

Report

Assignment- 4

Business Intelligence Reporting using Cognos

Measurable Filed:

- The fact table contains details about the date and time. It is selected as a fact measurable field because all the other dimensions like temperature, humidity dimension are highly dependent on fact table.
- Example: The temperature dimension and humidity dimension are dependent on fact table to get the exact time and date when there is drop or raise in temperature and humidity.

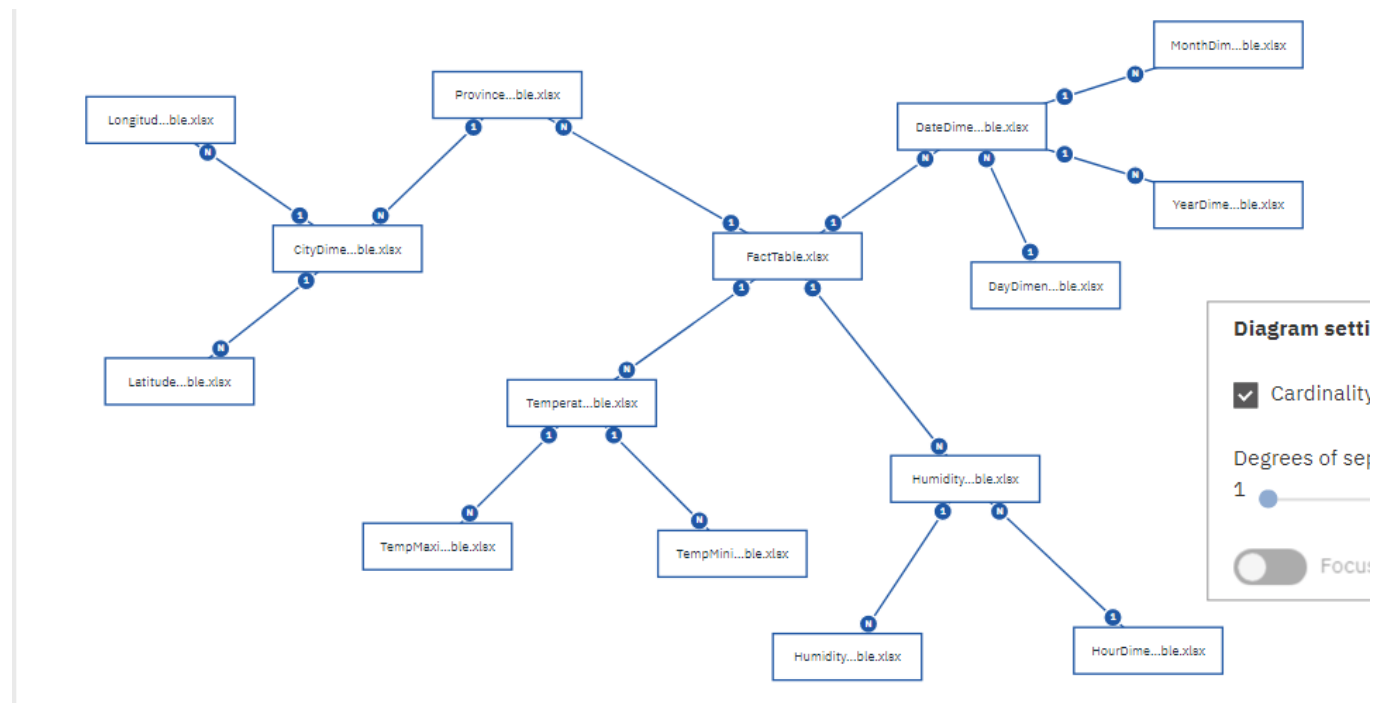


Figure: 1

- Figure 1 (please refer the attachment in the zip folder for the clear image), shows the snowflake schema of the dataset provided.
- The fact table is four different dimension tables namely temperature, humidity, province and date dimension tables.
- As the fact table consists of date and time which provides the details of the humidity and temperature at that particular instance of time.
- The fact table is also related to province dimension table because the position of provinces in this period of time where temperature and humidity varies constantly.

- The province dimension table is further has one-to-many relationship with cities where each area is connected to the city as they are many areas which again consist of longitude and latitude.
- Humidity and Temperature dimension table are further classified into minimum and maximum temperatures.
- The fact table maintains a one-to-many relationship date and day to retrieve the proper format of date and time individually.

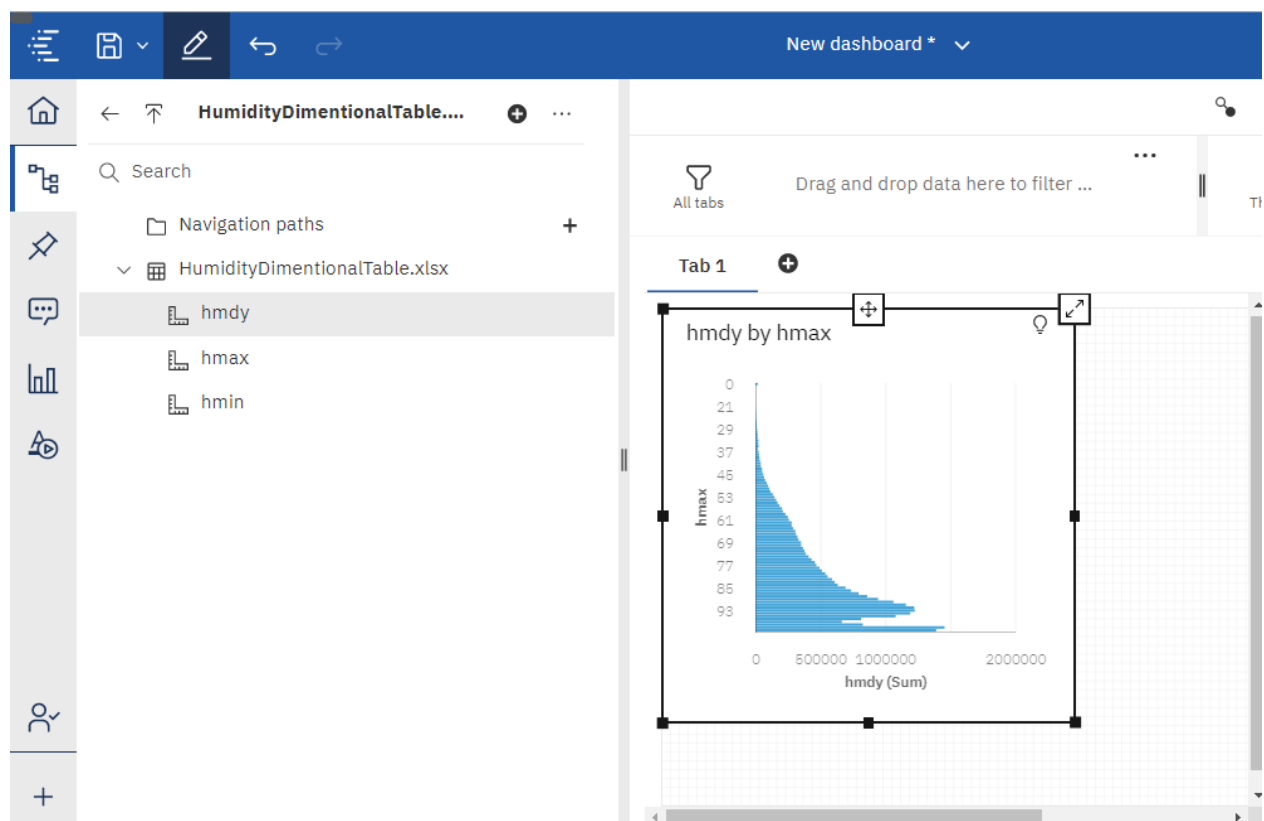


Figure 2

- The display visual analysis is suitable when plotted graphically, As from the Figure 2, the behaviour between humidity and maximum humidity can be analysed conveniently
- Video 1 (please refer the attachment in the zip folder for the clear video), I have explained how can we represent the information in a knowledgeable format like bar graph, pie chart, etc.

Data Cleaning:

- The municipalities names in the Brazil country specified are not clean. Cleaning the data by replacing "São Gonçalo" with "São Gonçalo", "Araxá" with "Araxá", "São Mateus" with "São Mateus", "Vitória" with "Vitória", "Afonso Cláudio" with "Afonso Cláudio", "Nova Venécia" with "Nova Venécia". This is done by Find/Replace tool present in Microsoft Excel.

- The column **wsnm** and **city** in the dataset **sudeste** are identical so the entire column, attribute **wsnm** can be deleted.
- The column **prcp** can be ignored as there are only few thousands of records out of million records filled out of which many records have value "0".
- The columns **yr**, **mo**, **da**, **date** and **hr** are the derived attributes from the column **mdct** so all the derived attributes can be retrieved later when required

Basic Data Analysis and Presentation using R

Git Repository URL: <https://git.cs.dal.ca/sarvi/r-programming.git>

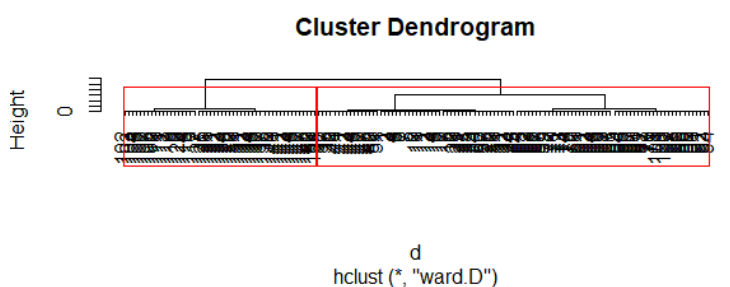
Procedure:

1. Connect to the tweeter Database via MongoClient.
2. After we get the access to the data, check for a field 'created_at' and underlying information.
3. Convert them into a timestamp variable by multiplying some constant value to the created_at field or using a pre-defined library.
4. The timestamp is then inserted successfully into the CSV file in the form of single column.
5. The data is loaded on R.
6. The data is loaded on RStudio by read.csv() method
7. As shown below in Figure 3, the clusters are being created.
8. The cluster head varying according to the number of clusters as shown in Figure 3 for various values of K.

The Cluster Dendrogram changes as we change the values of K.

Example1 :

```
d <- dist(mydata, method = "euclidean")
fit <- hclust(d, method="ward")
plot(fit)
groups <- cutree(fit, k=5)
rect.hclust(fit, k=2, border="red")
```



```

1 data = read.csv('C:/Users/user/PycharmProjects/PresentationR/sample1.csv')
2 print(data)
3
4 timestampData <- na.omit(data)
5 timestampData <- scale(data)
6
7 clustersTime <- (nrow(timestampData)-1)*sum(apply(timestampData,2,var))
8 for (i in 2:8) clustersTime[i] <- sum(kmeans(timestampData,
9                                     centers=i)$withinss)
10 plot(1:8, clustersTime, type="b", xlab="Number of clusters",
11      ylab="within groups sum of squares")
12
13 fit <- kmeans(timestampData, 5)
14 aggregate(mydata,by=list(fit$cluster),FUN=mean)
15 mydata <- data.frame(timestampData, fit$cluster)
16
17 fit <- kmeans(timestampData, 3)
18 aggregate(mydata,by=list(fit$cluster),FUN=mean)
19 mydata <- data.frame(timestampData, fit$cluster)

```

32:40 (Top Level) ↕ R

Console Terminal x Jobs x

```

~/
> mydata <- data.frame(timestampData, fit$cluster)
> fit <- kmeans(timestampData, 5)
> aggregate(mydata,by=list(fit$cluster),FUN=mean)
  Group.1 timestamp fit.cluster
1      1  0.66309897   3.677419
2      2 -0.70882993   5.586207
3      3 -1.38336164   3.600000
4      4  1.37192890   1.387097
5      5 -0.03429822   6.333333
> mydata <- data.frame(timestampData, fit$cluster)
> fit <- kmeans(timestampData, 3)
> aggregate(mydata,by=list(fit$cluster),FUN=mean)
  Group.1 timestamp fit.cluster
1      1  0.01143274   3.560000
2      2 -1.14327409   2.588235
3      3  1.15470683   2.860000

```

Figure 3

Cluster Image plotted using ggplot on the RStudio:

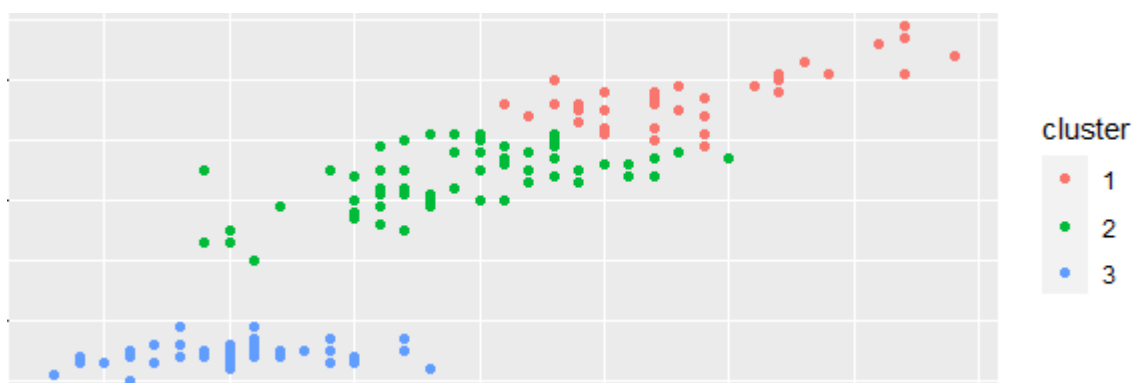


Figure 4

As the initial values for K changes, there is a minor shift in the centroid values. The completely cluster shifts little in some directions by the changes made in the value of K.

Sentiment Analysis

Git Repository URL: <https://git.cs.dal.ca/sarvi/sentiment-analysis.git>

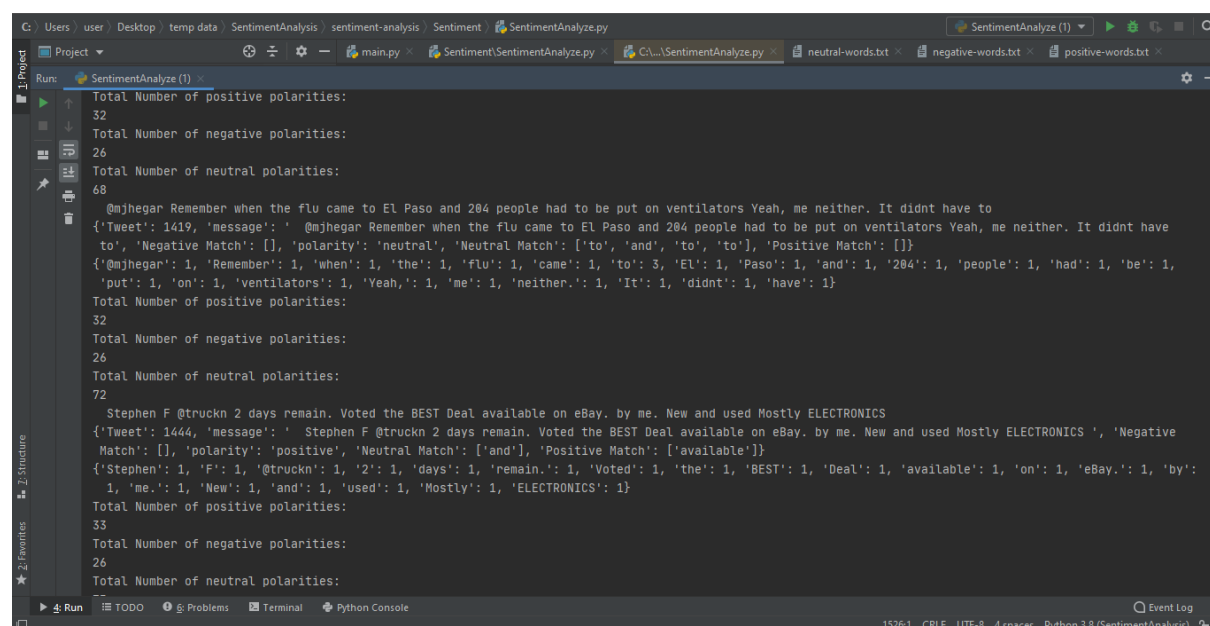
Procedure:

1. Making a list of positive words, negative and neutral words and storing it in a text file.
2. Connect to the MongoDB client with the help of URL. Processed data of the tweets from the database is retrieved which is followed by cleaning the code.
3. Checking the words from the tweet and comparing to the bag of words which can be positive, negative or neutral
4. Polarity is given to each tweet at the end to get an estimate and decide the tweet is a good one or not.
5. The negative match, positive match or neutral match are presented in a form of a list.
6. At the end total number of negative, positive and neutral polarity are displayed.

Output:

```
Via Staten Island news Things are going from bad to worse because not enough people are
{'Tweet': 1403, 'message': 'Via Staten Island news Things are going from bad to worse because not enough people are ', 'Negative Match': ['bad', 'worse'],
 'polarity': 'neutral', 'Neutral Match': ['Things', 'from', 'to', 'because'], 'Positive Match': ['enough']}
{'Via': 1, 'Staten': 1, 'Island': 1, 'news': 1, 'Things': 1, 'are': 2, 'going': 1, 'from': 1, 'bad': 1, 'to': 1, 'worse': 1, 'because': 1, 'not': 1,
 'enough': 1, 'people': 1}
Total Number of positive polarities:
32
Total Number of negative polarities:
26
Total Number of neutral polarities:
68
```

Figure 5 (please refer the attachment in the zip folder for the clear image)



```
Run: SentimentAnalyze (1)
Total Number of positive polarities:
32
Total Number of negative polarities:
26
Total Number of neutral polarities:
68

@mjhegar Remember when the flu came to El Paso and 204 people had to be put on ventilators Yeah, me neither. It didnt have to
{'Tweet': 1419, 'message': '@mjhegar Remember when the flu came to El Paso and 204 people had to be put on ventilators Yeah, me neither. It didnt have to', 'Negative Match': [], 'polarity': 'neutral', 'Neutral Match': ['to', 'and', 'to', 'to'], 'Positive Match': []}
{'@mjhegar': 1, 'Remember': 1, 'when': 1, 'the': 1, 'flu': 1, 'came': 1, 'to': 3, 'El': 1, 'Paso': 1, 'and': 1, '204': 1, 'people': 1, 'had': 1, 'be': 1,
 'put': 1, 'on': 1, 'ventilators': 1, 'Yeah': 1, 'me': 1, 'neither': 1, 'It': 1, 'didnt': 1, 'have': 1}
Total Number of positive polarities:
32
Total Number of negative polarities:
26
Total Number of neutral polarities:
72

Stephen F @truckn 2 days remain. Voted the BEST Deal available on eBay. by me. New and used Mostly ELECTRONICS
{'Tweet': 1444, 'message': 'Stephen F @truckn 2 days remain. Voted the BEST Deal available on eBay. by me. New and used Mostly ELECTRONICS ', 'Negative Match': [], 'polarity': 'positive', 'Neutral Match': ['and'], 'Positive Match': ['available']}
{'Stephen': 1, 'F': 1, '@truckn': 1, '2': 1, 'days': 1, 'remain': 1, 'Voted': 1, 'the': 1, 'BEST': 1, 'Deal': 1, 'available': 1, 'on': 1, 'eBay': 1, 'by': 1,
 'me': 1, 'New': 1, 'and': 1, 'used': 1, 'Mostly': 1, 'ELECTRONICS': 1}
Total Number of positive polarities:
33
Total Number of negative polarities:
26
Total Number of neutral polarities:
72
```

Figure 6 (please refer the attachment in the zip folder for the clear image)

Semantic Analysis

Git Repository URL: <https://git.cs.dal.ca/sarvi/semantic-analysis.git>

Procedure:

1. Connect to the MongoDB client and retrieve the field containing text and title.
2. Check the number of documents "Canada", "rain", "hot" and "cold."
3. Also calculate the inverse document frequency of the same words.

Output:

```
"C:\Users\user\PycharmProjects\Semantic Analysis\venv\Scripts\python.exe" "C:/U

Total news Articles: 2000

Number of documents containing the word 'Canada': 60
Number of documents containing the word 'rain': 89
Number of documents containing the word 'cold': 9
Number of documents containing the word 'hot': 13

Term frequency-inverse document frequency of 'Canada': 1.5228787452803376
Term frequency-inverse document frequency of 'rain': 1.3516399890190685
Term frequency-inverse document frequency of 'cold': 2.3467874862246565
Term frequency-inverse document frequency of 'hot': 2.1870866433571443

Process finished with exit code 0
|
```

Procedure (Highest Relative Frequency):

1. Connect to the MongoDB database and provide article number like 'Article#n' to track the daily progress.
2. After iterating through lot of records, finding out the document having maximum occurrence of the word 'Canada'
3. Each article is given a number for reference, Total word count of the article, frequency of 'Canada' and relative frequency
4. Sort the relative frequency list and word frequency list to get the article number
5. Finally the article with highest occurrence of word "Canada" and has Highest Relative Frequency of word "Canada" with title is given.
6. Highest Relative Frequency is calculated at the last.

Output:

```
Article#1531 has highest occurrence of word "Canada" with frequency 5
Article#319 has Highest Relative Frequency of word "Canada" with title :bBANK OF CANADA DOLLAR STERLING RATES
and Highest Relative Frequency :0.02857142857142857
```

```
Process finished with exit code 0
```

```
Run: RelativeFrequency x
Article#1553
Total words in the article: 323
Frequency of Canada in Article#1553: 2
Relative Frequency: 0.006191950464396285

Article#1635
Total words in the article: 800
Frequency of Canada in Article#1635: 2
Relative Frequency: 0.0025

Article#1672
Total words in the article: 479
Frequency of Canada in Article#1672: 1
Relative Frequency: 0.0020876826722338203

Article#1870
Total words in the article: 220
Frequency of Canada in Article#1870: 1
Relative Frequency: 0.004545454545454545

Article#1888
Total words in the article: 108
Frequency of Canada in Article#1888: 1
```

Run | TODO | Problems | Terminal | Python Console

Rainbow CSV: You can edit Rainbow CSV settings in Settings > Editor > General > Rainbow CSV (3)