##  $ cmv_loco_crpm_avg           : num [1:59] 0.2796 0.0173 0.0868 0.1312 0.1138 ...
##  $ np_10m_releaseheight_crpm_avg: num [1:59] 0.04425 0.00255 0.01733 0.01758 0.01431 ...
##  $ np_low_releaseheight_crpm_avg: num [1:59] 0.039641 0.000517 0.003976 0.021578 0.001931 ...
##  $ rwc_crpm_avg                : num [1:59] 0.10183 0.00783 0.06 0.25419 0.04439 ...
##  $ cmv_crpm_avg                : num [1:59] 1.97e-02 3.89e-05 4.22e-06 6.73e-12 9.29e-06 ...
##  $ biogenics_crpm_avg          : num [1:59] 0.0339 0.0406 0.1143 0.1226 0.1032 ...
##  $ fires_crpm_avg              : num [1:59] 0.0246 0.058 0.0751 0.0807 0.0776 ...
##  $ secondary_crpm_avg          : num [1:59] 0.304 0.319 0.55 0.664 0.53 ...
##  $ total_crpm_avg              : num [1:59] 1.694 0.475 1.038 1.691 0.966 ...
##  - attr(*, "groups")= tibble [1 x 2] (S3: tbl_df/tbl/data.frame)
##   ..$ state: chr "CA"
##   ..$ .rows: list<int> [1:1]
##   .. ..$ : int [1:59] 1 2 3 4 5 6 7 8 9 10 ...
##   .. ..@ ptype: int(0)
##   ..- attr(*, ".drop")= logi TRUE
```

15. Summarisation of the new dataset–

```
summary(df_cea_agg)
```

```
##     state              county          railyards_crpm_avg  airport_crpm_avg
##  Length:59          Length:59          Min.   :9.300e-07   Min.   :3.006e-05
##  Class :character   Class :character   1st Qu.:1.694e-03   1st Qu.:1.495e-03
##  Mode  :character   Mode  :character   Median :4.213e-03   Median :2.381e-03
##                                        Mean   :8.070e-03   Mean   :3.829e-03
##                                        3rd Qu.:6.711e-03   3rd Qu.:3.698e-03
##                                        Max.   :5.241e-02   Max.   :3.047e-02
##  lightduty_crpm_avg heavyduty_crpm_avg  cmv_loco_crpm_avg
##  Min.   :0.02727    Min.   :0.0008799   Min.   :0.01731
##  1st Qu.:0.15831    1st Qu.:0.0077556   1st Qu.:0.07572
##  Median :0.37522    Median :0.0190867   Median :0.11629
##  Mean   :0.37114    Mean   :0.0228742   Mean   :0.14111
##  3rd Qu.:0.46576    3rd Qu.:0.0367788   3rd Qu.:0.16001
##  Max.   :1.77834    Max.   :0.0671986   Max.   :0.73445
##  np_10m_releaseheight_crpm_avg np_low_releaseheight_crpm_avg  rwc_crpm_avg
##  Min.   :0.002551              Min.   :0.0005167              Min.   :0.005392
##  1st Qu.:0.008407              1st Qu.:0.0059145              1st Qu.:0.055422
##  Median :0.017581              Median :0.0208691              Median :0.083522
##  Mean   :0.024120              Mean   :0.0250474              Mean   :0.110702
##  3rd Qu.:0.032040              3rd Qu.:0.0353419              3rd Qu.:0.127085
##  Max.   :0.128614              Max.   :0.1458729              Max.   :0.630305
##   cmv_crpm_avg       biogenics_crpm_avg fires_crpm_avg    secondary_crpm_avg
##  Min.   :0.000e+00   Min.   :0.02950    Min.   :0.01081   Min.   :0.2234
##  1st Qu.:0.000e+00   1st Qu.:0.06072    1st Qu.:0.02697   1st Qu.:0.3193
##  Median :2.084e-05   Median :0.08265    Median :0.04486   Median :0.4522
##  Mean   :1.882e-03   Mean   :0.08291    Mean   :0.04895   Mean   :0.4441
##  3rd Qu.:5.624e-04   3rd Qu.:0.10364    3rd Qu.:0.06455   3rd Qu.:0.5459
##  Max.   :4.976e-02   Max.   :0.14315    Max.   :0.12878   Max.   :0.6755
##  total_crpm_avg
##  Min.   :0.3853
##  1st Qu.:0.9822
##  Median :1.2705
##  Mean   :1.2848
##  3rd Qu.:1.6208
##  Max.   :3.2576
```

16. Checking the dimensionality –

```
# Check the number of Rows & Columns in the data subset
nrow(df_cea_agg)
```

```
## [1] 59
```

```
ncol(df_cea_agg)
```

```
## [1] 15
```

```
dim(df_cea_agg)
```

```
## [1] 59 15
```

17. Apply skim() function to take a glance at the overall picture of the data

```
skim(df_cea_agg)
```

Data summary

Name	df_cea_agg
Number of rows	59
Number of columns	15
_____	
Column type frequency:	
character	1
numeric	13
_____	
Group variables	state

Variable type: character

skim_variable	state	n_missing	complete_rate	min	max	empty	n_unique	whitespace
county	CA	0	1	4	15	0	59	0

Variable type: numeric

skim_variable	state	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
railyards_crpm_avg	CA	0	1	0.01	0.01	0.00	0.00	0.00	0.01	0.05	▇▁▁▁▁
airport_crpm_avg	CA	0	1	0.00	0.01	0.00	0.00	0.00	0.00	0.03	▇▁▁▁▁
lightduty_crpm_avg	CA	0	1	0.37	0.29	0.03	0.16	0.38	0.47	1.78	▇▃▁▁▁
heavyduty_crpm_avg	CA	0	1	0.02	0.02	0.00	0.01	0.02	0.04	0.07	▇▅▇▃▂
cmv_loco_crpm_avg	CA	0	1	0.14	0.11	0.02	0.08	0.12	0.16	0.73	▇▂▁▁▁
np_10m_releaseheight_crpm_avg	CA	0	1	0.02	0.02	0.00	0.01	0.02	0.03	0.13	▇▂▁▁▁
np_low_releaseheight_crpm_avg	CA	0	1	0.03	0.02	0.00	0.01	0.02	0.04	0.15	▇▂▁▁▁
rwc_crpm_avg	CA	0	1	0.11	0.10	0.01	0.06	0.08	0.13	0.63	▇▂▁▁▁
cmv_crpm_avg	CA	0	1	0.00	0.01	0.00	0.00	0.00	0.00	0.05	▇▁▁▁▁
biogenics_crpm_avg	CA	0	1	0.08	0.03	0.03	0.06	0.08	0.10	0.14	▇▇▇▇▅
fires_crpm_avg	CA	0	1	0.05	0.03	0.01	0.03	0.04	0.06	0.13	▇▅▆▂▁
secondary_crpm_avg	CA	0	1	0.44	0.13	0.22	0.32	0.45	0.55	0.68	▇▅▂▇▅
total_crpm_avg	CA	0	1	1.28	0.49	0.39	0.98	1.27	1.62	3.26	▅▇▆▂▁

18. A custom function **normalise()** was created to normalize the dataset.

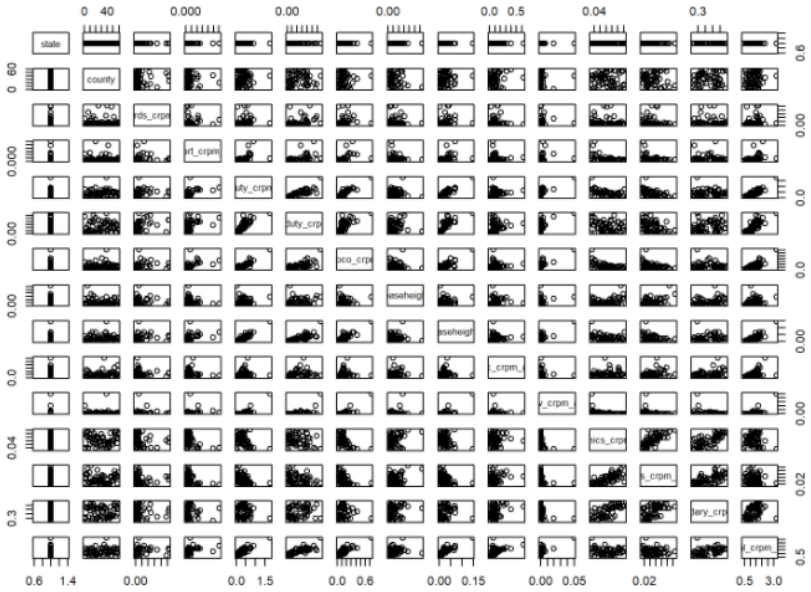
3. Custom function to normalize the data

```
#normalise function
normalise <- function(df)
{
  return(((df- min(df)) /(max(df)-min(df))*(1-0))+0)
}
```

3.5.2 Identify independent dependent variables (if any)

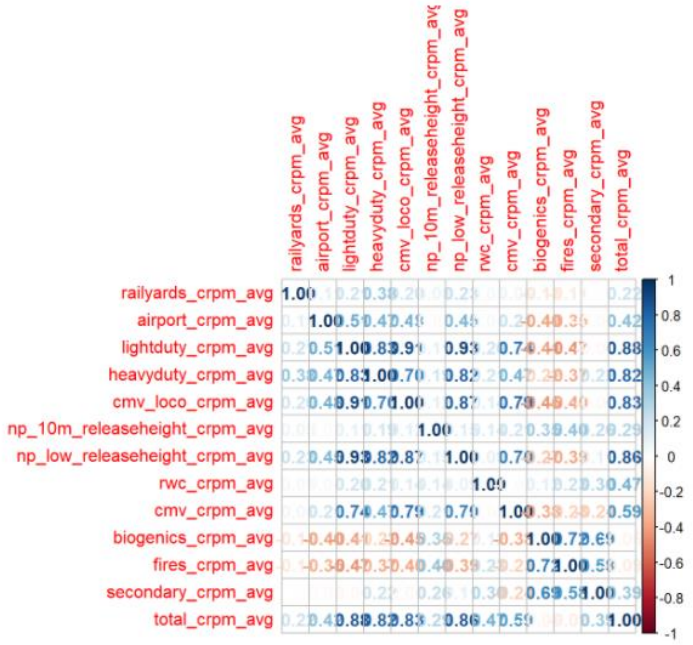
1. During the exploratory data analysis – we convert the state and county as factors and identify the relationships between each variable using pairs() plot.

```
# Pairs plot - Correlation plot
df_cea_agg$state <- as.factor(df_cea_agg$state)
df_cea_agg$county <- as.factor(df_cea_agg$county)
pairs(df_cea_agg)
```



2. Correlation matrix and correlation plot 2 –

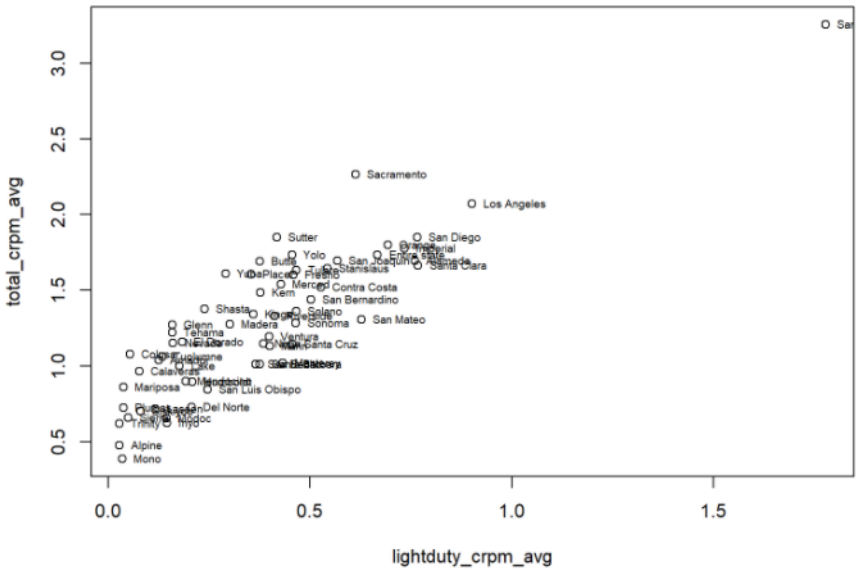
```
rownames(df_cea_agg) <- df_cea_agg$county

corrmatrix <- cor(df_cea_agg[,3:15])
corrplot(corrmatrix, method = 'number')
```



From above outcomes, we can deduce that the average total CRPM has strong correlations with averages of light duty crpm, heavy duty crpm, cmv loco crpm & np_low_release height crpm.

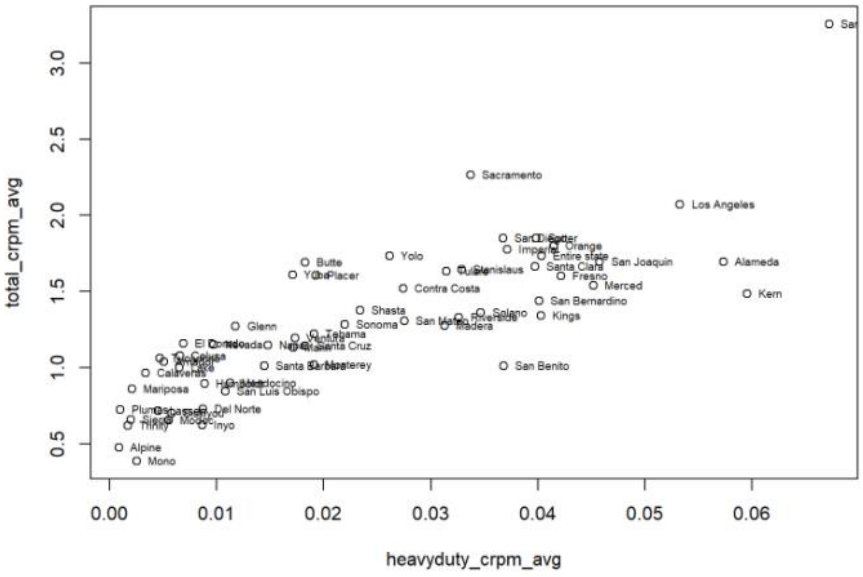
3. Total_crpm_avg vs. lightduty_crpm_avg

```
# Total CRPM avg vs. Light duty CRPM avg
plot(total_crpm_avg ~ lightduty_crpm_avg, data = df_cea_agg)
with(df_cea_agg,text(total_crpm_avg ~ lightduty_crpm_avg, labels= county,pos=4,cex=.6))
```



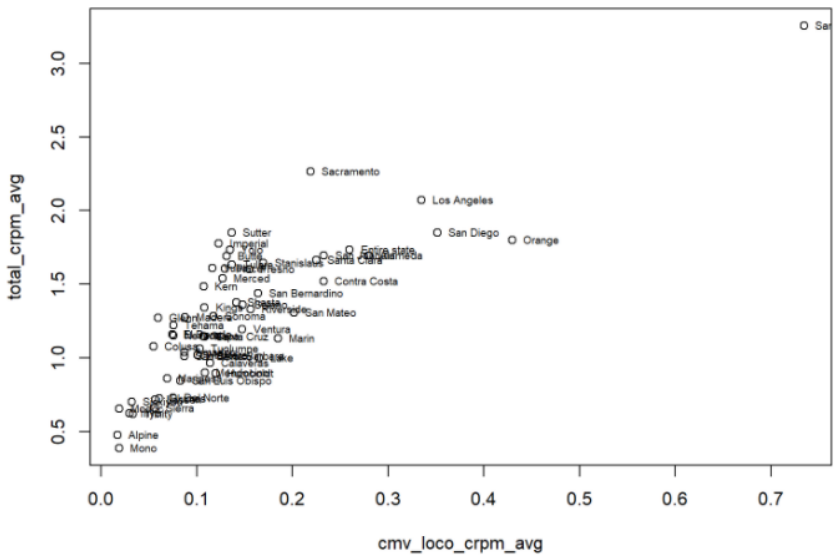
4. Total_crpm_avg vs. heavyduty_crpm_avg

```
# Total CRPM avg vs. Heavy duty CRPM avg
plot(total_crpm_avg ~ heavyduty_crpm_avg, data = df_cea_agg)
with(df_cea_agg,text(total_crpm_avg ~ heavyduty_crpm_avg, labels= county,pos=4,cex=.6))
```



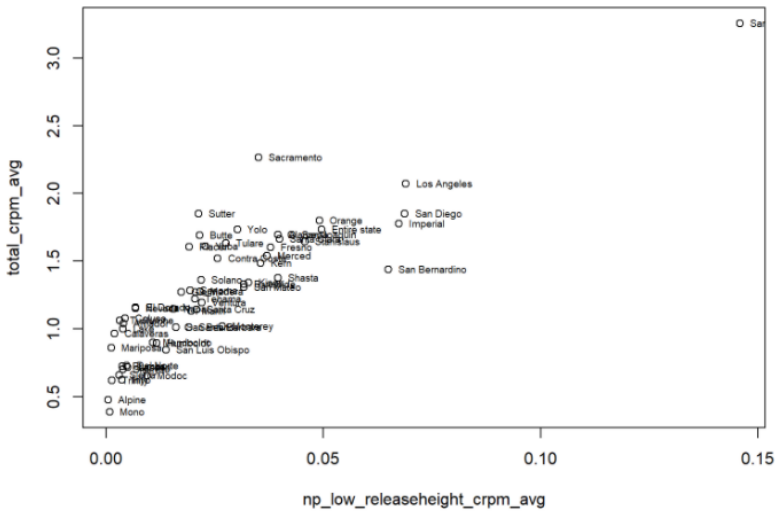
5. Total_crpm_avg vs. cmv_loco_crpm_avg

```
# Total CRPM avg vs. CMV_Loco_CRPM_avg
plot(total_crpm_avg ~ cmv_loco_crpm_avg, data = df_cea_agg)
with(df_cea_agg,text(total_crpm_avg ~ cmv_loco_crpm_avg, labels= county,pos=4,cex=.6))
```



6. Total_crpm_avg vs. np_low_releaseheight_crpm_avg

```
# Total CRPM avg vs. np_low_release_Height_crpm_avg
plot(total_crpm_avg ~ np_low_releaseheight_crpm_avg, data = df_cea_agg)
with(df_cea_agg,text(total_crpm_avg ~ np_low_releaseheight_crpm_avg, labels= county,pos=4,cex=.6))
```



3.5.3 Data Pre-processing steps

During this phase, we normalise the dataset and apply Euclidean distance across the variables for each of the county and create a distance matrix.

1. Separate the country.

```
# 6.1.1 Separate the county
df_cea_agg$county <- as.character(df_cea_agg$county)
county <- df_cea_agg[,2]
county
```

```
## # A tibble: 59 x 1
##    county
##    <chr>
##  1 Alameda
##  2 Alpine
##  3 Amador
##  4 Butte
##  5 Calaveras
##  6 Colusa
##  7 Contra Costa
##  8 Del Norte
##  9 El Dorado
## 10 Entire state
## # ... with 49 more rows
```

2. Normalise the dataset, merge with county and apply rownames and sort by county in alphabetical order, and later inspect the dimensionality.

```
# 6.1.2 Normalise the dataset using the custom created function
df_cea_agg_n <- as.data.frame(lapply(df_cea_agg[,3:15] ,normalise))

# 6.1.3 Merge with County, apply rownames & sort by County
df_cea_agg_n <- cbind(county, df_cea_agg_n)
rownames(df_cea_agg_n)<-df_cea_agg_n$county
df_cea_agg_n <- df_cea_agg_n[order(county),]

# 6.2 Inspect the normalized dataset
dim(df_cea_agg_n)
```

```
## [1] 59 14
```

3. Structure of the dataset

```
str(df_cea_agg_n)
```

```
## 'data.frame':    59 obs. of  14 variables:
##  $ county                      : chr  "Alameda" "Alpine" "Amador" "Butte" ...
##  $ railyards_crpm_avg          : num  0.34452 0 0.00287 0.05702 0.00669 ...
##  $ airport_crpm_avg            : num  0.3723 0 0.0519 0.0876 0.0132 ...
##  $ lightduty_crpm_avg          : num  0.418447 0.000396 0.055416 0.198708 0.028218 ...
##  $ heavyduty_crpm_avg          : num  0.8512 0 0.0631 0.2623 0.0371 ...
##  $ cmv_loco_crpm_avg           : num  0.3657 0 0.0969 0.1588 0.1345 ...
##  $ np_10m_releaseheight_crpm_avg: num  0.3308 0 0.1172 0.1192 0.0933 ...
##  $ np_low_releaseheight_crpm_avg: num  0.26917 0 0.0238 0.1449 0.00973 ...
##  $ rwc_crpm_avg                : num  0.15433 0.00391 0.08739 0.39814 0.0624 ...
##  $ cmv_crpm_avg                : num  3.95e-01 7.83e-04 8.48e-05 1.35e-10 1.87e-04 ...
##  $ biogenics_crpm_avg          : num  0.0384 0.0977 0.7461 0.8191 0.6485 ...
##  $ fires_crpm_avg              : num  0.117 0.4 0.545 0.593 0.567 ...
##  $ secondary_crpm_avg          : num  0.178 0.211 0.722 0.974 0.678 ...
##  $ total_crpm_avg              : num  0.4556 0.0311 0.2274 0.4545 0.2022 ...
```

4. Variables names –

```
names(df_cea_agg_n)
```

```
##  [1] "county"                          "railyards_crpm_avg"
##  [3] "airport_crpm_avg"                "lightduty_crpm_avg"
##  [5] "heavyduty_crpm_avg"              "cmv_loco_crpm_avg"
##  [7] "np_10m_releaseheight_crpm_avg"   "np_low_releaseheight_crpm_avg"
##  [9] "rwc_crpm_avg"                    "cmv_crpm_avg"
## [11] "biogenics_crpm_avg"              "fires_crpm_avg"
## [13] "secondary_crpm_avg"              "total_crpm_avg"
```

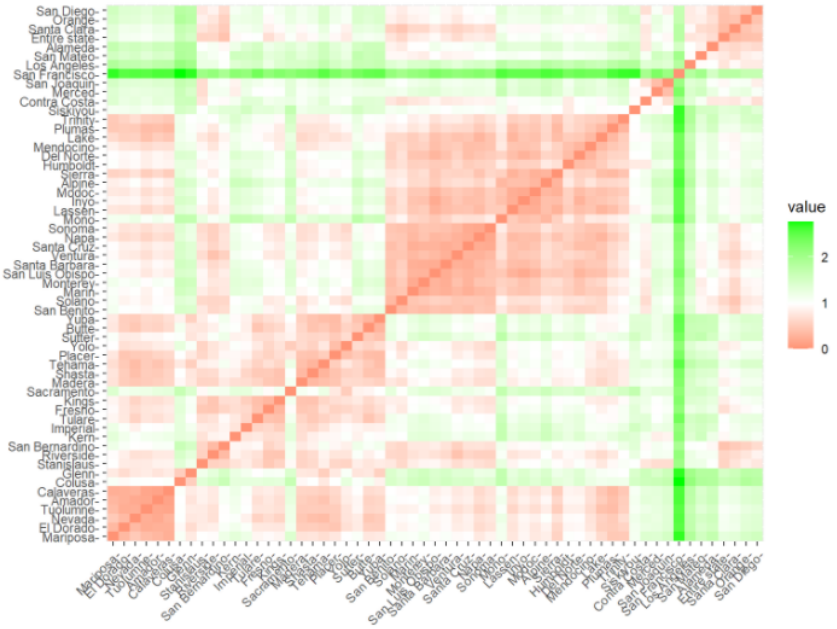
5. Create and analyse the distance between spatial points using the Euclidean method , and create the distance matrix.

```
# 6.3.1 Choosing Euclidean distance method and creating distance matrix
distance <- dist(df_cea_agg_n,method = "euclidean")
print(distance,digits=3)
```

```
##                Alameda Alpine Amador Butte Calaveras Colusa Contra Costa
## Alpine           1.436
## Amador           1.589  0.917
## Butte            1.590  1.334  0.571
## Calaveras        1.570  0.811  0.140 0.657
## Colusa           1.850  1.501  0.982 1.121     1.023
## Contra Costa     0.772  1.155  1.316 1.362     1.280  1.575
## Del Norte        1.274  0.586  0.902 1.269     0.853  1.404        0.994
## El Dorado        1.515  0.880  0.262 0.544     0.260  0.921        1.230
## Entire state     0.669  1.223  1.152 1.096     1.153  1.609        0.792
## Fresno           1.189  1.255  0.770 0.575     0.817  1.332        1.139
## Glenn            1.828  1.431  0.733 0.769     0.795  0.529        1.590
## Humboldt         1.178  0.789  0.907 1.200     0.878  1.260        0.796
## Imperial         1.401  1.543  0.963 0.834     1.050  1.287        1.360
## Inyo             1.295  0.398  0.857 1.246     0.792  1.472        1.046
## Kern             1.197  1.569  1.147 1.009     1.197  1.292        1.120
## Kings            1.280  1.205  0.683 0.639     0.744  1.299        1.194
## Lake             1.273  0.603  0.524 0.887     0.465  1.148        0.947
## Lassen           1.299  0.399  0.707 1.100     0.633  1.339        0.955
## Los Angeles      0.996  1.810  1.742 1.624     1.756  2.119        1.361
## Madera           1.238  1.060  0.540 0.599     0.586  0.919        1.069
## Marin            0.970  0.633  0.979 1.208     0.920  1.540        0.744
## Mariposa         1.680  0.830  0.320 0.761     0.244  1.062        1.402
## Mendocino        1.276  0.734  0.706 1.037     0.690  1.221        1.026
## Merced           1.249  1.554  1.294 1.175     1.312  1.613        0.931
## Modoc            1.296  0.317  0.770 1.173     0.698  1.362        1.048
## Mono             1.393  0.325  1.152 1.543     1.060  1.703        1.143
## Monterey         0.977  0.653  1.077 1.320     1.026  1.596        0.785
## Napa             1.140  0.591  0.681 0.930     0.635  1.344        0.887
## Nevada           1.557  1.010  0.199 0.466     0.298  1.002        1.283
## Orange           0.696  1.298  1.373 1.359     1.346  1.866        0.890
## Placer           1.349  1.141  0.588 0.380     0.630  0.991        1.080
## Plumas           1.516  0.619  0.398 0.896     0.341  1.180        1.214
## Riverside        1.085  0.987  0.784 0.845     0.788  1.462        1.005
## Sacramento       1.385  1.586  1.271 0.892     1.301  1.649        1.241
```

6. As we are not able to visualise everything in the same output previously, we use a correlogram to visualize the effectiveness of the spatial points. We use **fviz_dist()** function for this. Clearly San Francisco stands out with the maximum distance, which means it has the highest air pollution averages.

```
fviz_dist(distance, order = TRUE, gradient = list(low = "red", mid = "white", high = "green"))
```



7. Export the cleaned dataset to a .csv to apply clustering later using SAS.

4. Saving the cleaned dataset as a .csv (for SAS)

```
# 4.1 - Write to a .csv file
write.csv(df_cea_agg_n, "Average_CancerRisk_per_County.csv", row.names=TRUE)
```

3.5.4 Assumptions (if any)

- The average cancer risk per million values for each of the county is considered as a granular value for clustering. Hence, we would have as many rows as many counties in a state.

3.6 Task: Clustering

3.6.1 Data Exploration and Attribute Visualization in R

3.6.1.1 Model Building in R

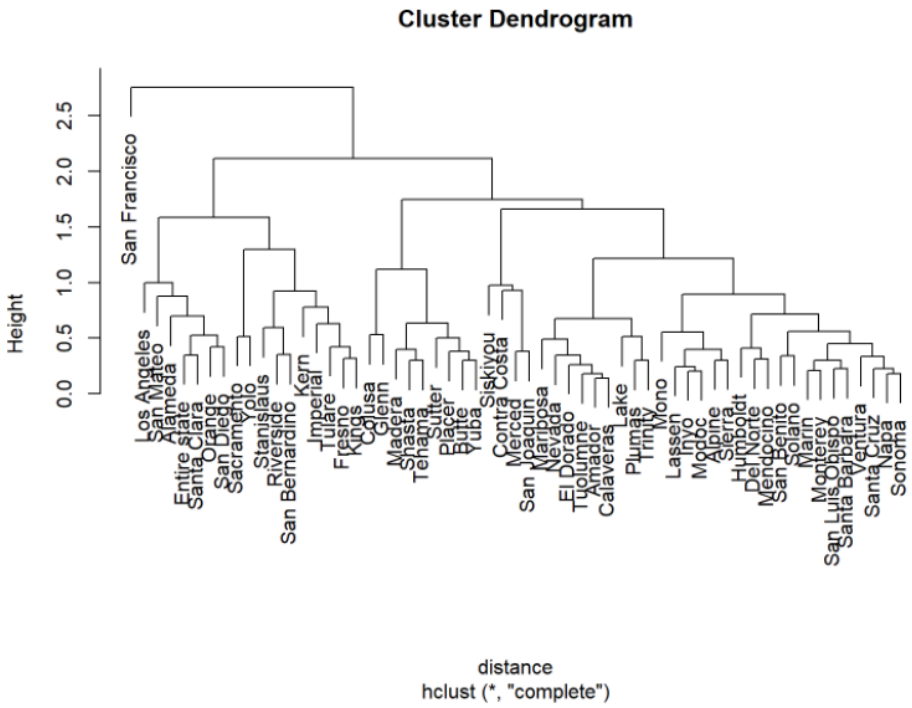
1. Hierarchical clustering using the Euclidean distance is checked first.

```
# 7.1.1 - Cluster Dendrogram
df_cea_agg.hclust <- hclust(distance)
df_cea_agg.hclust
```

```
##
## Call:
## hclust(d = distance)
##
## Cluster method   : complete
## Distance         : euclidean
## Number of objects: 59
```

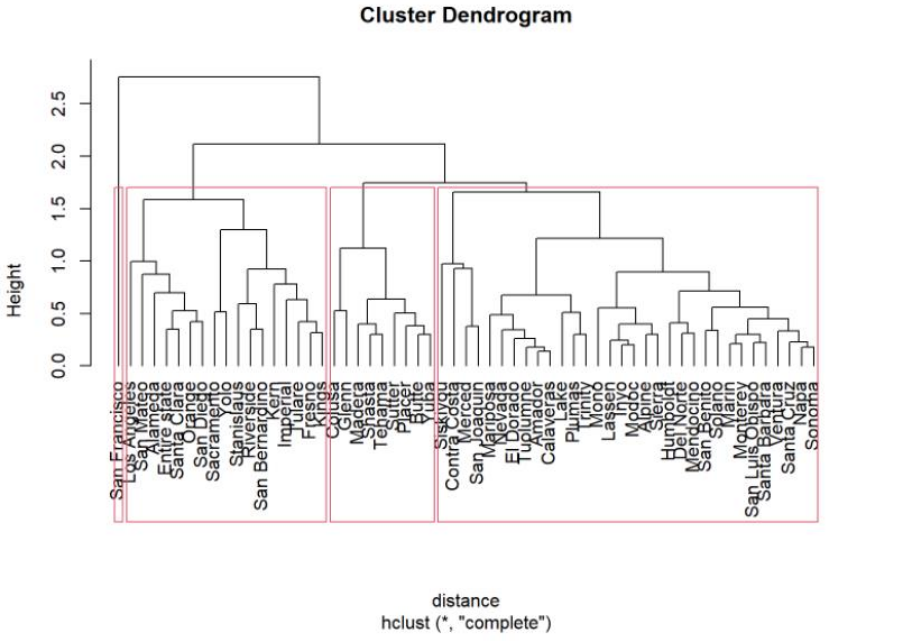
2. Plotting the hierarchical clustering using a dendrogram

```
plot(df_cea_agg.hclust)
```



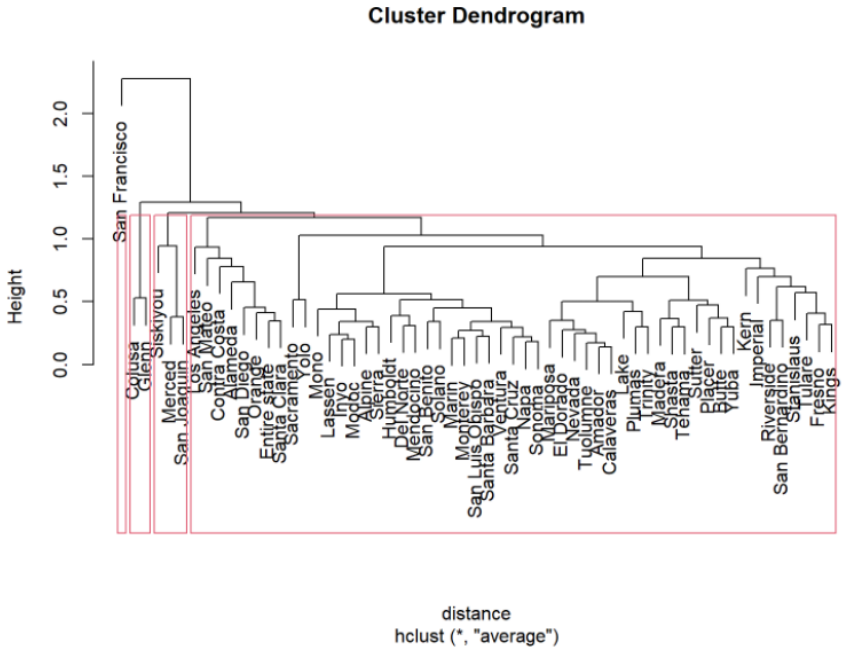
3. Applying the grouping using rectangles
4.

```
plot(df_cea_agg.hclust,hang=-1)

rect.hclust(df_cea_agg.hclust, 4)
```



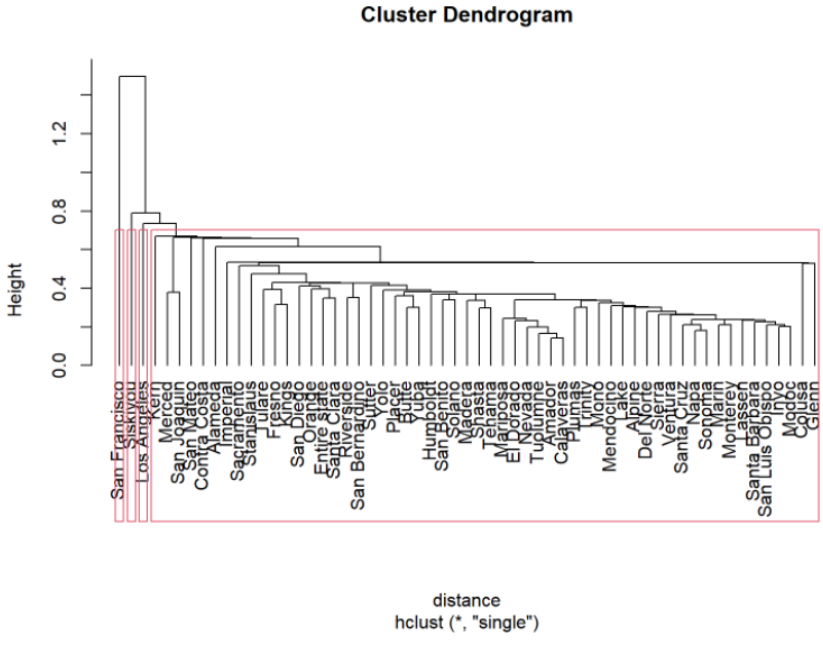
5. Hierarchical clustering using average linkage

```
# 7.1.2 Hierarchical clustering using average linkage
hclust.average <- hclust(distance, method = "average")
plot(hclust.average)
rect.hclust(hclust.average, 4)
```



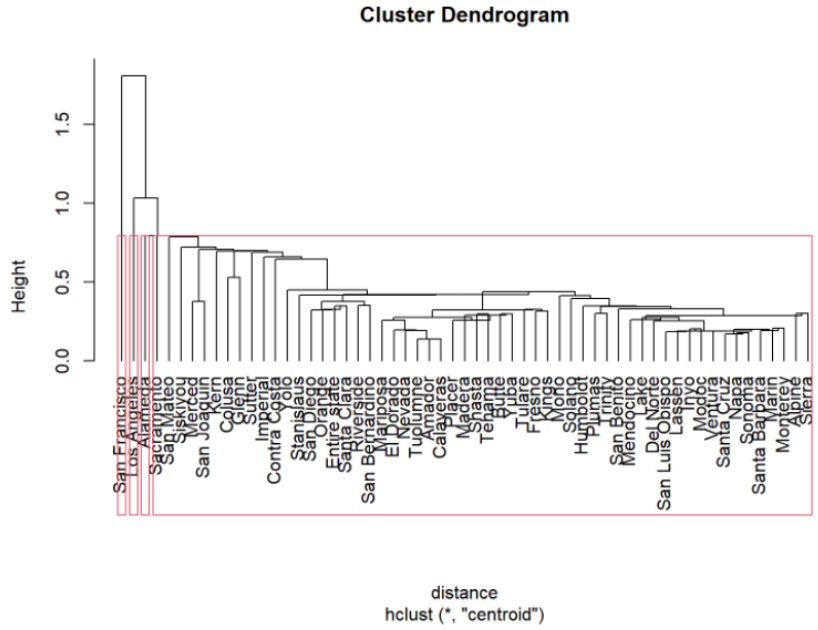
6. Hierarchical clustering using single linkage –

```
# 7.1.3 Hierarchical clustering using single linkage
hclust.single <- hclust(distance, method = "single")
plot(hclust.single,hang = -1)
rect.hclust(hclust.single, 4)
```



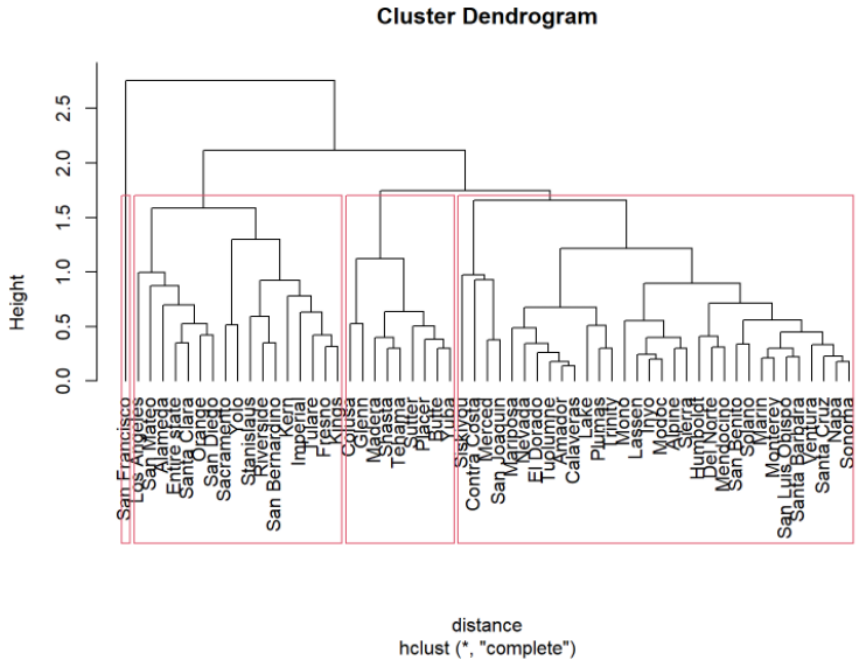
7. Hierarchical clustering using centroid linkage

```
# 7.1.4 Hierarchical clustering using centroid linkage
hclust.centroid <- hclust(distance, method = "centroid")
plot(hclust.centroid,hang = -1)
rect.hclust(hclust.centroid, 4)
```



8. Hierarchical clustering using complete linkage

```
# 7.1.5 Hierarchical clustering using complete linkage
hclust.complete <- hclust(distance, method = "complete")
plot(hclust.complete,hang = -1)
rect.hclust(hclust.complete, 4)
```



9. Identifying the members of the dendrogram & visualise it in a table format

```
# 7.1.6 Members of dendrogram
member.centroid <- cutree(hclust.centroid,4)
member.centroid
```

```
##         Alameda          Alpine          Amador          Butte       Calaveras
##               1               2               2               2               2
##          Colusa    Contra Costa       Del Norte      El Dorado    Entire state
##               2               2               2               2               2
##          Fresno          Glenn       Humboldt        Imperial           Inyo
##               2               2               2               2               2
##            Kern          Kings            Lake          Lassen     Los Angeles
##               2               2               2               2               3
##          Madera          Marin        Mariposa       Mendocino         Merced
##               2               2               2               2               2
##           Modoc           Mono        Monterey            Napa         Nevada
##               2               2               2               2               2
##          Orange         Placer          Plumas       Riverside     Sacramento
##               2               2               2               2               2
##      San Benito  San Bernardino       San Diego   San Francisco     San Joaquin
##               2               2               2               4               2
## San Luis Obispo       San Mateo  Santa Barbara     Santa Clara      Santa Cruz
##               2               2               2               2               2
##          Shasta          Sierra        Siskiyou          Solano         Sonoma
##               2               2               2               2               2
##       Stanislaus          Sutter          Tehama         Trinity         Tulare
##               2               2               2               2               2
##         Tuolumne         Ventura           Yolo            Yuba
##               2               2               2               2
```

```
member.complete <- cutree(hclust.complete,4)
member.complete
```

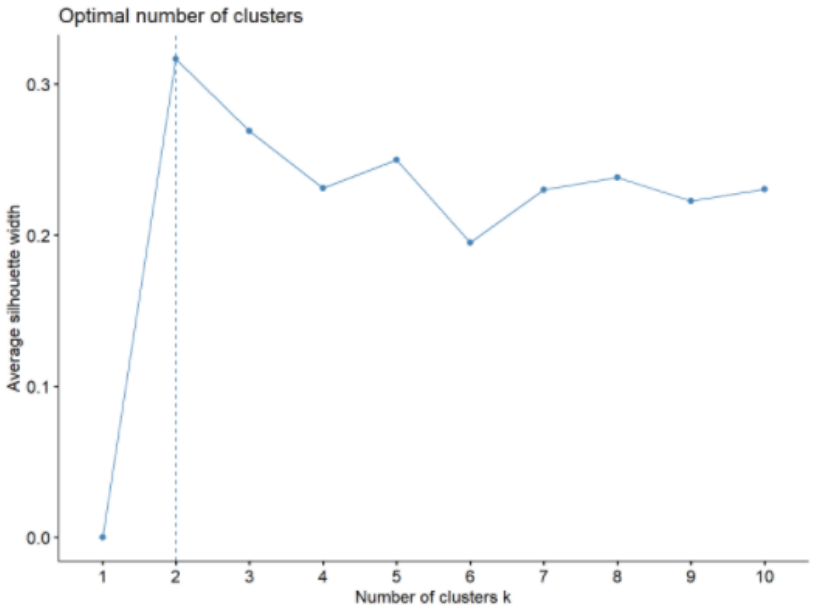
```
##         Alameda          Alpine          Amador          Butte       Calaveras
##               1               2               2               3               2
##          Colusa    Contra Costa       Del Norte      El Dorado    Entire state
##               3               2               2               2               1
##          Fresno          Glenn       Humboldt        Imperial           Inyo
##               1               3               2               1               2
##            Kern          Kings            Lake          Lassen     Los Angeles
##               1               1               2               2               1
##          Madera          Marin        Mariposa       Mendocino         Merced
##               3               2               2               2               2
##           Modoc           Mono        Monterey            Napa         Nevada
##               2               2               2               2               2
##          Orange         Placer          Plumas       Riverside     Sacramento
##               1               3               2               1               1
##      San Benito  San Bernardino       San Diego   San Francisco     San Joaquin
##               2               1               1               4               2
## San Luis Obispo       San Mateo  Santa Barbara     Santa Clara      Santa Cruz
##               2               1               2               1               2
##          Shasta          Sierra        Siskiyou          Solano         Sonoma
##               3               2               2               2               2
##       Stanislaus          Sutter          Tehama         Trinity         Tulare
##               1               3               3               2               1
##         Tuolumne         Ventura           Yolo            Yuba
##               2               2               1               3
```

```
table(member.centroid,member.complete)
```

```
##                member.complete
## member.centroid  1  2  3  4
##               1  1  0  0  0
##               2 15 32  9  0
##               3  1  0  0  0
##               4  0  0  0  1
```

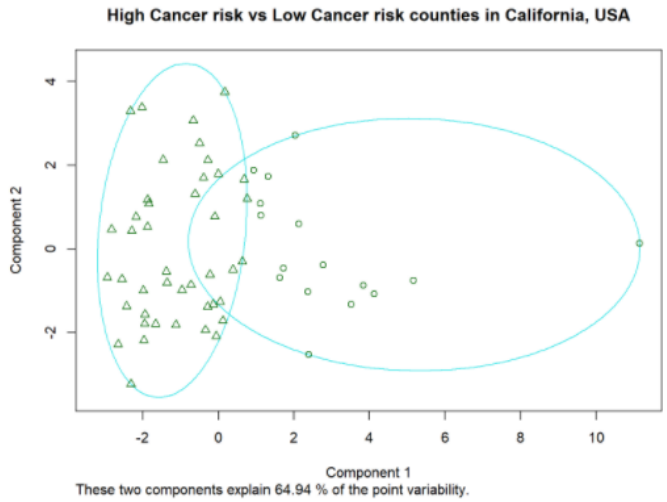
10. For K-means clustering, we'll identify the optimal number of clusters first. Optimal number of clusters = 2

```
# 7.2.1 - Scale the continuous data
scaled_df <- scale(df_cea_agg_n[,2:14])

# 7.2.2 - Calculate the Optimal number of clusters using factoextra package
fviz_nbclust(scaled_df, kmeans, method='silhouette')
```



11. Plotting the graphs to observe the distinction / overlap between the clusters.

```
# 7.2.3 - Apply the kmeans() algorithm based on the optimal no. of clusters
kclus <- kmeans(scaled_df, 2)$cluster

# 7.2.4 - Plot the graphs (without data points)
clusplot(scaled_df, kclus, main = "High Cancer risk vs Low Cancer risk counties in California, USA")
```

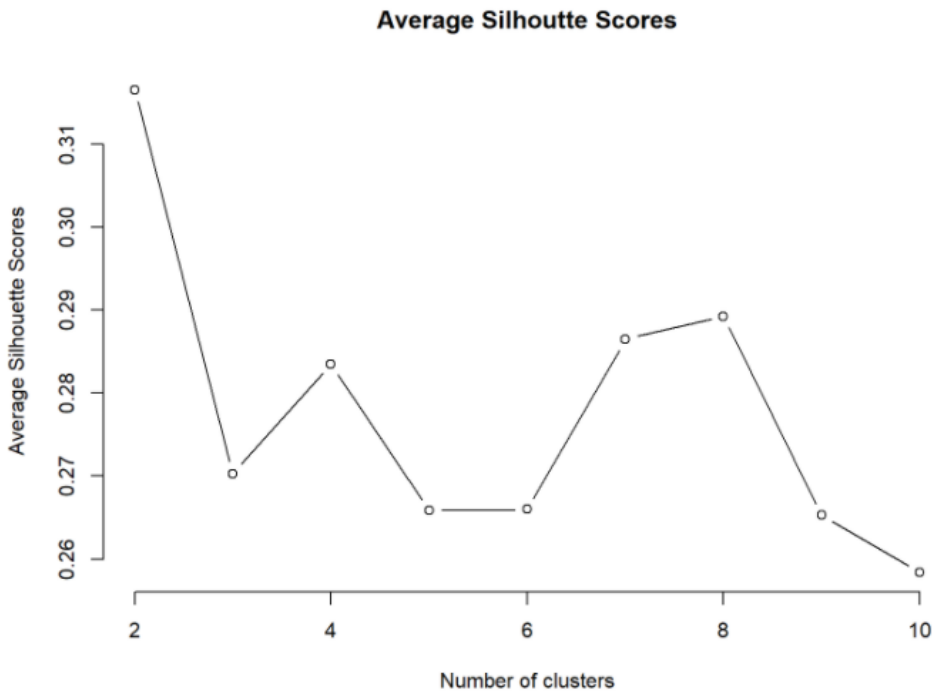



```
# 7.2.5 - Plot the graphs (with data points)
clusplot(scaled_df, kclus, color = TRUE, shade = TRUE, cex=0.6, labels = 2, lines = 0, main = "High Cancer risk vs Low Cancer risk counties in California, USA")
```

3.6.1.2 Model Assessment in R

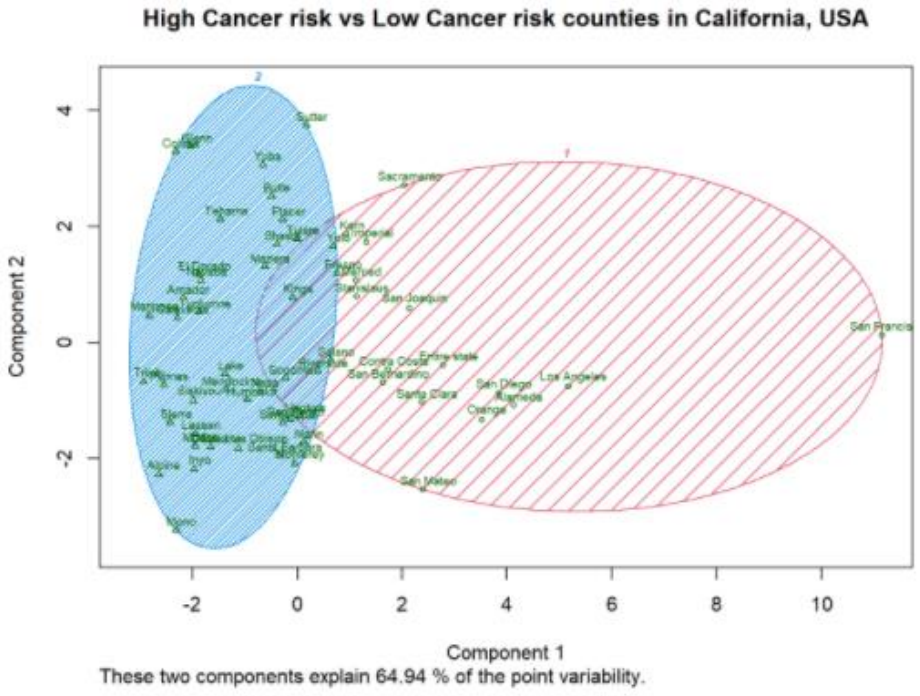
- The created unsupervised model is evaluated using Silhoutte scores

```r
# Silhoutte Score
silhouette_score <- function(k){
  km <- kmeans(scaled_df, centers = k, nstart=25)
  ss <- silhouette(km$cluster, dist(scaled_df))
  mean(ss[, 3])
}
k <- 2:10
avg_sil <- sapply(k, silhouette_score)
plot(k, type='b', avg_sil, main = 'Average Silhoutte Scores', xlab='Number of clusters', ylab='Average Silhouette Scores', frame=FALSE)
```



3.6.1.3 Results visualisation in R

1. Explained with data points and point variability.



3.6.2 Data Exploration and Attribute Visualization in SAS EM

3.6.2.1 Model Building in SAS EM

1. We will import the file into SAS EM using 'File Import' option with the below settings. Score role is Train.

.. Property	Value
General	
Node ID	FIMPORT
Imported Data	...
Exported Data	...
Notes	...
Train	
Variables	...
Import File	D:\University\ASDM\Develo ...
Maximum Rows to Import	1000000
Maximum Columns to Import	10000
Delimiter	,
Name Row	Yes
Number of Rows to Skip	0
Guessing Rows	500
File Location	Local
File Type	csv
Advanced Advisor	No
Rerun	No
Score	
Role	Train
Report	
Summarize	No
Status	
Create Time	30/12/21 18:22
Run ID	acb0ff57-6115-4d86-b366-67f
Last Error	
Last Status	Complete
Last Run Time	30/12/21 18:34
Run Duration	0 Hr. 0 Min. 2.83 Sec.
Grid Host	
User-Added Node	No

2. Adjust the variable types as shown below.

Name	Role	Level	Report
airport_crpm_a	**Input**	**Interval**	**No**
biogenics_crpn	**Input**	**Interval**	**No**
cmv_crpm_avg	**Input**	**Interval**	**No**
cmv_loco_crpn	**Input**	**Interval**	**No**
county	**Label**	**Nominal**	**No**
fires_crpm_av	**Input**	**Interval**	**No**
heavyduty_crp	**Input**	**Interval**	**No**
lightduty_crpm	**Input**	**Interval**	**No**
np_10m_relea	**Input**	**Interval**	**No**
np_low_releas	**Input**	**Interval**	**No**
railyards_crpm	**Input**	**Interval**	**No**
rwc_crpm_avg	**Input**	**Interval**	**No**
secondary_crp	**Input**	**Interval**	**No**
total_crpm_av	**Input**	**Interval**	**No**
VAR1	**ID**	**Nominal**	**No**

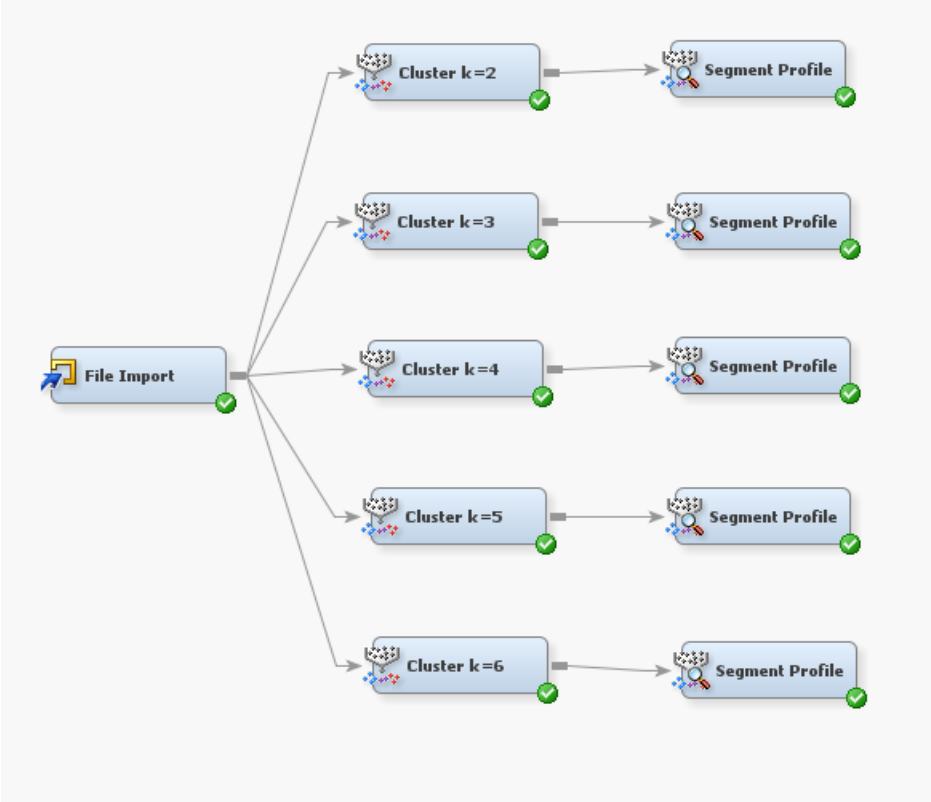
3. Applying cluster tab and use the below settings, we would be using cluster k value with multiple scenarios and check each of them later.

Property	Value
General	
Node ID	Clus2
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Internal Standardization	Standardization
Number of Clusters	
Specification Method	User Specify
Maximum Number of Cluster	2
Selection Criterion	
Clustering Method	Ward
Preliminary Maximum	50
Minimum	2
Final Maximum	20
CCC Cutoff	3
Encoding of Class Variables	
Ordinal Encoding	Rank
Nominal Encoding	GLM
Initial Cluster Seeds	
Seed Initialization Method	Default
Minimum Radius	0.0
Drift During Training	No
Training Options	
Use Defaults	Yes
Settings	
Missing Values	
Interval Variables	Default
Nominal Variables	Default
Ordinal Variables	Default
Scoring Imputation Method	None
Score	
Cluster Variable Role	Segment
Hide Original Variables	Yes
Cluster Label Editor	
Report	
Cluster Graphs	Yes
Tree Profile	Yes
Distance Plot and Table	Yes
Status	

4. Apply segment profile

Property	Value
General	
Node ID	Prof2
Imported Data	
Exported Data	
Notes	
Train	
Variables	
General	
Number of Midpoints	8
Profile All	No
Cutoff Percentage	95
Input Variables	
Number of Inputs	13
Minimum Worth	0.01
Maximum Depth	1
Print Worth Statistics	Yes
Target Variables	
Analysis Role	None
Report Variables	
Use Report Variables	Yes
Number of Report Variables	13
Status	
Create Time	30/12/21 18:40
Run ID	8d258623-8655-471a-8cb8-0f
Last Error	
Last Status	Complete
Last Run Time	30/12/21 18:42
Run Duration	0 Hr. 0 Min. 3.42 Sec.
Grid Host	
User-Added Node	No

5. Final model looks like –

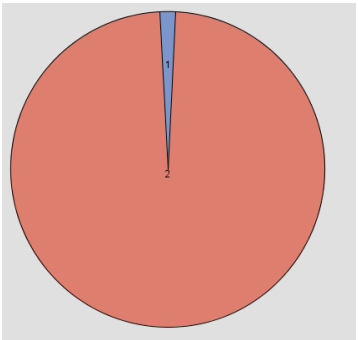


3.6.2.2 Model Assessment in SAS EM

1. The contents procedure is used to view the data from the cleaned dataset.

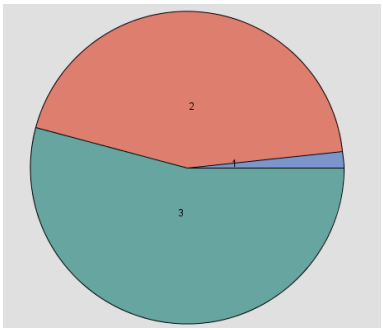
```
25    The CONTENTS Procedure
26
27    Data Set Name        EMWS3.FIMPORT_DATA          Observations          59
28    Member Type          DATA                        Variables             15
29    Engine               V9                          Indexes               0
30    Created              30/12/2021 18:34:31         Observation Length    144
31    Last Modified        30/12/2021 18:34:31         Deleted Observations  0
32    Protection                                       Compressed            NO
33    Data Set Type                                    Sorted                NO
34    Label
35    Data Representation  WINDOWS_32
36    Encoding             wlatinl  Western (Windows)
```

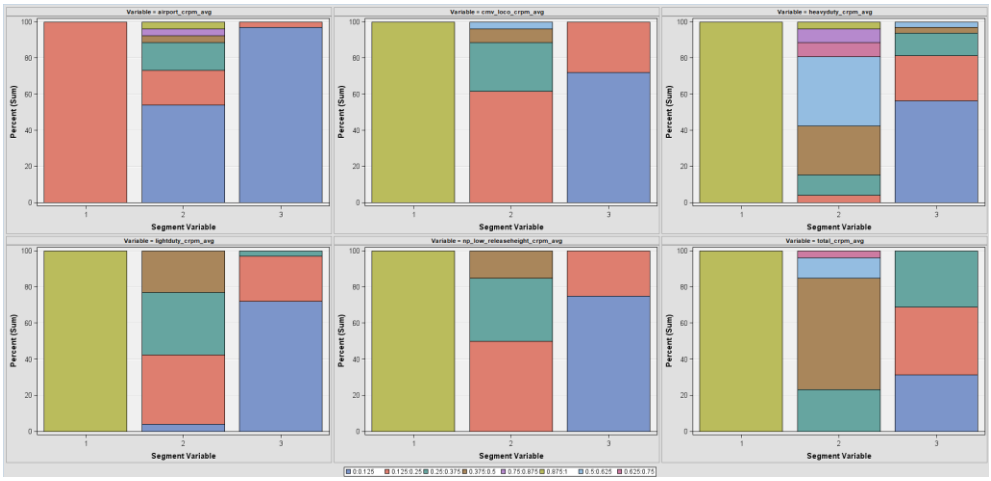
2. Cluster segment sizes & Mean statistics when k=2, k=3, k=4
    a. When K=2, the results are not consistent.



Clustering Criterion	Maximum Relative Change in Cluster Seeds	Improvement in Clustering Criterion	Segment Id	Frequency of Cluster	Root-Mean-Square Standard Deviation	Maximum Distance from Cluster Seed	Nearest Cluster	Distance to Nearest Cluster	airport_crpm_avg	biogenics_crpm_avg	cmv_crpm_avg	cmv_loco_crpm_avg	fires_crpm_avg	heavyduty_crpm_avg	lightduty_crpm_avg	np_10m_releaseheight_crpm_avg	np_low_releaseheight_crpm_avg	railyards_crpm_avg	rwc_crpm_avg	secondary_crpm_avg	total_crpm_avg
0.883886	0	0	1	1		0	2	12.54777	0.135103	0.030035	1	1	0.126974	1	1	0.417694	1	0.101361	0.098362	0.151103	1
0.883886	0	0	2	58	0.899259	6.815379	1	12.54777	0.124627	0.477593	0.021239	0.158375	0.326733	0.320122	0.18252	0.16684	0.154431	0.154885	0.16973	0.493983	0.301303

b. When K=3, the results are consistent.





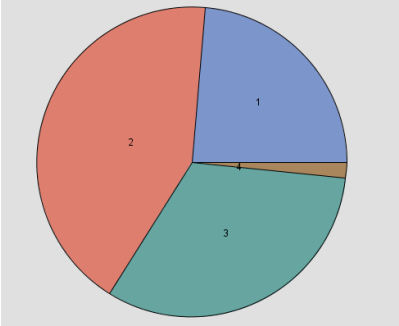
Clustering Criterion	Maximum Relative Change in Cluster Seeds	Improvement in Clustering Criterion	Segment Id	Frequency of Cluster	Root-Mean-Square Standard Deviation	Maximum Distance from Cluster Seed	Nearest Cluster	Distance to Nearest Cluster	airport_crpm_avg	biogenics_crpm_avg	cmv_crpm_avg	cmv_loco_crpm_avg	fires_crpm_avg	heavyduty_crpm_avg	lightduty_crpm_avg	np_10m_releaseheight_crpm_avg	np_low_releaseheight_crpm_avg	railyards_crpm_avg	rwc_crpm_avg	secondary_crpm_avg	total_crpm_avg
0.759401	0	0	1	1		0	2	11.42894	0.135103	0.030035	1	1	0.126974	1	1	0.417694	1	0.101361	0.098362	0.151103	1
0.759401	0	0	2	26	0.903305	5.578012	3	3.307241	0.214012	0.462232	0.041323	0.24144	0.289759	0.53385	0.286808	0.186351	0.263362	0.243524	0.245347	0.635584	0.436451
0.759401	0	0	3	32	0.662976	5.611084	2	3.307241	0.052002	0.490074	0.004921	0.090885	0.356775	0.146467	0.097786	0.150988	0.065925	0.082866	0.108291	0.378932	0.191495

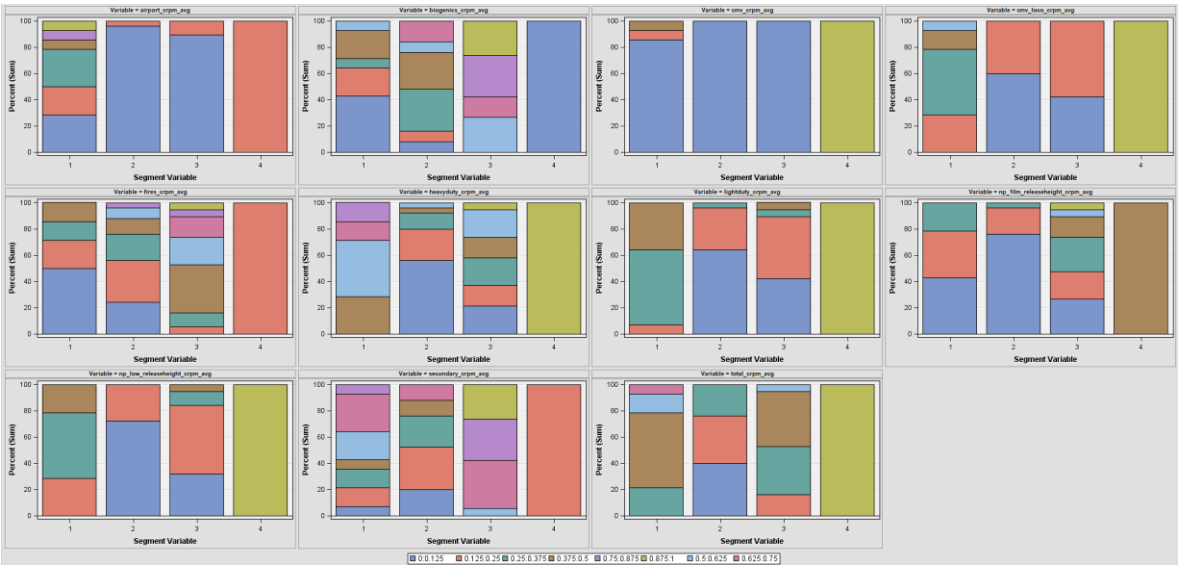
Variable Importance

Variable Name	Label	Number of Splitting Rules	Number of Surrogate Rules	Importance
total_crpm_avg		1	0	1.00000
np_low_releaseheight_crpm_avg		0	1	0.96551
heavyduty_crpm_avg		0	1	0.95669
cmv_loco_crpm_avg		0	1	0.94779
lightduty_crpm_avg		0	1	0.92057
airport_crpm_avg		0	1	0.89253

\

c. When K=4, additional validation is performed, but the results are satisfactory.





Clustering Criterion	Maximum Relative Change in Cluster Seeds	Improvement in Clustering Criterion	Segment Id	Frequency of Cluster	Root-Mean-Square Standard Deviation	Maximum Distance from Cluster Seed	Nearest Cluster	Distance to Nearest Cluster	airport_crpm_avg	biogenics_crpm_avg	cmv_crpm_avg	cmv_loco_crpm_avg	fires_crpm_avg	heavyduty_crpm_avg	lightduty_crpm_avg	np_10m_releaseheight_crpm_avg	np_low_releaseheight_crpm_avg	railyards_crpm_avg	rwc_crpm_avg	secondary_crpm_avg	total_crpm_avg
0.6644	0	0	1	14	0.90268	5.329858	3	3.983852	0.313865	0.225328	0.072984	0.312062	0.179882	0.585783	0.344589	0.16167	0.30166	0.349773	0.234858	0.466895	0.45219
0.6644	0	0	2	25	0.540404	3.830115	3	3.248094	0.050215	0.398017	0.00604	0.097315	0.261401	0.153575	0.109907	0.076835	0.070855	0.094698	0.097959	0.279204	0.172362
0.6644	0	0	3	19	0.684859	4.735047	2	3.248094	0.083098	0.768176	0.003111	0.125475	0.520903	0.343512	0.158645	0.289078	0.155916	0.090477	0.216176	0.796546	0.359783
0.6644	0	0	4	1	.	0	1	10.73348	0.135103	0.030035	1	1	0.126974	1	1	0.417694	1	0.101361	0.098362	0.151103	1

Variable Importance

Variable Name	Label	Number of Splitting Rules	Number of Surrogate Rules	Importance
airport_crpm_avg		0	3	1.00000
total_crpm_avg		0	2	0.98812
fires_crpm_avg		0	2	0.80462
secondary_crpm_avg		1	0	0.79360
np_10m_releaseheight_crpm_avg		1	1	0.77395
cmv_loco_crpm_avg		0	2	0.75622
np_low_releaseheight_crpm_avg		0	2	0.75622
biogenics_crpm_avg		0	1	0.75578
lightduty_crpm_avg		1	0	0.71384
cmv_crpm_avg		0	1	0.67015
heavyduty_crpm_avg		0	1	0.32480

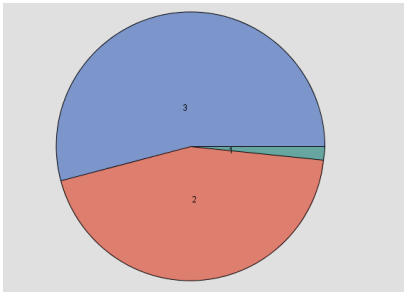
3. Cluster statistics when k=2, k=3, k=4

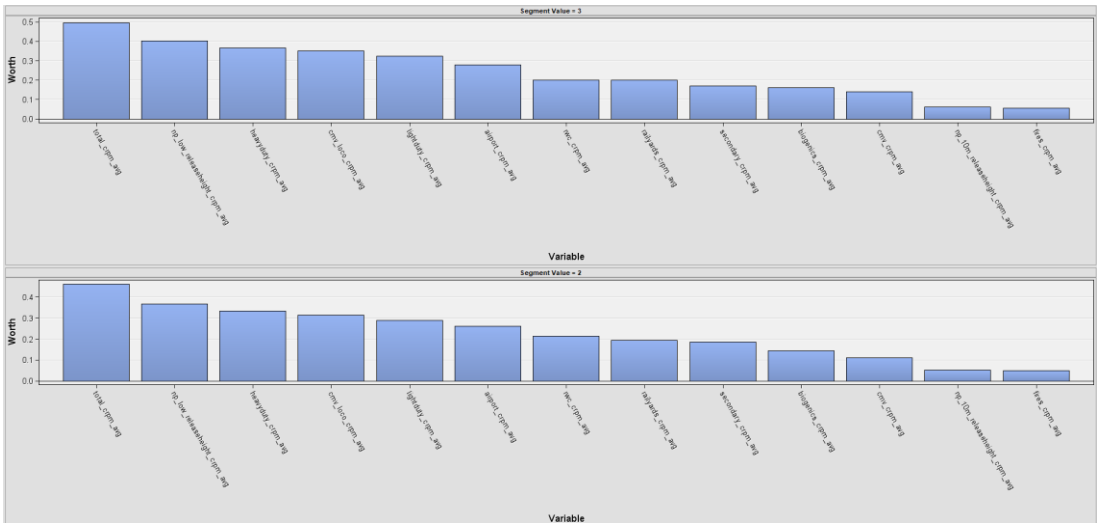
Cluster Statistics

Type of Observation	Segment Id	Statistic Applying Over All Variables	airport_crpm_avg	biogenics_crpm_avg	cmv_crpm_avg	cmv_loco_crpm_avg	fires_crpm_avg	heavyduty_crpm_avg	lightduty_crpm_avg	np_10m_releaseheight_crpm_avg	np_low_releaseheight_crpm_avg	railyards_crpm_avg	rwc_crpm_avg	secondary_crpm_avg	total_crpm_avg
DMDB FREQ			59	59	59	59	59	59	59	59	59	59	59	59	59
DMDB WEIGHT			59	59	59	59	59	59	59	59	59	59	59	59	59
DMDB MEAN			0.124804	0.470007	0.037829	0.17264	0.323348	0.331645	0.196376	0.171092	0.168763	0.153978	0.16852	0.488171	0.313145
DMDB STD		0	0.174298	0.273851	0.139864	0.158723	0.228214	0.25675	0.166296	0.175122	0.167304	0.231596	0.164823	0.28464	0.172254
LOCATION			0.124804	0.470007	0.037829	0.17264	0.323348	0.331645	0.196376	0.171092	0.168763	0.153978	0.16852	0.488171	0.313145
SCALE			0.174298	0.273851	0.139864	0.158723	0.228214	0.25675	0.166296	0.175122	0.167304	0.231596	0.164823	0.28464	0.172254
DMDB MIN			0	0	0	0	0	0	0	0	0	0	0	0	0
DMDB MAX			1	1	1	1	1	1	1	1	1	1	1	1	1
CRITERION		0.983886													
PSEUDO F		14.72299													
ERSQ		0.091659													
CCC		12.60482													
TOTAL STD		1	0.174298	0.273851	0.139864	0.158723	0.228214	0.25675	0.166296	0.175122	0.167304	0.231596	0.164823	0.28464	0.172254
WITHIN STD		0.899259	0.175815	0.269917	0.058167	0.115839	0.228708	0.243116	0.128895	0.173553	0.127085	0.233513	0.165998	0.283573	0.147555
RSQ		0.205276	6.123E-5	0.045271	0.830025	0.476548	0.012986	0.118847	0.409582	0.034778	0.432947	.0009053	0.003178	0.024595	0.278861
RSQ_RATIO		0.258298	6.124E-5	0.047418	4.88323	0.910395	0.013157	0.134877	0.693715	0.036031	0.763504	.0009061	0.003188	0.025215	0.386695
SEED	1	1	0.135103	0.030035	1	1	0.126974	1	1	0.417694	1	0.101361	0.098362	0.151103	1
SEED	2	58	0.124627	0.477593	0.021239	0.158375	0.326733	0.320122	0.18252	0.16684	0.154431	0.154885	0.16973	0.493983	0.301303
CLUS_MEAN	1		0.135103	0.030035	1	1	0.126974	1	1	0.417694	1	0.101361	0.098362	0.151103	1
CLUS_MEAN	2		0.124627	0.477593	0.021239	0.158375	0.326733	0.320122	0.18252	0.16684	0.154431	0.154885	0.16973	0.493983	0.301303
CLUS_STD	1														
CLUS_STD	2	0.899259	0.175815	0.269917	0.058167	0.115839	0.228708	0.243116	0.128895	0.173553	0.127085	0.233513	0.165998	0.283573	0.147555
CLUS_MIN	1		0.135103	0.030035	1	1	0.126974	1	1	0.417694	1	0.101361	0.098362	0.151103	1
CLUS_MIN	2		-2.5E-18	3.41E-17	-1.8E-18	-7.7E-18	-3.1E-18	-8.9E-18	1.76E-17	2.74E-18	9.46E-18	-1.4E-17	2.3E-17	1.35E-17	3.24E-17
CLUS_MAX	1		0.135103	0.030035	1	1	0.126974	1	1	0.417694	1	0.101361	0.098362	0.151103	1
CLUS_MAX	2		1	1	0.395267	0.575178	1	0.884429	0.499021	1	0.471234	1	1	1	0.654684
CLUS_FREQ	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
CLUS_FREQ	2	58	58	58	58	58	58	58	58	58	58	58	58	58	58

3.6.2.3 Results visualisation in SAS EM

1. Segment profiles
   a. K=3





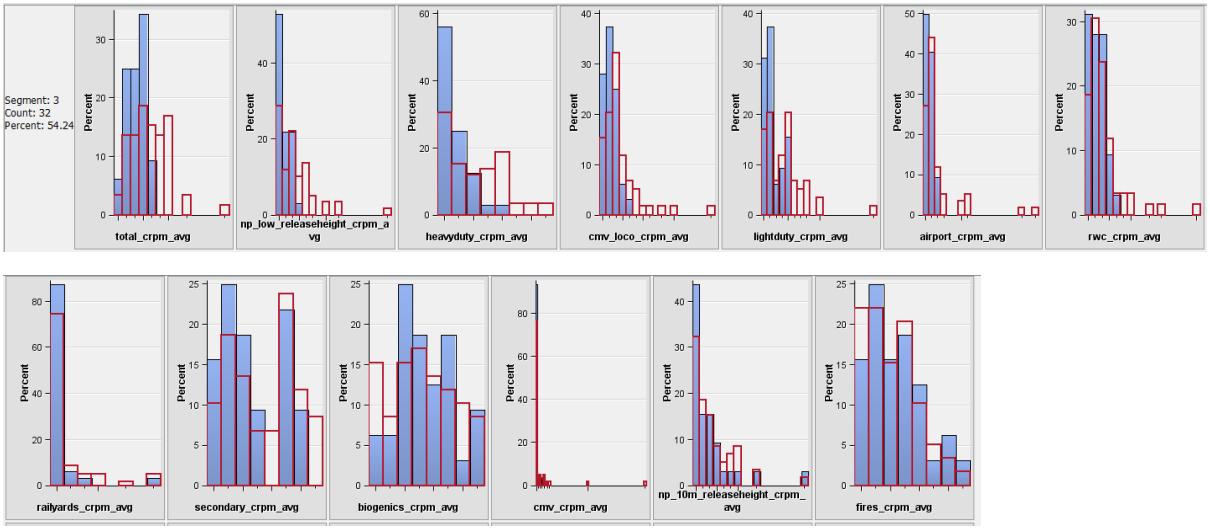
Frequencies: _SEGMENT_

Segment Variable	Segment Value	Frequency Count	Percent of Total Frequency
SEGMENT	3	32	54.2373
SEGMENT	2	26	44.0678
SEGMENT	1	1	1.6949

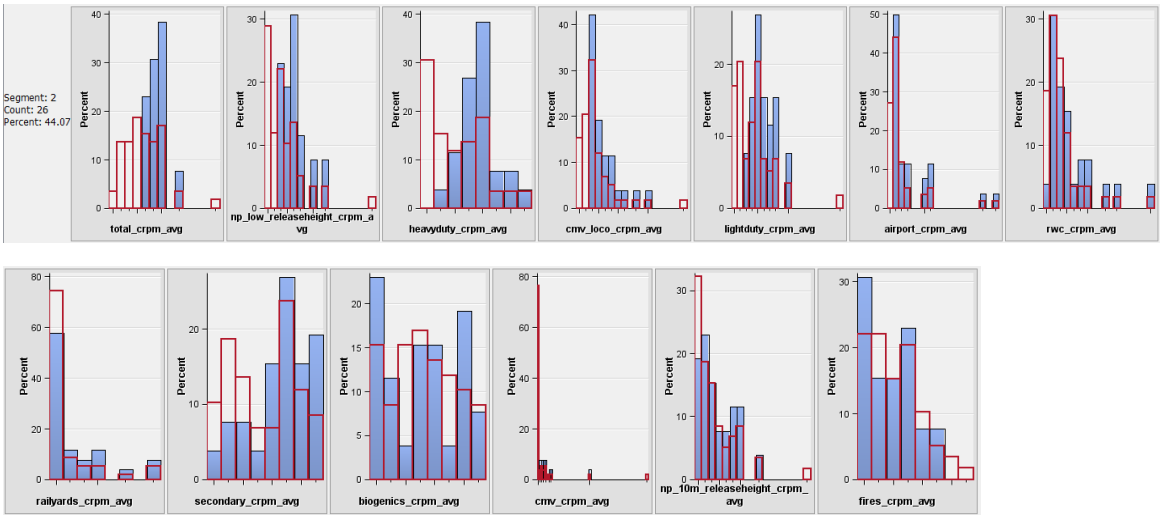
Variable: _SEGMENT_ Segment: 3 Count: 32
Decision Tree Importance Profiles

Variable	Worth	Rank
total_crpm_avg	0.49641	1
np_low_releaseheight_crpm_avg	0.40149	2
heavyduty_crpm_avg	0.36647	3
cmv_loco_crpm_avg	0.35005	4
lightduty_crpm_avg	0.32333	5
airport_crpm_avg	0.27946	6
rwc_crpm_avg	0.20020	7
railyards_crpm_avg	0.19963	8
secondary_crpm_avg	0.16957	9
biogenics_crpm_avg	0.16111	10
cmv_crpm_avg	0.13914	11
np_10m_releaseheight_crpm_avg	0.06282	12
fires_crpm_avg	0.05687	13

When K=3



When K=2



3.7 Results analysis and discussion

3.7.1 Result comparison between R and SAS EM

- In R, we could observe strong correlations between total cancer risk per million and cancer risk due to light duty vehicle emissions, heavy duty vehicle emissions, non-road locomotive emissions, and non-point sources with low release height.
- From the Correlogram and cluster dendrograms, we observe that San Francisco, Los Angeles, Alameda, San Diego, Sacramento, Siskiyou, San Joaquin, Colusa, etc are prominently visible and their distances are relatively possible to cluster, which can be discovered from various methods either by centroid linkage, complete linkage, or single linkage.
- The optimal number of clusters for the given model with all the variables is 2 with around 32% probability or silhouette width, whereas for 3 clusters the silhouette width is only 27%.
- If we carefully observe in the clusters within R, even during the overlap of the clusters, there seem to be no identified counties from cluster 1 however, there were counties from cluster 2 within the intersection. This significantly explains that the overlapped region's counties are the next emerging hotspots for high cancer risk.
- The component 1 and component 2 signify the first 2 principal components, and since we have multivariate data, it would be strenuous to look into individual variables and perform bivariate analysis. "64.94% of the point variability" explains that more than half of the information about the multivariate data is captured by this plot of components 1 and 2.
- In SAS, however, it was observed that 3 clusters is optimum but the variable importance is similar to R of course with adequate values of percentages explaining the variability of the data.
- Alternatively, the findings from 4 clusters explains that airport emissions-based cancer risk is highly important with highest number of surrogate rules followed by total cancer risk per million.

3.7.2 Critical findings

- The correlogram is the key identifier to locate where the cancer risks could be found densely, based on which the investigation had continued.

3.8 Conclusion

In conclusion, we can say that we performed clustering analysis successfully to identify the potentially high-risk zones, low-risk zones for cancer, and the overlapping region which can target the possibly rising risk within the low-risk zones/counties within California, USA.

4. Customer Sentiment Analysis for burger joints in Thailand

4.1 Abstract

This module of the project aims to identify the top 30 burger-serving restaurants in various locations across Thailand and analyse the customers' sentiment based on the feedback provided over the last few years using R and SAS Enterprise Miner tools. In this task, we create a scalable model that could classify emotions based on the reviews given by the customers, and we could scale it to pizza-chains, Indian restaurants, Chinese restaurants, ice cream parlours etc. To proceed with the Data mining process, we could use any of the Data mining algorithms like SEMMA or CRISP-DM. We used CRISP-DM methodology to perform Data mining for this text-mining task. An R markdown file has been used to prepare the k-means cluster model with the slidy presentation rendering an HTML output. The business or operational understanding and data understanding is first performed as part of the requirements gathering. Later we import the dataset and clean it as part of data preparation. To optimize the functionality and reduce the redundancy, we use R functions enabling to break down or decompose a problem into smaller chunks. In addition, the code can be reproducible and reusable, and it was prepared in a systematic, organised, robust and an efficient manner. 'tm', 'wordcloud', 'SnowballC' and 'syuzhet' libraries were used to perform this data mining task. We first train the model and discover the fitting, and if necessary, we perform tuning operation to improve the results. To identify the scales of goodness of fit within sentiment, we use NRC sentiment analysis. Plots derived from ggplot2 were used wherever necessary to showcase the quality and aesthetics of the graphs. We later utilise SAS Enterprise Miner as a secondary data mining tool where we produce process flow diagrams and parameterize the tasks and compare the results with R.

4.2 Introduction

4.2.1 Brief background of the task

S.No	Variable Name	Type	Brief Description
1	Hotel.Restaurant_name	**identifier**	Name of the hotel or restaurant
2	Review	*text*	The text-based field where we observe user's feedback

4.2.2 Formulation of the research question

The research question, simply put, is if we could recognize what sort of feedback the customers are giving for the burger joints within Thailand for which we first identify the burger joints alone using filtering function and extract only those reviews in which the word 'burger' is mentioned.

4.2.3 Justification: Why did I choose this topic/dataset?

N/A

4.3 Aim and Objective of the task

To identify the top 30 Burger Joints among the given tourist accommodations and analyse the sentiment by restaurant and produce overall emotional feedback for all the burger joints.

4.4 Brief Literature Review

Analysing customer feedback and responding with better service brings both satisfaction for customers and companies especially in the food industry, or for that matter in any industry. The traditional-based analysis is difficult to analyse, there are some challenges to overcome this problem. Some methods for analysing feelings, such as prediction of user subjects, polarity of feelings scores, qualitative analysis and a large data mining application, cross-domain classification of feelings, identification of emotional differences, meaning and theme detection, classification of hashtag sentiment rates, sales forecasts, etc. are used. It also briefly addressed the complexities of sentimental

analytics to do the job. Some of the challenges such as parallel computing for massive data, sarcasm, grammatically incorrect words, review the author's segmentation, handling noise, and dynamism. The consumer can compare products according to the people's reviews on these products. So, for making this more successful they have produced supervised techniques for the consumer reviews. There are two types of methods are mentioned that is, opinion mining and sentiment analysis.

An R package for the extraction of sentiment and sentiment-based plot arcs from text. The name "Syuzhet" comes from the Russian Formalist Victor Shklovsky and Vladimir Propp who divided narrative into two components, the "Fabula" and "Syuzhet". Syuzhet refers to the "device" or technique of a narrative whereas fabula is the chronological order of events . The package comes with four sentiment dictionaries and provides a method for accessing the robust, but computationally expensive, sentiment extraction tool developed in the NLP group at Stanford. Syuzhet incorporates four dictionaries:

- The default "Syuzhet" lexicon was developed in the Nebraska Literary Lab under the direction of Matthew L. Jockers
- The "afinn" lexicon was developed by Finn Arup Nielsen as the AFINN WORD DATABASE.
- The "bing" lexicon was developed by Minqing Hu and Bing Liu as OPINION LEXICON.
- The "nrc" lexicon was developed by Mohammad, Saif M. and Turney, Peter D.

NRC is used to calculate the existence of eight different emotion and the corresponding valance within text file.

4.5 Explanation and preparation of datasets

4.5.1 Description of the dataset

- **Data Source** - Link (tourist_accommodation_reviews.csv)
- **Additional data used** - Positive and Negative lexicon in .txt format

- The dataset contains various reviews for restaurants across the landscape of Thailand with a restaurant ID, reviewed date, location where the restaurant is situated and the Hotel/restaurant name.

- **Nominal Variables** -
  - ID: The restaurant ID that uniquely identifies the restaurant.
  - Location: Geographical city / town where the restaurant is present.
  - Hotel/restaurant name: Self-explanatory variable.
  - Review: The customer feedback received in the form of text mentioning good / bad things in a restaurant.
  - Reviewed Date: Date on which the review was provided by the customers. (can be made ordinal)

- **Environment setup :– The following libraries Installed and activated**
  - dplyr : Data manipulations
  - tidyverse : Data science tasks
  - skimr : Statistical summary
  - tm : Text mining
  - SnowballC :Text stemming
  - wordcloud : Word-cloud generator
  - RColorBrewer : color palettes
  - syuzhet : Sentiment analysis
  - ggplot2 : Plotting graphs

Steps performed in R:

1. Setup the working directory using setwd(<filepath>).
2. Import the positive and negative lexicon into R.
3. Import Tourist Accommodation reviews file into R using read.csv() along with the header.

```
# Step 3 - Data Acquisition
# 3.1 - Setup Lexicon for Sentiment Analysis
p_lex <- read.csv("Sentiment Analysis Data/positive-lexicon.txt")
n_lex <- read.csv("Sentiment Analysis Data/negative-lexicon.txt")

# 3.2 - Import the Tourist Accommodation reviews file into R
df_revs_raw <- read.csv("tourist_accommodation_reviews.csv", header= TRUE)
```

4. Inspect the positive and negative lexicon – top 6 rows

```
# 3.3 - Inspect lexicon
head(p_lex)
```

```
##           a.
## 1     abound
## 2    abounds
## 3  abundance
## 4   abundant
## 5 accessable
## 6 accessible
```

```
head(n_lex)
```

```
##     X2.faced
## 1    2-faces
## 2   abnormal
## 3    abolish
## 4 abominable
## 5 abominably
## 6  abominate
```

5. Inspect the positive and negative lexicon – bottom 6 rows

```
tail(p_lex)
```

```
##            a.
## 2000      yay
## 2001 youthful
## 2002     zeal
## 2003   zenith
## 2004     zest
## 2005    zippy
```

```
tail(n_lex)
```

```
##       X2.faced
## 4777    zapped
## 4778      zaps
## 4779    zealot
## 4780   zealous
## 4781 zealously
## 4782    zombie
```

6. Inspect the variable names of acquired dataset – tourist accommodation reviews

```
# 3.4 - Inspect the acquired data
names(df_revs_raw)
```

```
## [1] "ID"                    "Review.Date"          "Location"
## [4] "Hotel.Restaurant.name" "Review"
```

7. Observe the top 6 rows

```
head(df_revs_raw)
```

```
##            ID             Review.Date Location        Hotel.Restaurant.name
## 1 rn579778340     Reviewed 1 week ago     Kathu Thong Dee The Kathu Brasserie
## 2 rn576350875    Reviewed 3 weeks ago     Kathu Thong Dee The Kathu Brasserie
## 3 rn574921678    Reviewed 4 weeks ago     Kathu Thong Dee The Kathu Brasserie
## 4 rn572905503 Reviewed April 12, 2018     Kathu Thong Dee The Kathu Brasserie
## 5 rn572364712 Reviewed April 10, 2018     Kathu Thong Dee The Kathu Brasserie
## 6 rn572308369  Reviewed April 9, 2018     Kathu Thong Dee The Kathu Brasserie
##
Review
## 1
Just been for sunday roast lamb and beef truly excellent,11out of 10\nCoudnt fault it one bit meat was so tender
## 2                    Quietly set off the main road, nice atmosphere. Immaculate and friendly service. But the
real reason to go is the food.\nFood of this quality, at this price, is absolutely remarkable. My new fav restaurant
in Phuket, and probably beyond.More
## 3           I made a reservation for a birthday two days in advance assuming we would have a quiet table and a
nice dinner. Upon arriving, the staff set up a nice table for us and gave us incredible service. It was unlike other
cookie-cutter restaurants we...More
```

8. Observe the bottom 6 rows

```
tail(df_revs_raw)
```

```
##               ID           Review.Date Location Hotel.Restaurant.name
## 53639 rn163047718 Reviewed June 5, 2013    Patong               Bite in
## 53640 rn162368197 Reviewed May 29, 2013    Patong               Bite in
## 53641 rn161843734 Reviewed May 25, 2013    Patong               Bite in
## 53642 rn161734077 Reviewed May 24, 2013    Patong               Bite in
## 53643 rn161218072 Reviewed May 19, 2013    Patong               Bite in
## 53644 rn161212765 Reviewed May 19, 2013    Patong               Bite in
##
Review
## 53639                                                                                    we came here for lunch with
our family.\nRestaurant nice and cozy.\nFood was great and big.\nstaffs are nice as well. Perfect lication in patong
Jungcrylon shopping center.Bravo!!Must try!!!!
## 53640 I love this small restaurant, for the great food, and the extreme friendliness of its staff.\nWell located
in Jungceylon shopping, we went there during the rain, and enjoy the free internet, good coffee.\nThe Manager & the
owner are 2 friendly ladies, who speaks...More
## 53641                  We stopped at this restaurant after shopping at Jung Ceylon. It's close to main "square" of
the mall, so you don't need to walk far. Burgers were delicious, but I should say separate word about desserts. I
would never expect to have such a tasty...More
## 53642
Great times ,This is one of the best restaurant in Phuket town. The owners and staffs are very friendly and helpful.
Many selections of foods. Great food!
```

9. Summarize the raw dataset – all character variables.

```
summary(df_revs_raw)
```

```
##       ID             Review.Date          Location         Hotel.Restaurant.name
##  Length:53644       Length:53644       Length:53644       Length:53644
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##     Review
##  Length:53644
##  Class :character
##  Mode  :character
```

10. Check for the structure of the raw dataset.

```
str(df_revs_raw)
```

```
## 'data.frame':    53644 obs. of  5 variables:
##  $ ID                  : chr  "rn579778340" "rn576350875" "rn574921678" "rn572905503" ...
##  $ Review.Date         : chr  "Reviewed 1 week ago " "Reviewed 3 weeks ago " "Reviewed 4 weeks ago " "Reviewed
April 12, 2018 " ...
##  $ Location            : chr  " Kathu" " Kathu" " Kathu" " Kathu" ...
##  $ Hotel.Restaurant.name: chr  "Thong Dee The Kathu Brasserie" "Thong Dee The Kathu Brasserie" "Thong Dee The
Kathu Brasserie" "Thong Dee The Kathu Brasserie" ...
##  $ Review              : chr  "Just been for sunday roast lamb and beef truly excellent,11out of 10\nCoudnt fault
it one bit meat was so tender" "Quietly set off the main road, nice atmosphere. Immaculate and friendly service. But
the real reason to go is t"| __truncated__ "I made a reservation for a birthday two days in advance assuming we would
have a quiet table and a nice dinner."| __truncated__ "We visit here regularly and never fail to be impressed by the
quality and presentation of the food. We have tri"| __truncated__ ...
```

11. Observing the dimensionality – 53,644 rows and 5 columns

```
dim(df_revs_raw)
```

```
## [1] 53644     5
```

12. Duplicate (to lowercase) the “review” column - to transform it later
13. Identify the “burger joints” & Filter them out as a subset
14. Inspect the new subset

```
# 4.1 Duplicate (to lowercase) the "review" column - to transform it later
df_revs_raw$Review_TRF <- tolower(df_revs_raw$Review)

# 4.2 Identify the "burger joints" & Filter them out as a subset
df_revs_raw$Is_Burg_Joint <- grepl(c("burg"), df_revs_raw$Review_TRF)
df_burg <- subset(df_revs_raw, Is_Burg_Joint==TRUE)

# 4.3 - Inspect the new subset
names(df_burg)
```

```
## [1] "ID"                  "Review.Date"          "Location"
## [4] "Hotel.Restaurant.name" "Review"             "Review_TRF"
## [7] "Is_Burg_Joint"
```

15. Inspect the subset data to identify burger joints based on the reviews.

```
head(df_burg)
```

```
##             ID             Review.Date Location  Hotel.Restaurant.name
## 201 rn580748664     Reviewed 4 days ago     Rawai Green Tamarind Kitchen
## 202 rn579783154     Reviewed 1 week ago     Rawai Green Tamarind Kitchen
## 203 rn579651071     Reviewed 1 week ago     Rawai Green Tamarind Kitchen
## 204 rn574834726    Reviewed 4 weeks ago     Rawai Green Tamarind Kitchen
## 205 rn574713927    Reviewed 4 weeks ago     Rawai Green Tamarind Kitchen
## 206 rn573036147 Reviewed April 13, 2018     Rawai Green Tamarind Kitchen
##
Review
## 201         If you want the best burger in Phuket, look no further. Some people will tell you that the best
burger is at a resort, or that Hooters does a really good one, or that they know this 5 star restaurant that has an
amazing burger,...More
## 202      I have been here a couple of times now,and have had mixed expieriences. I seems to be that it has to be
the right person bbq'ing the food! First time the burgers and bacon was made right,but the next time the burger fell
apart when...More
## 203                                                                                                        This
place is amazing! Burgers are delicious! High recommended! I will come again. When I will be off my diet<f0><U+009F>
<U+0098><U+0084>
```

16. Summary of the data subset. Total number of reviews / rows observed = 1454

```
summary(df_burg)
```

```
##       ID              Review.Date          Location          Hotel.Restaurant.name
##  Length:1454        Length:1454        Length:1454        Length:1454
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##     Review            Review_TRF        Is_Burg_Joint
##  Length:1454        Length:1454        Mode:logical
##  Class :character   Class :character   TRUE:1454
##  Mode  :character   Mode  :character
```

```
dim(df_burg)
```

```
## [1] 1454    7
```

17. Inspect the review column to validate the search

```
# 4.4 - Inspect the review column to validate the search
head(df_burg$Review)
```

```
## [1] "If you want the best burger in Phuket, look no further. Some people will tell you that the best burger is at a resort, or that Hooters does
a really good one, or that they know this 5 star restaurant that has an amazing burger,...More"
## [2] "I have been here a couple of times now,and have had mixed expieriences. I seems to be that it has to be the right person bbq'ing the food!
First time the burgers and bacon was made right,but the next time the burger fell apart when...More"
## [3] "This place is amazing! Burgers are delicious! High recommended! I will come again. When I will be off my diet<f0><U+009F><U+0098><U+0084>"
## [4] "This is a must when in Phuket. Their burgers rank top worldwide. The service is fast and friendly, food is served fresh hot and fast - very
rare for Thailand. They have a great varié of burgers, also other dishes, but why come here for...More"
## [5] "The reviews speak for themselves. This is mainly the reason we went here and they didn't disappoint. The burgers are out of this world and
cooked in front of you. Very reasonable price for what you get. Only thing I would say is share a...More"
## [6] "Fancied a change from Thai food and had been told about this place before. Excellent recommendation, burgers were huge and delicious - our
whole group loved the place, some so much they went back a few nights later!"
```

18. Creating a function for sentimental analysis

```
# 4.5 - Creating a function for sentimental analysis
f_sentiment_burg <- function(sten_corpus)
{

  # 4.5.1 - Variable initialization
  tot_pos_cnt <- 0
  tot_neg_cnt <- 0
  pos_cnt_vec <- c()
  neg_cnt_vec <- c()
  size <- length(stem_corpus)

  # 4.5.2 - Loop to identify positive and negative lexicon
  for(i in 1:size)
  {
    # List the words by splitting them by spaces
    corpus_words <- list(strsplit(stem_corpus[[i]]$content, split = " "))

    # Segregate Positive and Negative Counts based on Lexicon
    pos_cnt <-length(intersect(unlist(corpus_words), unlist(p_lex)))
    neg_cnt <-length(intersect(unlist(corpus_words), unlist(n_lex)))

    # Overall positive & negative counts
    tot_pos_cnt <- tot_pos_cnt + pos_cnt
    tot_neg_cnt <- tot_neg_cnt + neg_cnt
  }

  # 4.5.3 - Total Count is the sum of positive & negative counts
  tot_pos_cnt
  tot_neg_cnt
  tot_cnt <- tot_pos_cnt + tot_neg_cnt

  # 4.5.4 - Formulate the Overall Positive % & Overall Negative %
  overall_pos_percent <- (tot_pos_cnt*100)/tot_cnt
  overall_neg_percent <- (tot_neg_cnt*100)/tot_cnt

  overall_pos_percent  # Inspect

  # 4.5.5 - Create a data frame to be utilized at a later stage
  df<-data.frame(Positive=num(tot_pos_cnt),
                 Negative=num(tot_neg_cnt),
                 Total = num(tot_cnt),
                 Overall_Rating = overall_pos_percent)
  return(df)
}
```

19. Creating a function for word cloud –

```
# 4.6 - Creating a function for Word Cloud
f_wc <- function(stem_corpus, v_jt)
{
print(v_jt)
wordcloud(stem_corpus,
          min.freq = 3,
          colors=brewer.pal(8, "Dark2"),
          random.color = TRUE,
          max.words = 100)
}
```

4.5.2 Identify independent dependent variables (if any)

- N/A

4.5.3 Data Pre-processing steps

1. Identify the unique burger joints and sort them by alphabetical order.

```r
# Step 5 - Data Preparation & Cleanup
# 5.1 - Identify the unique burger joints and sort them by alphabetical order.
v_burgJoints <- sort(unique(df_burg$Hotel.Restaurant.name))
head(v_burgJoints)
```

```
## [1] "2gether Restaurant"                "360 ° Bar"
## [3] "44 Thaikitchen \"KATA FOOD COURT\"" "9' Sea Breeze"
## [5] "After Beach Bar"                   "Anchor Inn"
```

```r
tail(v_burgJoints)
```

```
## [1] "Wine Connection Deli & Bistro - Central Phuket"
## [2] "Wine Connection Deli & Bistro - Chalong, Phuket"
## [3] "Wine Connection Deli & Bistro - Jungceylon, Patong Beach"
## [4] "YamThai Restaurant"
## [5] "Yorkshire Hotel Restaurant"
## [6] "You and Me Patong"
```

2. Variable Initialization

```r
# 5.2 - Variable Initialization
df_final = data.frame( burg_joint=rep(0, 10), Positive=rep(0,10), Negative=rep(0,10), Total=rep(0,10), Overall_Rating=rep(0,10))

iter = 0 # Loop counter variable
```

3. Loop using Burger Sentiment function ("f_sentiment_burg()")

```r
# 5.3 - Loop using Burger Sentiment function ("f_sentiment_burg()")
for (burg_joint in v_burgJoints) {

  # 5.3.1 - Loop counter variable
  iter = iter+1

  # 5.3.2 - Create text vectors
  vec_burg_reviews <- subset(df_burg$Review_TRF,df_burg$Hotel.Restaurant.name==burg_joint)

  # gsub() function replaces all matches of a string

  # 5.3.3 - Remove hyperlinks from the reviews (if any)
  vec_burg_reviews <- gsub("http\\S+\\s*", "", vec_burg_reviews)

  # 5.3.4 - Remove punctuation from the reviews
  vec_burg_reviews <- gsub("[[:punct:]]", "", vec_burg_reviews)

  # 5.3.5 - Remove numerical values from the reviews
  vec_burg_reviews <- gsub("[[:digit:]]", "", vec_burg_reviews)

  # 5.3.6 - Remove leading blank spaces at the beginning from the reviews
  vec_burg_reviews <- gsub("^ ", "", vec_burg_reviews)

  # 5.3.7 - Remove blank spaces at the end from the reviews
  vec_burg_reviews <- gsub(" $", "", vec_burg_reviews)

  # 5.3.8 - Replace "\n" word with a space from the reviews
  vec_burg_reviews <- gsub("\n", " ", vec_burg_reviews)

  # 5.3.9 - Converting the text vectors to corpus
  corpus_burg_revs <- Corpus(VectorSource(vec_burg_reviews))

  # 5.3.10 - Clean up corpus by removing stop words and Whitespace
  corpus_burg_revs <- tm_map(corpus_burg_revs, removeWords,stopwords("english"))
  corpus_burg_revs <- tm_map(corpus_burg_revs, stripWhitespace)

  # 5.3.11 - Stem the words to their root of all reviews present in the corpus
  stem_corpus_burg_revs <- tm_map(corpus_burg_revs, stemDocument)

  # 5.3.12 - Utilize the sentiment analysis function for the stemmed corpus and append it to a dataframe
  df_final[iter,] <- c(burg_joint, f_sentiment_burg(stem_corpus_burg_revs))
}
```

4. Inspect the outcomes for each restaurant

```
# 5.4 - Inspect the outcomes for each restaurant
head(df_final)
```

```
##                         burg_joint Positive Negative Total Overall_Rating
## 1                2gether Restaurant        3        0     3      100.00000
## 2                         360 º Bar        2        0     2      100.00000
## 3 44 Thaikitchen "KATA FOOD COURT"        6        2     8       75.00000
## 4                     9' Sea Breeze       88       16   104       84.61538
## 5                   After Beach Bar        3        3     6       50.00000
## 6                        Anchor Inn        1        1     2       50.00000
```

```
tail(df_final)
```

```
##                                                 burg_joint Positive Negative
## 247           Wine Connection Deli & Bistro - Central Phuket        7        3
## 248          Wine Connection Deli & Bistro - Chalong, Phuket        1        2
## 249 Wine Connection Deli & Bistro - Jungceylon, Patong Beach        2        0
## 250                                       YamThai Restaurant        1        0
## 251                                Yorkshire Hotel Restaurant        8        6
## 252                                         You and Me Patong        1        0
##     Total Overall_Rating
## 247    10       70.00000
## 248     3       33.33333
## 249     2      100.00000
## 250     1      100.00000
## 251    14       57.14286
## 252     1      100.00000
```

```
names(df_final)
```

```
## [1] "burg_joint"     "Positive"       "Negative"       "Total"
## [5] "Overall_Rating"
```

5. Sort by total reviews & pick top 30 restaurants with highest total reviews

```
# 5.5 - Sort by total reviews & pick top 30 restaurants with highest total reviews
df_final <- df_final[order(-df_final$Total),]
df_pop30 <- df_final[1:30,]
head(df_pop30)
```

```
##                  burg_joint Positive Negative Total Overall_Rating
## 141     New York Burger Co.      231       37   268       86.19403
## 113      Le Brooklyn Patong      208       16   224       92.85714
## 30  Burger House Kata Beach      198       23   221       89.59276
## 78   Green Tamarind Kitchen      191       28   219       87.21461
## 69                Flip Side      152       25   177       85.87571
## 216  The Frying Kiwi Eatery      143       25   168       85.11905
```

6. Setup rownames as Restaurant names

7. Pivot the Positive & Negative reviews for comparative analysis

```r
# 5.6 - Setup rownames as Restaurant names
rownames(df_pop30) <- df_pop30$burg_joint

# 5.7 - Pivot the Positive & Negative reviews for comparative analysis
df_pop30_comp <- df_pop30[,1:3] %>%
  pivot_longer(
    cols = ends_with("tive"),
    names_to = "Review Type",
    values_to = "Review Count",
    values_drop_na = TRUE
  )
head(df_pop30_comp)
```

```
## # A tibble: 6 x 3
##   burg_joint              `Review Type` `Review Count`
##   <chr>                   <chr>                 <dbl>
## 1 New York Burger Co.     Positive                231
## 2 New York Burger Co.     Negative                 37
## 3 Le Brooklyn Patong      Positive                208
## 4 Le Brooklyn Patong      Negative                 16
## 5 Burger House Kata Beach Positive                198
## 6 Burger House Kata Beach Negative                 23
```

```r
tail(df_pop30_comp)
```

```
## # A tibble: 6 x 3
##   burg_joint              `Review Type` `Review Count`
##   <chr>                   <chr>                 <dbl>
## 1 La Boucherie - Chalong  Positive                 20
## 2 La Boucherie - Chalong  Negative                  4
## 3 Climax on Bangla        Positive                 15
## 4 Climax on Bangla        Negative                  4
## 5 Joe's Downstairs        Positive                 18
## 6 Joe's Downstairs        Negative                  1
```

```r
names(df_pop30_comp)
```

```
## [1] "burg_joint"   "Review Type"  "Review Count"
```

8. Subset & sort top 30 reviews for Word cloud

```
# 5.8 - Subset & sort top 30 reviews for Word cloud
# Subset
df_pop30_revs <- df_burg %>% filter(Hotel.Restaurant.name %in% df_pop30$burg_joint)
dim(df_pop30_revs)
```

```
## [1] 873   7
```

```
# Sort
v_top30_burg_jts <- sort(unique(df_pop30_revs$Hotel.Restaurant.name))
v_top30_burg_jts
```

```
##  [1] "9' Sea Breeze"
##  [2] "Ann's Kitchen Bar and Grill"
##  [3] "Benny's American Bar & Grill"
##  [4] "Bill Bentley Pub"
##  [5] "Bondi Aussie Bar & Grill Phuket"
##  [6] "Buffalo Steak House - Karon Beach"
##  [7] "Burger House Kata Beach"
##  [8] "Climax on Bangla"
##  [9] "EAT. bar & grill"
## [10] "ELLA Bar & Bistro"
## [11] "Flip Side"
## [12] "Full Moon Brewworks - Microbrewery & Lobs n' Roll"
## [13] "Green Tamarind Kitchen"
## [14] "Grill Bill"
## [15] "Hakan's Bar & Restaurant"
## [16] "Happy Days"
## [17] "i-Kroon Cafe"
## [18] "Joe's Downstairs"
## [19] "La Boucherie - Chalong"
## [20] "Le Brooklyn Patong"
## [21] "Legends Sports Bar & Grill"
## [22] "Lucky 13 Sandwich Patong"
## [23] "New York Burger Co."
## [24] "Nicky's Handlebar"
## [25] "Outdoor Restaurant"
## [26] "Peony Cafe & Restaurant"
## [27] "Rider Cafe"
## [28] "Rustic - Eatery & Bar"
## [29] "The Frying Kiwi Eatery"
## [30] "The Sandwich Club"
```

9. Loop for Word Clouds (top 30 only)

```
# 5.9 - Loop for Word Clouds (top 30 only)
for (burg_joint in v_top30_burg_jts) {

  # 5.9.1 - Create text vectors
  vec_burg_reviews <- subset(df_burg$Review_TRF,df_burg$Hotel.Restaurant.name==burg_joint)

  # gsub() function replaces all matches of a string

  # 5.9.2 - Remove hyperlinks from the reviews (if any)
  vec_burg_reviews <- gsub("http\\S+\\s*", "", vec_burg_reviews)

  # 5.9.3 - Remove punctuation from the reviews
  vec_burg_reviews <- gsub("[[:punct:]]", "", vec_burg_reviews)

  # 5.9.4 - Remove numerical values from the reviews
  vec_burg_reviews <- gsub("[[:digit:]]", "", vec_burg_reviews)

  # 5.9.5 - Remove leading blank spaces at the beginning from the reviews
  vec_burg_reviews <- gsub("^ ", "", vec_burg_reviews)

  # 5.9.6 - Remove blank spaces at the end from the reviews
  vec_burg_reviews <- gsub(" $", "", vec_burg_reviews)

  # 5.9.7 - Replace "\n" word with a space from the reviews
  vec_burg_reviews <- gsub("\n", " ", vec_burg_reviews)

  # 5.9.8 - Converting the text vectors to corpus
  corpus_burg_revs <- Corpus(VectorSource(vec_burg_reviews))

  # 5.9.9 - Clean up corpus by removing stop words and Whitespace
  corpus_burg_revs <- tm_map(corpus_burg_revs, removeWords,stopwords("english"))
  corpus_burg_revs <- tm_map(corpus_burg_revs, stripWhitespace)

  # 5.9.10 - Stem the words to their root of all reviews present in the corpus
  stem_corpus_burg_revs <- tm_map(corpus_burg_revs, stemDocument)

  # 5.9.11 - Execute Word Cloud function for the Stem Corpus
  f_wc(stem_corpus_burg_revs, burg_joint)
}
```

```
## [1] "9' Sea Breeze"
```


```
## [1] "Ann's Kitchen Bar and Grill"
```

```
## [1] "Benny's American Bar & Grill"
```


```
## [1] "Bill Bentley Pub"
```


```
## [1] "Bondi Aussie Bar & Grill Phuket"
```


```
## [1] "Buffalo Steak House - Karon Beach"
```


```
## [1] "Burger House Kata Beach"
```


```
## [1] "Climax on Bangla"
```


```
## [1] "EAT. bar & grill"
```


```
## [1] "ELLA Bar & Bistro"
```

[1] "Flip Side"

[1] "Full Moon Brewworks - Microbrewery & Lobs n' Roll"

[1] "Green Tamarind Kitchen"

[1] "Grill Bill"

[1] "Hakan's Bar & Restaurant"

[1] "Happy Days"

[1] "i-Kroon Cafe"

[1] "Joe's Downstairs"

[1] "La Boucherie - Chalong"

[1] "Le Brooklyn Patong"

[1] "Legends Sports Bar & Grill"

[1] "Lucky 13 Sandwich Patong"

[1] "New York Burger Co."

[1] "Nicky's Handlebar"

[1] "Outdoor Restaurant"

[1] "Peony Cafe & Restaurant"

```
## [1] "Rider Cafe"
```


```
## [1] "Rustic - Eatery & Bar"
```


```
## [1] "The Frying Kiwi Eatery"
```


```
## [1] "The Sandwich Club"
```

4.5.4 Assumptions (if any)

- We assume that the burger joints are only the burger-offering restaurants based on the reviews. There could be others too for which there were no reviews, perhaps.

4.6 Task: Text Mining

4.6.1 Data Exploration and Attribute Visualization in R

4.6.1.1 Model Building in R

1. Highest Overall ratings

I. Highest Overall Ratings

```
options(digits=2)
df_pop30 %>% mutate(burg_joint = fct_reorder(burg_joint, desc(Overall_Rating))) %>%
  ggplot(aes(burg_joint,Overall_Rating))+
  geom_col() +
  labs(title="Top 30 Burger Joints by Highest Overall Positive Ratings (%)")+
  theme(plot.title = element_text(hjust = 0.5, size = 18, face = 'bold'))+
  xlab("Name of the Restaurant")+
  ylab("Overall Positive Rating in %")+
  theme(axis.title.x = element_text(size=14, face = 'bold'))+
  theme(axis.title.y = element_text(size=14, face = 'bold'))+
  geom_text(aes(label = signif(Overall_Rating, digits = 3)), nudge_y = 4, size = 3)+
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1))
```

2. Comparative analysis – positive vs. negative reviews

2. Comparative analysis (Positive vs. Negative reviews)

```
# using 100% stacked bar graph
ggplot(df_pop30_comp,
       aes(fill=df_pop30_comp$`Review Type`,
           y=df_pop30_comp$`Review Count`,
           x=df_pop30_comp$burg_joint)) +
  geom_bar(position="fill", stat="identity") +
  xlab("Name of the Restaurant")+
  ylab("Positive vs. Negative reviews")+
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1)) +
  ggtitle("Top 30 Burger joints in Thailand by Positive & Negative Reviews") +
  theme(axis.title.x = element_text(size=14, face = 'bold'))+
  theme(axis.title.y = element_text(size=14, face = 'bold'))+
  theme(plot.title = element_text(hjust = 0.5, size = 20, face = 'bold'))+
  guides(fill=guide_legend(title="Review Type"))
```

3. However, these overall positive ratings may not fully justify our cause, since the number of ratings also need to be checked first. So we need to check total reviews too.

3. Total Reviews

```
# 6.3.1 - Custom Bar plot (ability to rotate labels)
# Arguments - data - dataframe ; col - column to plot ; lab_vec - Labels vector, aor - Angle of Rotation
custom_x <- function(data, col, lab_vec, aor, title, ylabs) {
  plt <- barplot(data[[col]], main = title, ylab = ylabs, col='goldenrod1', xaxt="n")
  text(plt, par("usr")[3], labels = lab_vec, srt = aor, adj = c(1.1,1.1), xpd = TRUE, cex=0.6)
}

# 6.3.2 - Plotting the top 30 burger joints using number of reviews
custom_x(df_pop30, 'Total', row.names(df_pop30), 45, "Top 30 Burger Joints in Thailand by total number of reviews", "Total number of reviews")
```

4. Comparative analysis for total reviews –

4. Comparative analysis for Total Reviews

```
ggplot(df_pop30_comp, aes(fill=df_pop30_comp$`Review Type`,
                          y=df_pop30_comp$`Review Count`,
                          x=df_pop30_comp$burg_joint)) +
  geom_bar(position="stack", stat="identity")+
  xlab("Name of the Restaurant")+
  ylab("Positive vs. Negative reviews")+
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1)) +
  ggtitle("Top 30 Burger joints in Thailand by Positive & Negative Reviews") +
  theme(axis.title.x = element_text(size=14, face = 'bold'))+
  theme(axis.title.y = element_text(size=14, face = 'bold'))+
  theme(plot.title = element_text(hjust = 0.5, size = 20, face = 'bold'))+
  guides(fill=guide_legend(title="Review Type"))
```

5. Comparison using combo chart

5. Comparison using Combo chart (Bar+Line graph)

```
ggplot(df_pop30) +
  geom_col(aes(x = burg_joint, y = Total),
           size = 1, color = "white", fill = "steelblue") +
  geom_line(aes(x = burg_joint, y = Overall_Rating),
            size = 1, color="red", group = 1) +
  xlab("Name of the Restaurant")+
  ylab("Number of Reviews")+
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1)) +
  ggtitle("Top 30 Burger joints in Thailand by Positive & Negative Reviews") +
  theme(axis.title.x = element_text(size=14, face = 'bold'))+
  theme(axis.title.y = element_text(size=14, face = 'bold'))+
  theme(plot.title = element_text(hjust = 0.5, size = 20, face = 'bold'))+ geom_hline(yintercept=90, linetype="dashed", color = "red")
```

4.6.1.2 Model Assessment in R

1. Syuzhet, bing and affin methods are used to get the sentiments from these feedbacks of the top 30 burger joints.

```
# 7.1 - Use various methods (scales) to create vectors
# 7.1.1 - Syuzhet
burg_syuzhet <- get_sentiment(df_burg$Review, method="syuzhet")
head(burg_syuzhet)
```

```
## [1] 2.4 0.7 1.2 1.9 0.6 3.2
```

```
summary(burg_syuzhet)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    -2.0     1.2     2.1     2.2     3.1     7.8
```

```
# 7.1.2 - Bing
burg_bing <- get_sentiment(df_burg$Review, method="bing")
head(burg_bing)
```

```
## [1] 3 0 3 6 0 4
```

```
summary(burg_bing)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    -4.0     1.0     2.0     2.5     4.0    10.0
```

```
# 7.1.3 - Affin
burg_afinn <- get_sentiment(df_burg$Review, method="afinn")
head(burg_afinn)
```

```
## [1] 10  0  9  8 -1 10
```

```
summary(burg_afinn)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    -8.0     3.0     6.0     6.5     9.0    25.0
```

2. NRC Sentiment analysis was used to evaluate the emotions and classify them based on the internal options.

```
# 7.2 - NRC Sentiment Analysis
# anger, anticipation, disgust, fear, joy, sadness, surprise, trust
burg_nrc <- get_nrc_sentiment(df_burg$Review)
head(burg_nrc)
```

```
##   anger anticipation disgust fear joy sadness surprise trust negative positive
## 1     0            2       0    0   2       0        1     2        0        2
## 2     0            1       0    0   1       1        0     1        1        1
## 3     0            0       0    0   1       0        0     0        0        1
## 4     1            2       0    0   2       0        0     3        0        3
## 5     1            1       1    0   1       1        0     1        1        2
## 6     0            0       0    1   3       0        0     2        0        3
```

```
dim(burg_nrc)
```

```
## [1] 1454   10
```

3. Quick transformations – transpose, aggregation, clean-up are performed

```
# 7.3.1 - Transpose
burg_nrc_t <- data.frame(t(burg_nrc))

# 7.3.2 - Grouping using Sum
burg_nrc_g <- data.frame(rowSums(burg_nrc_t[1:1454]))

# 7.3.3 - Cleanup
names(burg_nrc_g)[1] <- "count"
burg_nrc_g <- cbind("sentiment" = rownames(burg_nrc_g), burg_nrc_g)
rownames(burg_nrc_g) <- NULL
burg_nrc_final<-burg_nrc_g[1:8,]
```

4. Plotting the emotional classification

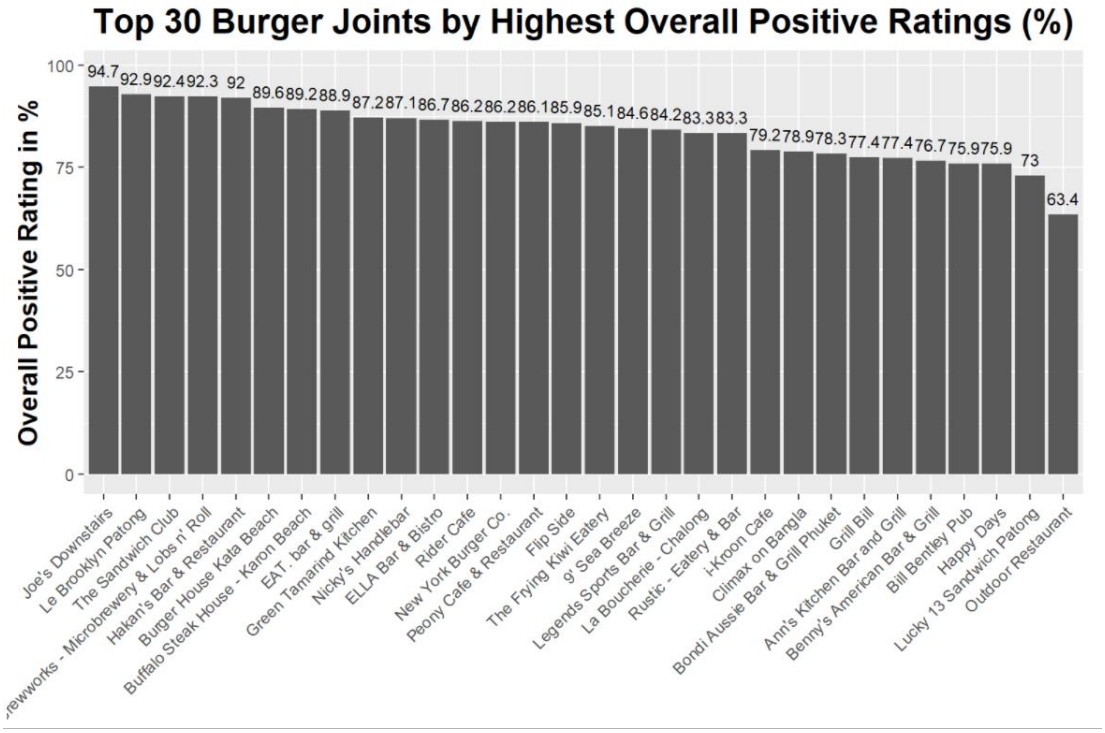
```
# 7.3.4 - Plot using quickplot
#Plot One - count of words associated with each sentiment
quickplot(sentiment, data=burg_nrc_final, weight=count, geom="bar", fill=sentiment, ylab="count")+
  ggtitle("Survey sentiments")+
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1)) +
  theme(axis.title.x = element_text(size=14, face = 'bold'))+
  theme(axis.title.y = element_text(size=14, face = 'bold'))+
  theme(plot.title = element_text(hjust = 0.5, size = 20, face = 'bold'))
```

5. A second plot to visualise strong and weak emotions (based on sorting).

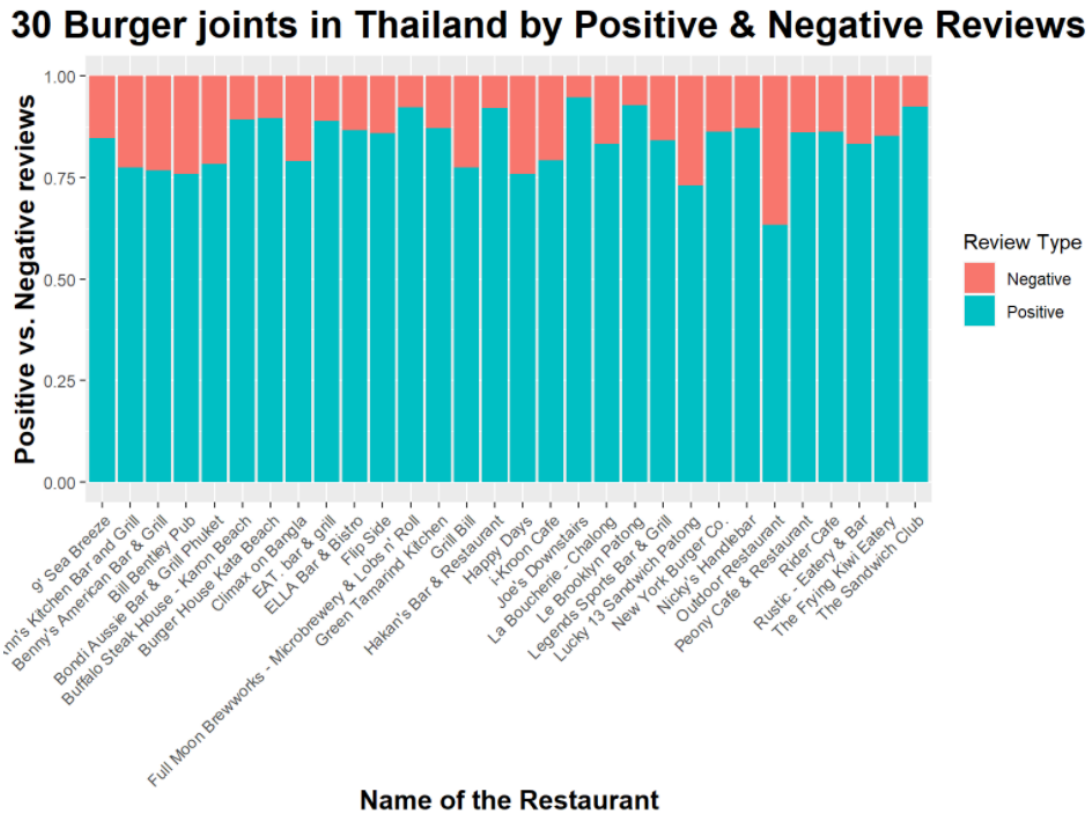
```
# 7.3.4 - Plot to identify strong vs weak emotions
barplot(
  sort(colSums(prop.table(burg_nrc[, 1:8]))),
  horiz = TRUE,
  cex.names = 0.7,
  col='steelblue',
  las = 1,
  main = "Emotions in Reviews", xlab="Percentage"
)
```

4.6.1.3 Results visualisation in R

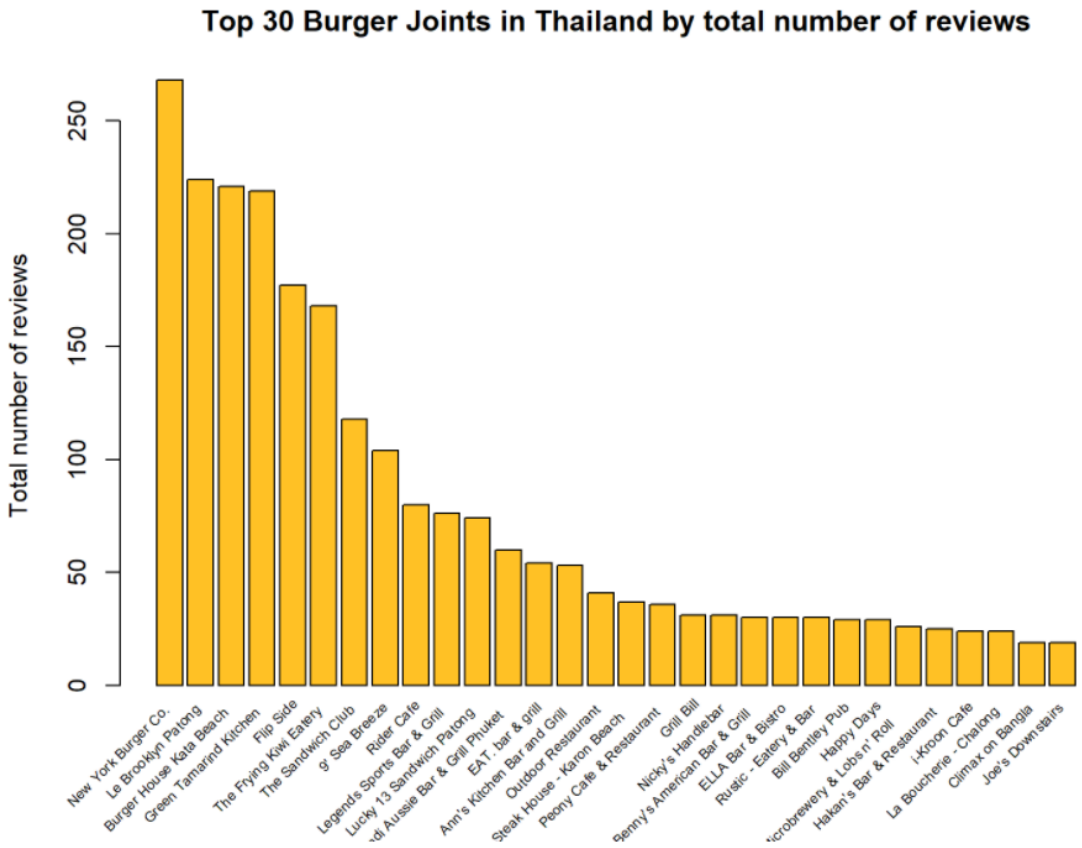
1. Highest Overall ratings



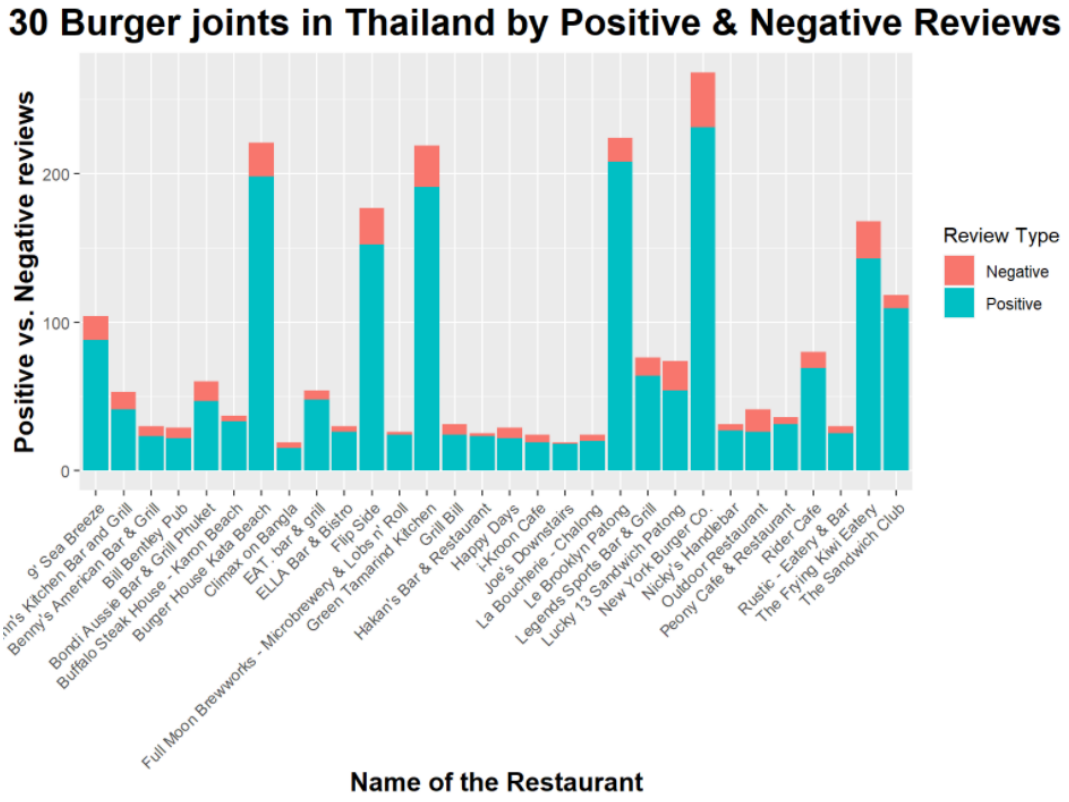
2. Comparative analysis for positive and negative ratings



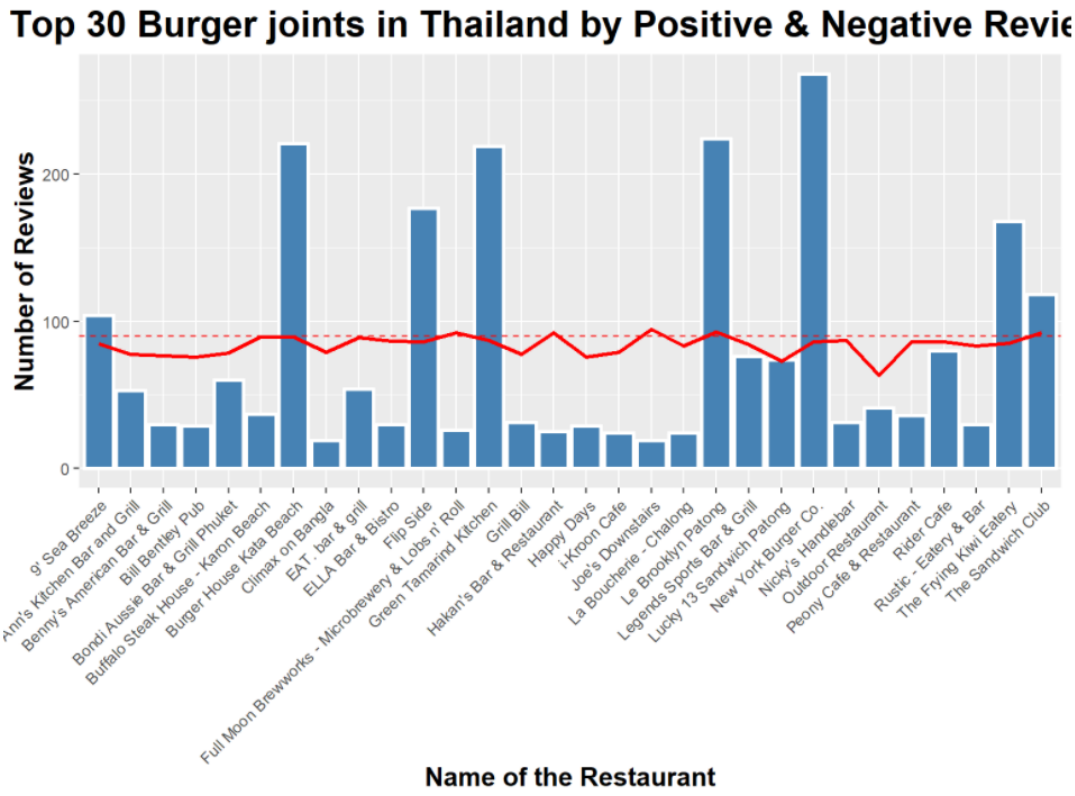
3. Top 30 burger joints by total number of reviews



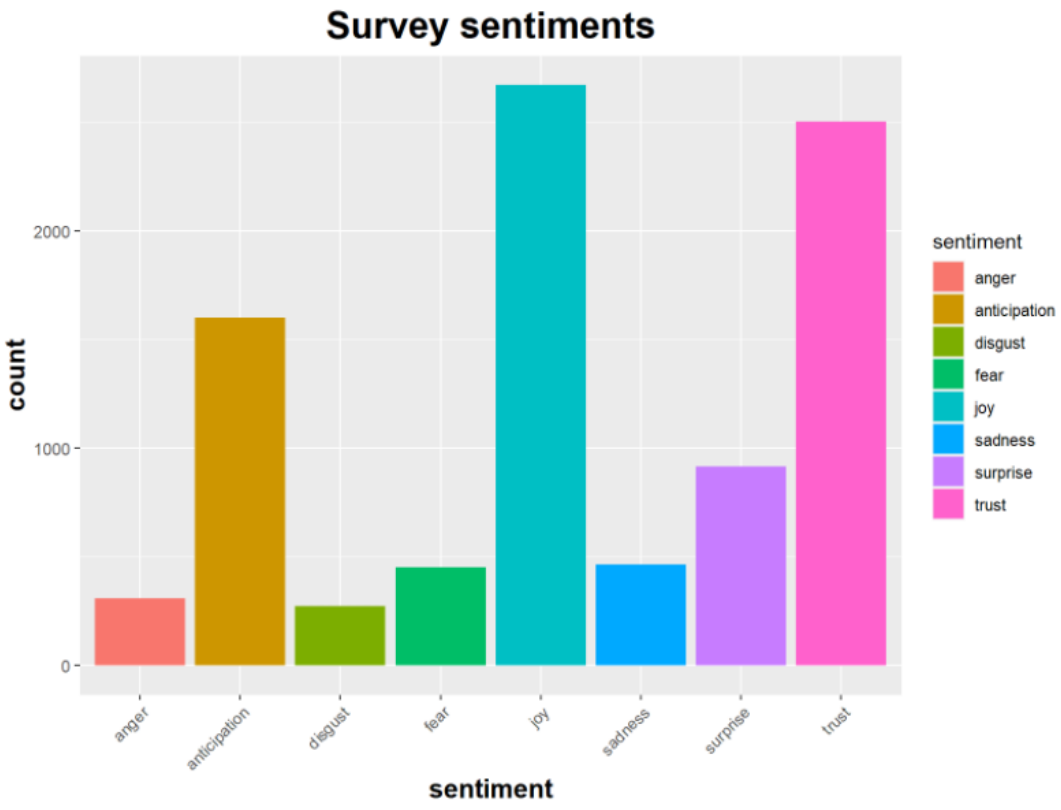
4. Comparing total reviews



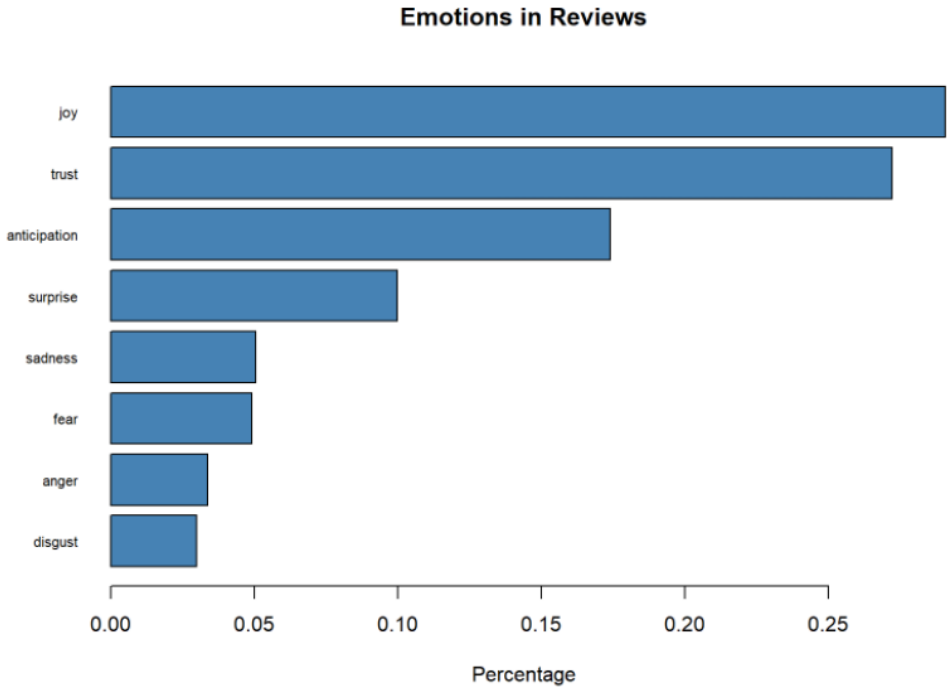
5. Comparison using combo chart –



6. Survey Sentiments – NRC Sentiment analysis



7. Emotions in all 1454 Reviews for burgers –



4.6.2 Data Exploration and Attribute Visualization in SAS EM

4.6.2.1 Model Building in SAS EM

1. Import the cleaned dataset from the path using the below settings.

.. Property	Value
General	
Node ID	FIMPORT
Imported Data	...
Exported Data	...
Notes	...
Train	
Variables	...
Import File	D:\University\ASDM\Develo ...
Maximum Rows to Import	1000000
Maximum Columns to Import	10000
Delimiter	,
Name Row	Yes
Number of Rows to Skip	0
Guessing Rows	500
File Location	Local
File Type	xlsx
Advanced Advisor	No
Rerun	No
Score	
Role	Train
Report	
Summarize	No
Status	
Create Time	30/12/21 19:09
Run ID	32ae6706-ed20-4091-bd10-9f
Last Error	
Last Status	Complete
Last Run Time	31/12/21 00:02
Run Duration	0 Hr. 0 Min. 2.91 Sec.
Grid Host	
User-Added Node	No

2. Edit the variables to identify ID, Text and Target variables.

Name	Role	Level	Report
A	**Rejected**	**Nominal**	**No**
Hotel_Restaura	**Target**	**Nominal**	**No**
ID	**ID**	**Nominal**	**No**
Location	**Rejected**	**Nominal**	**No**
Review	**Text**	**Nominal**	**No**
Review_Date	**Rejected**	**Nominal**	**No**

3. Perform test parsing

.. Property	Value
General	
Node ID	TextParsing
Imported Data	
Exported Data	
Notes	
Train	
Variables	
⊟ Parse	
├ Parse Variable	Review
└ Language	English
⊟ Detect	
├ Different Parts of Speech	Yes
├ Noun Groups	Yes
├ Multi-word Terms	SASHELP.ENG_MULTI
├ Find Entities	None
└ Custom Entities	
⊟ Ignore	
├ Ignore Parts of Speech	'Aux' 'Conj' 'Det' 'Interj' 'Pa
├ Ignore Types of Entities	
└ Ignore Types of Attributes	'Num' 'Punct'
⊟ Synonyms	
├ Stem Terms	Yes
└ Synonyms	SASHELP.ENGSYNMS
⊟ Filter	
├ Start List	
├ Stop List	SASHELP.ENGSTOP
└ Select Languages	
Report	
Number of Terms to Display	20000
Status	
Create Time	30/12/21 19:26
Run ID	dfd80464-8767-4c31-9184-56
Last Error	
Last Status	Complete
Last Run Time	31/12/21 00:02
Run Duration	0 Hr. 0 Min. 8.14 Sec.
Grid Host	
User-Added Node	No

4. Apply the below settings for Text Filter

.. Property	Value
General	
Node ID	TextFilter
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Spelling	
Check Spelling	No
Dictionary	SOURCE.ENGDICT
Weightings	
Frequency Weighting	Default
Term Weight	Default
Term Filters	
Minimum Number of Documen	4
Maximum Number of Terms	.
Import Synonyms	
Document Filters	
Search Expression	
Subset Documents	
Results	
Filter Viewer	
Spell-Checking Results	
Exported Synonyms	
Report	
Terms to View	All
Number of Terms to Display	20000
Status	
Create Time	30/12/21 19:28
Run ID	25aa350a-349b-4d24-ad3b-8a
Last Error	
Last Status	Complete
Last Run Time	31/12/21 00:02
Run Duration	0 Hr. 0 Min. 5.18 Sec.
Grid Host	
User-Added Node	No

5. Text Rule builder settings

.. Property	Value
General	
Node ID	TextRule
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Generalization Error	Medium
Purity of Rules	Medium
Exhaustiveness	Medium
Score	
Content Categorization Code	
Change Target Values	
Status	
Create Time	30/12/21 20:42
Run ID	
Last Error	Run time error was encoun
Last Status	Failed
Last Run Time	31/12/21 00:02
Run Duration	0 Hr. 0 Min. 6.89 Sec.
Grid Host	
User-Added Node	No

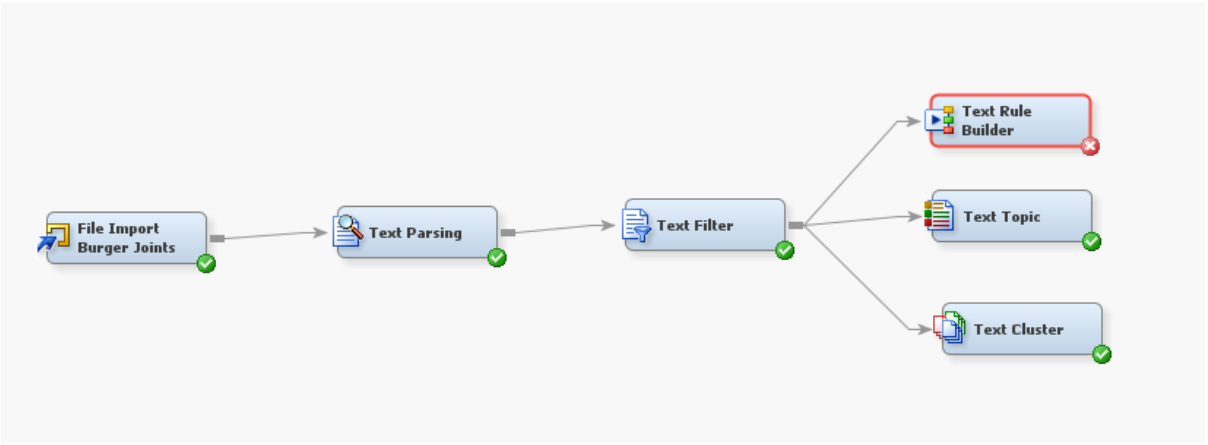
6. Text Topic settings

Property	Value
General	
Node ID	TextTopic
Imported Data	
Exported Data	
Notes	
Train	
Variables	
User Topics	
Term Topics	
Number of Single-term Topics	0
Learned Topics	
Number of Multi-term Topics	30
Correlated Topics	No
Results	
Topic Viewer	
Status	
Create Time	30/12/21 20:47
Run ID	a9b64bcd-1c3d-4d4b-823c-f4f
Last Error	
Last Status	Complete
Last Run Time	31/12/21 00:03
Run Duration	0 Hr. 0 Min. 5.12 Sec.
Grid Host	
User-Added Node	No

7. Text cluster settings

Property	Value
General	
Node ID	TextCluster
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Transform	
SVD Resolution	Low
Max SVD Dimensions	100
Cluster	
Exact or Maximum Number	Maximum
Number of Clusters	30
Cluster Algorithm	Expectation-Maximization
Descriptive Terms	25
Status	
Create Time	30/12/21 19:46
Run ID	47ea32f1-35b1-420d-927d-4e
Last Error	
Last Status	Complete
Last Run Time	31/12/21 00:06
Run Duration	0 Hr. 0 Min. 5.57 Sec.
Grid Host	
User-Added Node	No

8. Process flow diagram looks like this

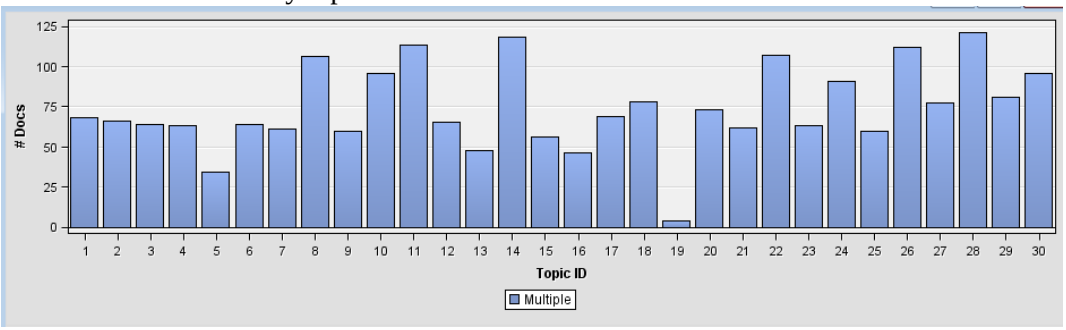


4.6.2.2 Model Assessment in SAS EM

1. Topics –

Topic ID	Document Cutoff	Term Cutoff	Topic	Number of Terms	# Docs
1	0.112	0.049	+nice,+bite,+night,kata,+quick	37	68
2	0.129	0.048	friendly,+price,+fair price,fair,definitely	29	66
3	0.097	0.050	+find,brooklyn,cheese,+order,+work	46	64
4	0.109	0.047	fish,+chip,fried,good,+love	35	63
5	0.109	0.048	009f,f0,008d,008f,00bb	36	34
6	0.108	0.048	always,+year,+veggie,+sweet,favourite	42	64
7	0.117	0.047	+serve,french,french fries,+extra,chef	24	61
8	0.104	0.050	+good,phuket,thailand,+best burger,+find	48	106
9	0.112	0.048	+day,+breakfast,+good,+eat,+time	34	60
10	0.115	0.048	+restaurant,+pizza,+beach,+serve,+ha...	47	96
11	0.131	0.048	+food,thai,thai food,+menu,western	31	113
12	0.103	0.049	mac,cheese,+bite,delicious,+owner	44	65
13	0.111	0.047	+steak,+rib,+time,+salad,+love	36	48
14	0.105	0.049	+eat,+review,+love,great,+meat	42	118
15	0.105	0.049	perfect,+choice,great service,service,g...	37	56
16	0.111	0.049	brooklyn,le,especially,friendly,+month	45	46
17	0.107	0.049	+beer,+cold,+rib,great,bar	47	69
18	0.115	0.049	+sandwich,beef,+beef burger,+club,fal...	53	78
19	0.089	0.043	064a,062a,060c,062d,062c	14	4
20	0.099	0.050	+lamb burger,lamb,back,delicious,first	65	73
21	0.100	0.049	+want,fries,oil,+cook,+dry	46	62
22	0.108	0.050	+rmeat,+taste,beef,great,+order	49	107
23	0.099	0.049	lunch,+love,chicken,+find,+dinner	44	63
24	0.099	0.050	+price,great,+serve,+love,+food	51	91
25	0.107	0.047	+review,french fries,+taste,+pay,combi...	32	60
26	0.107	0.050	+visit,phuket,+love,good,+time	59	112
27	0.117	0.049	great,atmosphere,+great burger,+nice,...	50	77
28	0.117	0.050	good,+good,service,+coffee,+nice	73	121
29	0.103	0.049	+night,+disappoint,great,+dinner,back	45	81
30	0.109	0.050	chicken,bacon,+curry,chicken,+chicke...	57	96

2. Number of documents by topics

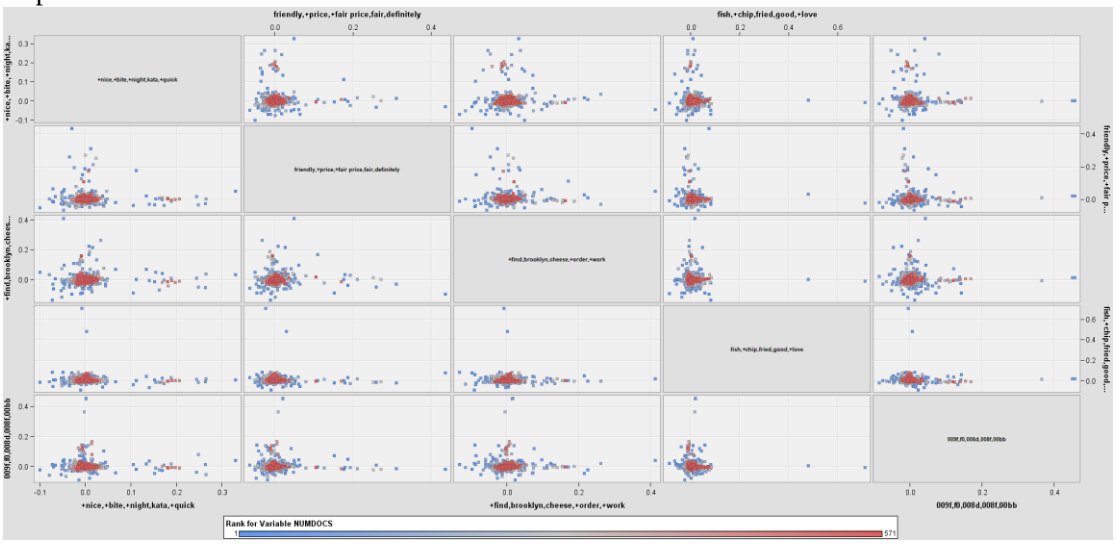


3. Terms –

Term	Role	Attribute	WEIGHT	Freq	# Docs	Keep	Rank for Variable NUMDOCS	+nice,+bite,+night,kata,+quick	friendly,+price,+fair,price,fair,definitely	+find,brooklyn,cheese,+order,+work	fish,+chip,fried,good,+love	009f,f0,008d,008f,00bb	always,+year,+veggie,+sweet,favourite	+serve,french,french fries,+extra,chef	+good,phuket,thailand,+best burger,+find	+day,+breakfast,+good,+eat,+time	+restaurant,+pizza,+beach,+serve,+hamburger	+food,thai,thai food,+menu,western	mac,cheese,+bite,delicious,+owner	+steak,+rib,+time,+salad,+love	+eat,+review,+love,great,+meat	perfect,+choice,great service,service,great	brooklyn,le,especially,friendly,+month	+beer,+cold,+rib,great,bar
+ burger	Noun	Alpha	0.040647	1108	795Y		1	0.004	0.02	0.016	0.014	0.023	0.013	-0.001	0.045	0.021	0.001	0.016	0.022	0.017	0.006	0.013	0.013	0.0
+ good	Adj	Alpha	0.313356	403	330Y		2	-0.007	-0.046	-0.089	0.059	0.143	0.049	-0.016	0.406	0.126	0.073	-0.032	0.071	0.074	0	-0.061	0.124	0.0
+ food	Noun	Alpha	0.427353	311	267Y		3	-0.07	-0.017	-0.06	0.007	-0.002	0.039	-0.016	-0.106	0.083	0.047	0.617	0.019	-0.009	-0.049	-0.009	0.15	-0.0
+ place	Noun	Alpha	0.173302	237	213Y		4	0.015	0.053	0.011	0.004	0.006	0.007	-0.014	0.032	-0.001	-0.005	0.061	0.002	0.007	0.018	0.066	-0.015	-0.0
great	Adj	Alpha	0.389077	191	162Y		5	-0.066	0.017	0.009	0.006	0.011	0.025	0.117	-0.092	0.023	0.066	0.076	0.077	0.024	0.002	0.228	-0.071	0.1
service	Noun	Alpha	0.418135	150	150Y		6	-0.02	-0.042	-0.044	0.011	0.031	-0.008	0.075	0.006	-0.037	0.09	-0.017	0.065	0.026	-0.039	0.235	-0.047	-0.0
+ staff	Noun	Alpha	0.339389	150	148Y		7	-0.001	0.187	-0.038	0.031	0.05	0.041	0	0.008	-0.013	-0.012	-0.016	-0.011	-0.018	0.005	0.071	0.115	-0.0
friendly	Adj	Alpha	0.51548	151	148Y		7	-0.03	0.436	-0.097	0.075	-0.018	-0.049	0.134	-0.02	-0.044	-0.043	-0.033	0.016	-0.013	0.013	0.014	0.199	-0.0
+ nice	Adj	Alpha	0.512879	149	134Y		9	0.329	0.05	0.033	0.009	0.04	-0.04	-0.042	-0.018	-0.112	0.112	0.087	-0.001	-0.046	0.085	-0.102	-0.034	-0.1
+ eat	Verb	Alpha	0.610195	130	119Y		10	0.067	-0.019	0.072	-0.011	-0.043	-0.019	0.149	0.027	0.123	-0.083	-0.008	-0.142	0.04	0.675	-0.03	-0.018	-0.0
fries	Noun	Alpha	0.513285	131	119Y		10	-0.035	-0.043	0	-0.052	0.012	0.031	0.111	0.059	-0.002	-0.043	-0.037	0.04	-0.018	-0.134	0.196	0.181	-0.0
+ recom	Verb	Alpha	0.360946	120	117Y		12	-0.008	0.152	-0.086	-0.023	0.071	-0.055	-0.018	0.066	0.037	-0.022	0.026	0.076	0.038	-0.033	-0.013	-0.025	-0.0
+ time	Noun	Alpha	0.51941	111	96Y		13	-0.01	-0.01	-0.091	-0.088	0.022	0.173	-0.023	-0.028	0.106	-0.026	0.004	-0.049	0.183	0.052	0.024	0.041	-0.0
+ order	Verb	Alpha	0.500105	104	92Y		14	0.016	0.027	0.216	0.048	-0.046	-0.088	-0.011	-0.082	0.009	0.103	-0.045	0.031	0.009	-0.083	-0.056	0.012	-0.0
cheese	Noun	Alpha	0.474888	95	90Y		15	0.008	0.008	0.225	-0.006	-0.023	0.002	-0.018	0.001	0.022	-0.012	-0.028	0.363	0.013	-0.004	-0.025	-0.033	-0.0
+ restaur	Noun	Alpha	0.649202	98	89Y		16	-0.052	0.041	0.005	-0.049	0.008	0.101	0.001	0.062	0.052	0.682	-0.065	-0.123	0.046	0.002	-0.007	0.008	-0.0
+ tasty	Adj	Alpha	0.514503	92	89Y		16	0.112	0.177	0.005	0.001	-0.019	-0.049	-0.044	0.015	-0.039	0.01	0.031	0.014	-0.039	0.031	0.179	0.139	0.0
good	Noun	Alpha	0.524374	91	87Y		18	-0.049	-0.032	-0.02	0.063	-0.011	0.036	-0.023	0.094	-0.027	-0.026	-0.002	-0.024	0.112	-0.044	-0.004	-0.003	-0.0
+ price	Noun	Alpha	0.664552	86	84Y		19	0.012	0.311	-0.027	-0.024	-0.015	0.028	-0.018	0.049	0.032	0.004	-0.055	0.019	0.057	0.033	-0.065	0.019	0.0
phuket	Prop	Alpha	0.566266	86	79Y		20	0.02	-0.023	0.007	-0.001	-0.018	0.046	0.037	0.303	0.038	0.045	0.012	-0.018	-0.035	0.048	0.078	-0.038	-0.0
+ owner	Noun	Alpha	0.524929	71	69Y		21	0.011	0.037	-0.04	0.008	-0.003	-0.034	0.017	0.144	-0.001	-0.007	0.018	0.193	-0.036	0.04	-0.057	0.007	-0.0
+ beer	Noun	Alpha	0.813909	72	67Y		22	0.134	0.021	0.072	0.044	0.019	0.024	0.005	-0.025	0.047	-0.05	-0.03	-0.044	-0.009	-0.058	-0.044	0.013	0.7
+ best bu	Noun Gro	Alpha	0.415969	68	67Y		22	-0.003	-0.002	-0.024	0.009	0.14	0.054	-0.004	0.184	0.035	0.003	-0.024	-0.018	0.007	0.016	-0.023	0.005	-0.0
+ visit	Verb	Alpha	0.682138	71	66Y		24	0.038	0.02	0.004	-0.061	0.016	0.174	-0.016	-0.054	0.054	0.043	-0.032	-0.012	-0.014	-0.1	-0.043	-0.064	-0.0
little	Adj	Alpha	0.563829	67	65Y		25	0.008	0.18	0.035	-0.024	-0.008	-0.025	-0.038	0.001	0.04	0.066	0.041	-0.007	-0.003	-0.052	-0.011	0.007	0.0
thai	Adj	Alpha	0.622951	68	64Y		26	0.006	-0.022	-0.006	-0.012	0.017	-0.039	0.018	0.029	-0.005	-0.03	0.431	0.017	0.013	0.009	-0.021	-0.01	0.0
best	Noun	Alpha	0.50965	62	60Y		27	-0.004	0.016	-0.005	-0.019	-0.002	0.013	0.018	0.101	-0.002	-0.018	0.033	0.002	0.011	0.062	0.021	0.024	-0.0
+ meal	Noun	Alpha	0.605891	63	59Y		28	-0.014	0	-0.009	0.028	-0.024	-0.066	-0.014	0.089	0.038	0.024	-0.029	0.066	-0.002	-0.063	-0.044	0	0.0
+ day	Noun	Alpha	0.842717	63	57Y		29	0.044	-0.018	-0.028	-0.025	0.028	-0.042	0	-0.019	0.804	0.001	-0.018	0.051	-0.041	0.018	0.017	0.022	-0.0
+ want	Verb	Alpha	0.708018	62	57Y		29	0.169	-0.003	0.043	0.024	-0.026	-0.031	-0.007	0.025	0.059	-0.002	0.095	-0.056	0.043	0.037	-0.101	-0.049	0.0
+ love	Verb	Alpha	0.856438	57	54Y		31	-0.072	-0.011	-0.133	0.082	0.012	0.078	-0.067	-0.318	0.06	0.06	-0.068	0.186	0.121	0.163	-0.013	0.171	-0.0
beef	Noun	Alpha	0.731499	60	54Y		31	-0.005	0.001	-0.01	-0.005	0.04	-0.058	0.03	0.063	0.005	-0.065	-0.006	-0.047	-0.015	-0.106	-0.006	0.004	0
+ enjoy	Verb	Alpha	0.642707	55	53Y		33	-0.035	0.008	-0.045	0.033	0.013	-0.027	0.007	-0.01	0.001	0.032	0.026	0.025	0.066	0.048	0.022	0.01	0.0
delicious	Noun	Alpha	0.739617	54	53Y		33	-0.071	0.064	-0.027	-0.059	0.009	-0.008	-0.022	0.034	0.072	0.09	0.003	0.154	0.022	0.096	0.073	-0.015	0
great	Noun	Alpha	0.74789	52	52Y		35	-0.1	-0.051	0.049	-0.017	-0.022	-0.009	-0.058	-0.006	0.036	-0.059	-0.029	-0.035	-0.043	0.183	0.045	-0.094	0.1
+ chip	Noun	Alpha	0.770334	55	51Y		36	0.003	0.03	0.002	0.481	0.006	-0.018	0.009	0.022	0.019	0.007	-0.004	-0.052	0.01	-0.039	-0.027	-0.025	-0.0
+ fresh	Adj	Alpha	0.669173	53	51Y		36	0.152	-0.033	-0.014	0.01	-0.047	0.002	-0.003	0.017	0.026	-0.01	0.028	0.043	0.033	0.056	0.03	0.083	-0.0
thai food	Noun Gro	Alpha	0.493081	54	51Y		36	0.016	0.025	0.001	-0.018	0.009	-0.033	0.01	-0.025	0.012	-0.003	0.296	-0.002	-0.005	0.004	-0.01	-0.007	-0.0
+ amaze	Verb	Alpha	0.588832	52	50Y		39	-0.006	0.023	-0.06	0.039	0.043	-0.021	0.002	0	-0.011	0.016	-0.005	0.1	-0.019	0.007	0.044	-0.023	-0.0
excellent	Adj	Alpha	0.588832	54	50Y		39	-0.045	-0.023	-0.032	-0.017	0.016	-0.021	0.002	0.029	-0.053	0.058	0.067	0.048	0.014	-0.032	0.014	-0.014	0.0
+ meat	Noun	Alpha	0.852878	50	49Y		41	-0.004	0.012	-0.04	-0.018	0.061	0.165	-0.021	-0.125	-0.087	0.015	0.024	0.041	0.042	0.161	0.03	-0.037	0.0
delicious	Adj	Alpha	0.695502	50	48Y		42	-0.053	-0.028	-0.073	0.023	-0.052	-0.026	-0.027	0.02	-0.008	0.02	-0.003	0.245	0.012	-0.037	-0.002	-0.001	-0.0
lunch	Noun	Alpha	0.90759	52	48Y		42	-0.043	0.009	0.001	-0.023	-0.026	0.001	0.178	-0.016	0.057	-0.05	-0.044	-0.049	0.04	-0.07	-0.014	-0.052	-0.0
thailand	Prop	Alpha	0.657713	48	48Y		42	-0.033	0.019	0.04	0	0.04	0.057	-0.003	0.215	-0.017	-0.005	0.047	-0.068	-0.043	0.042	0.02	-0.054	-0.0
+ cook	Verb	Alpha	0.615704	49	47Y		45	-0.049	-0.015	-0.011	0.021	0.036	-0.047	-0.029	0.073	-0.022	0.005	-0.016	0.03	0.033	0.002	0.05	-0.03	0.0
definitely	Adv	Alpha	0.615704	50	47Y		45	-0.034	0.26	-0.047	-0.016	0.009	0.017	-0.003	0.125	-0.003	-0.048	0.032	0.019	0.01	-0.021	0.024	-0.025	0.0
thai	Noun	Alpha	0.615704	49	47Y		45	0.02	0.028	0.033	0.035	0.025	0.011	-0.005	-0.073	-0.005	0.032	0.124	-0.039	0	0.029	-0.02	-0.038	-0.0
+ find	Verb	Alpha	0.81068	47	45Y		48	-0.047	0.049	0.412	0.017	0.042	0.059	-0.035	0.183	0.003	0.012	0.115	0.011	-0.004	0.04	0.056	-0.065	-0

4.6.2.3 Results visualisation in SAS EM

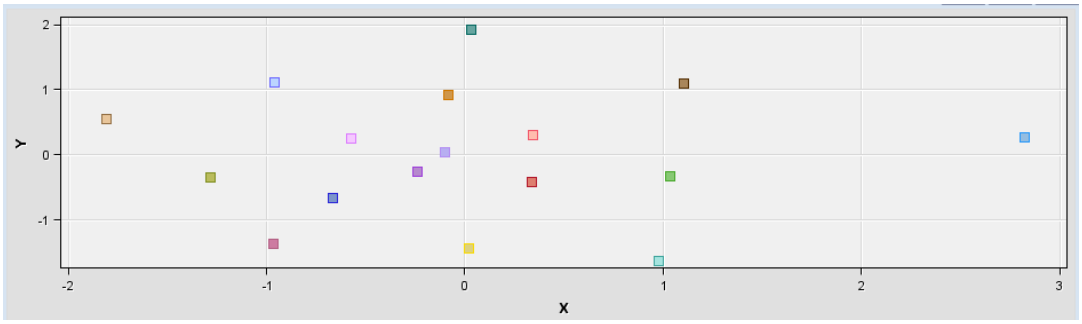
1. Topic terms – and sentiment can be seen in the below matrix for the most used terms



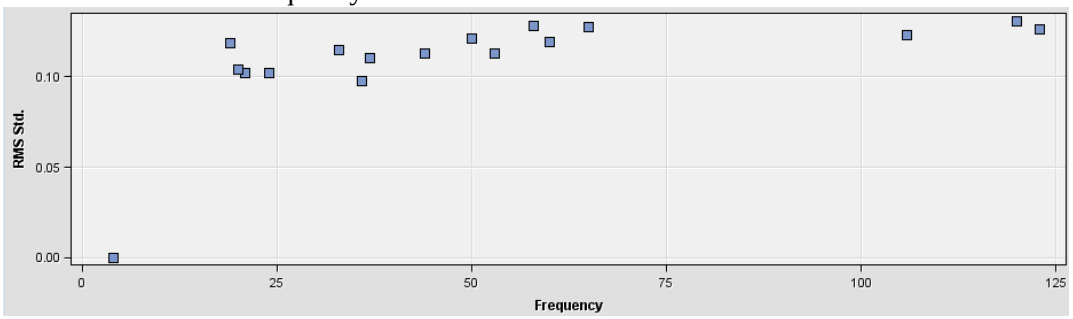
2. The following clusters were created

Cluster ID	Descriptive Terms	Frequency	Percentage	Coordinate 1	Coordinate 2	Coordinate 3	Coordinate 4	Coordinate 5	Coordinate 6	Coordinate 7
1	+serve piri +'chicken burger' chicken +dish +'beef burger' +restaurant apple +chip +portion +side reasonable perfectly french +crispy english 'quick service' +batter ...	50	6%	0.361716	0.048554	-0.01055	-0.05071	-0.0189	-0.01596	-0.02
2	+steak +'dry burger' +'positive review' 'french fries' fries french crunchy good +mention +pay combination +coffee +base +sauce +taste +kid average bacon +fry +s...	65	7%	0.295978	0.038031	-0.00747	0.004837	0.159692	0.022806	0.0255
3	009f f0 008d 008f 00bb 'amazing staff' 'perfect ambiance' franch 'american burger' ambiance +guy +speak english american patong 008b +well amazing perfect +'...	21	2%	0.24868	-0.07449	0.006915	-0.07313	-0.10918	0.524497	0.2003
4	+'choice topping' flavoursome variation perfect +'burger place' 'attentive staff' +choice +topping 'great service' lovely +'good choice' attentive different +juice service si...	20	2%	0.297147	-0.04205	0.031805	0.103818	0.022041	0.021184	-0.044
5	+food beef 'thai food' +sandwich thai +club +'western food' +thing break +menu western +'beef burger' superb 'patong beach' +drink selection double excellent delici...	123	14%	0.372256	-0.09082	-0.0055	-0.10788	-0.03013	-0.04117	0.0647
6	+restaurant +pizza +pasta 'excellent food' +beach excellent +family +sit 'great service' clean +cocktail +dish +look service phuket +thing huge juicy +hamburger fir...	44	5%	0.387061	-0.08518	0.002382	-0.07853	.0007704	-0.10364	0.0270
7	060c 062a 062c 062d 062e 062f 063a 064a bty oily town +'best burger' fries +good +burger	4	0%	0.033634	-0.00975	0.997799	0.033219	0.008274	-0.00236	-0.003
8	lunch 'great lunch' 'quick lunch' +'extra praline' +praline crumble apple +drop +extra chef 'french fries' free +coffee +quick +stop +nacho oily french service kata +s...	37	4%	0.365128	0.154576	-3.73E-5	0.031583	-0.06014	-0.03218	-0.029
9	+day +eat +'lamb burger' lamb cafe +breakfast music second hot hotel +time kata +cafe phuket twice +meal husband first right +friend +big +'good choice' +gree...	60	7%	0.348995	-0.05282	0.005743	-0.06005	-0.04905	-0.06783	0.0847
10	'main road' +road +beach +locate main 'quick service' bar +view sea 'patong beach' +aussie +miss +busy +pizza cool rawai +'good burger' +'great burger' superb ...	19	2%	0.293413	-0.06028	0.011481	-0.03637	-0.05633	-0.00138	-0.04
11	always le especially +meat +read +year brooklyn +review +curry green +night +food +love +time +eat thai +fresh +dinner bacon +live western +week +crave +gu...	120	14%	0.343578	-0.0771	-0.00942	-0.03378	0.058519	-0.08482	0.0767
12	+beer +cold craft +'cold beer' +wing cold +watch +aussie selection +salad 'great place' free bar sea good +nacho +quick +'cheese burger' great +bite +'good choi...	53	6%	0.35157	-0.09361	-0.01177	0.19895	-0.07811	-0.03651	-0.230
13	+find 'onion capsicum cheese' capsicum reading advisor trip +review 'trip advisor' hard gem +'awesome burger' better 'veggie burger' +work +flavour +look +onion b...	33	4%	0.270711	-0.0602	0.001488	-0.06454	0.160469	-0.02889	0.0544
14	completely dead successful +disappoint full oil town +rib feeling +check +dry +expect +cook +keep size +'best burger' 'good value' +world +want +coffee +portion ...	58	7%	0.291632	-0.00852	0.023901	-0.05918	0.066253	-0.03518	0.017
15	fish +chip +batter snapper +kiwi fried +greet +pull +buy range +die +friend +add +live +yummy +keep +family +'different burger' +sauce +'friendly staff' +area +sit ...	36	4%	0.320352	-0.08136	-0.00441	-0.07686	-0.02251	0.136183	-0.492
16	mac +bite cheese +'cheese bite' jalapeño +ring +owner +onion perfectly delicious +veggie today +'good burger' +'cheese burger' better +flavour 'veggie burger' +am...	24	3%	0.286616	-0.04782	-0.00455	0.072547	0.076602	0.099204	0.1139
17	friendly +'fair price' fair +recommend +staff +'great burger' 'nice atmosphere' definitely +owner worth +nice atmosphere +'friendly staff' super clean helpful 'good servi...	106	12%	0.369422	-0.05494	0.006203	-0.01955	-0.01374	0.045658	-0.041

3. Distance between clusters is also seen



4. RMS validation and frequency



4.7 Results analysis and discussion

4.7.1 Result comparison between R and SAS EM

- Using R and SAS EM, the text mining could be achieved, but due to limited time and resources, sentiment analysis was only performed in R.
- New York Burger co. has the highest number of reviews and Joe's downstairs has the lowest number of reviews.
- 4 of the restaurants that have more than 200 reviews in the last 10 years are New York Burger co., Le Brooklyn Patong, Burger House Kate Beach, Green tamarind Kitchen.
- Between quality and quantity, it is always important to identify the quality, but in our case, there needs to be a middle ground where there is moderate number of reviews as well. It seems the Outdoor restaurant is performing poorly of all with only 63.4 % of positive ratings.
- In terms of quality and quantity, the best burger restaurant according to the data is Le Brooklyn Patong, with more than 200 reviews and 92.9% of positive ratings.

4.7.2 Critical findings

- According to the overall sentiment of the top 30 burger joints, we could see relatively a more amount of joy, trust, and anticipation, with less amount of anger and disgust and a moderate amount of surprise.

4.8 Conclusion

In conclusion, we could confidently say that the top 30 burger joints were successfully identified, data was mined and synthesized properly based on a document matrix, evaluated based on the reviews, and identified key findings that highlight some of the top reviewed, highest and lowest positive ratings, best and worst performing burger joints were identified and reported.

5. References

1. EPA FAQ - EPA - Frequently Asked Questions
2. Air Toxicity due to pollutants - https://www.nature.com/articles/jes201715
3. Source data for classification, clustering, and association rules mining - Click here for link
4. Census Tract Chemical Exposure Analysis
5. American Association for Cancer Research - Air Pollution effects
6. The Institute of Cancer Research
7. Cancer Death measurement in a year - WebMd
8. RI Publication - Social network analysis
9. Fernandez, M.P., Chan, Y.B., Yew, J.Y., Billeter, J.C., Dreisewerd, K., Levine, J.D., Kravitz, E.A. (2010). Pheromonal and Behavioral Cues Trigger Male-to-Female Aggression in Drosophila. PLoS Biol. 8(11): e1000541.
10. B. Rajen and M. Gopal, "Neuro-Fuzzy Decision Trees," International Journal of Neural Filters, Vol. 16, No. 1, 2006, pp. 63-68. doi:10.1142/S0129065706000470
11. Kazienko P and Kajdanowicz T (2010), *"Base classifiers in boosting-based classification of sequential structures"*, Neural Network World. Vol. 20(7), pp. 839-851.
12. Jockers, M. (2017) "Syuzhet Package in R." Available from: https://www.rdocumentation.org/packages/syuzhet/versions/1.0.4. [Accessed 7th June 2019].
13. Jockers, M. (2017) "Syuzhet Package." Available from: https://cran.r-project.org/web/packages/syuzhet/vignettes/syuzhet-vignette.html. [Accessed 7th June 2019].
14. Deepu, S., Pethuru Raj, and S.Rajaraajeswari. (2016) "A Framework for Text Analytics using the Bag of Words (BoW) Model for Prediction"
15. Ashima Sethi, Prerna Mahajan, The International Journal of Computer Science & Applications (TIJCSA) ISSN – 2278-1080, Vol. 1 No. 9 November 2012

6. Appendix

All the code can be accessed using the following URL - **Click HERE for link** which is basically an Access key or a Git Deploy key.

It is important to note that this is not a public repository / open-sourced code and the information within this URL is MIT licensed. This work belongs to Ambareesh Jonnavittula, and it must be only used for educational/allowed purposes as per the license, as part of data ethics.

This link was created using SSH and produces a read-only webpage. SSH connections have mostly been used to secure different types of communications between a local machine and a remote host, including Secure remote access to resources.