**Department of Engineering**
**University of Ottawa**

# CSI4142
# Introduction to Data Science

Muhammad Owais Bawany - 7993647

Matteo Colombi - 8238926

Patrick Langis - 8196917

Felix Singerman - 7970742

# Table of Contents

# Background

We designed, built and implemented a data mart using data on all disasters that have affected Canadians worldwide[5]. We did data staging, designed and implemented OLAP queries, had a user interface to interact and visualize the queries and finally applied several machine learning algorithms to the data.

# Data Staging -- Preprocessing

We decided to do the data staging in python. We have one large python script norm.py that does all the preprocessing. We used the library pandas in order to help with the dataframes.

We used an external data source for population statistics. The data source used was retrieved from Stats Canada 2011 census[3]. We removed the columns that were of no concern to us. We parsed out location column to extract the province and put it in its own column using regex. This data set has information on the population statistics of every city in Canada. This includes whether a city is an Indian Reserve, the population rank, land area etc. The resulting file is cleanedPop.csv.

Using the CanadianDisaster dataset we convert the rows with no values to be set as ‘unknown’ as described in the instruction manual. We then got rid of the rows with NaN values in several of the columns where it wouldn't make sense to have NaN such as location or event_category. We also looked for anomalies in the first column and found many of the to begin with ‘note’ so we removed those rows as well. We had many issues with the encoding of french-canadian names such as occurrences of accents. We solved this by using latin1 encoding.

We created new columns for city, province, country and ID which links the same disaster that took place in multiple cities. We then wrote several regex for each province and its corresponding geographical name (ie: prairie provinces) and wrote regex for typical city data splitting the city and province. It then uses the regex to get the city and province data and write out to a new row. If it has multiple locations it writes out to multiple rows. If the location was not in Canada we set the city, province and country to OTHER. We then wrote the csv out to ‘Disaster_clean_final.csv’

The script can be run by using `python norm.py --icsv 'CanadianDisasterDatabase.csv' --ocsv 'Disaster_clean_final.csv' --population 'canadianstats.csv' --stopwords 'stopwords.txt'
`
The cleaned canadianstats.csv will be found as cleanedPop.csv

We planned on using a list of stop words that we found on NLTK common stop word list in order to have an easier time at parsing out the summary column and create a keyword column[4]. However, we decided against doing so as we did not see the use of having a keyword column.

During this process, we considered using an external data set for meteorological history in Canada. After searching, we found a free public data set offered by the Canadian Government. However, after discussing, we decided not to implement a weather dimension since the data set did not offer, what we considered to be relevant data[1]. We also attempted to first do a fuzzy match to the population statistics we found, however, there would be too much data loss.

## High-Level Schematic

See Figure 1 for the full example. The Canadian Disaster Database was supplied to us by Professor Viktor[5]. We used a public data set from Statistics Canada to get an accurate representation of population in Canadian cities[3]. Pgadmin was used as our postgresql database management system throughout the project. We used Amazon Web Services (AWS) to host our sql server. It was simple to connect to it in our pgadmin interface. The Tableau desktop application was used for all data visualization purposes. We made use of weka to test various machine learning algorithms on our data set. In the end, we had six label targets: Locations, Dates, Costs, Populations, Disasters and Facts.

## Physical Design -- Data Mart

Our data mart was created in Postgresql. We hosted the database on AWS on datascience.crfo0qa9zng9.ca-central-1.rds.amazonaws.com:8080. We manually created all the tables and columns based on the guideline provided in the manual (see figure 6). We imported the main data set into a placeholder table using pgadmin ‘import’ option. We then imported the cleanedPop.csv straight into the population_dimension. We then wrote a script datamart.sql for all the ‘insert into’ statements for all the dimensions and the fact table. We created our surrogate keys for each table using the data type serial.

## OLAP Queries

We wrote the OLAP queries in SQL located in olap.sql

As per the instruction we ran several different type of queries on our data.
Roll up: Determine total estimated costs of each disaster category IN each canadian city (See figure 2 and figure 3)
Slice: Determining the estimated total cost of incident disasters IN Canada (See figure 4)
Slice: For instance determine the total number of fatalities in Ottawa/Ontario during 1999
Slice: Determining estimated total costs caused by Disasters IN each province
Dice: Determining the total cost of Technology related Incidences IN Canada (See figure 5)
Iceberg: For instance, determine the 5 cities IN Canada with the most Fires
Roll down: Determining the total number of fatalities IN each province during 1999
Dice/ Roll down: Determining the total number of fatalities caused by a Fire IN each province during 1999

# Business Intelligence Dashboard

The data visualization portion of the project was completed using the Tableau 10.5 desktop application. This allowed us to visualize our complete and cleaned data set whilst also conveniently and efficiently displaying the results of our OLAP queries (See Figures 2,3,4,5,7 and 8, 9, 10, 11). It was simple to navigate and create possible queries we could implement on our data.

# Machine Learning

We ran \copy (Select * FROM <dimension>) to 'output.csv' with csv HEADER from the postgres pgql command line in order to create a csv for weka. We ran it on each of our tables in our postgres database and combined them all into one table.

The Weka 3.8.2 desktop application was used to complete the machine learning portion of the project.
We used multiple different algorithms in weka, that each serve a different purpose. For classification we used, zeroR (baseline), J48 trees, SMO function, NaiveBayes and adaboost1 (using several of the classification previously described).

The results are given below of the algorithms run on our pre-processed data and discussion follows.

1. **rules.ZeroR**

=== Run information ===

Scheme: weka.classifiers.rules.ZeroR
Relation: Weka_file
Instances: 1381
Attributes: 26
disaster_key
day
month
year
weekend
city
province
country
canada
disaster_type
disaster_subgroup
disaster_group

disaster_category
magnitude
utility_people_affected
estimated_total_cost
normalized_total_cost
federal_dfaa_payements
provincal_dfaa_payement
municipal_cost
insurance_payements
ogd_cost
ngo_payement
fatalities
injured
evacuated

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

ZeroR predicts class value: 1052.7233888486603

Time taken to build model: 0 seconds

=== Cross-validation ===
=== Summary ===

| | |
|---|---|
| Correlation coefficient | -0.0915 |
| Mean absolute error | 1730.949 |
| Root mean squared error | 8201.4427 |
| Relative absolute error | 100 % |
| Root relative squared error | 100 % |
| Total Number of Instances | 1381 |

**Trees J48**

=== Summary ===

| | | |
|---|---|---|
| Correctly Classified Instances | 1038 | 75.1629 % |
| Incorrectly Classified Instances | 343 | 24.8371 % |
| Kappa statistic | 0.0743 | |
| Mean absolute error | 0.3693 | |
| Root mean squared error | 0.4326 | |
| Relative absolute error | 96.2825 % | |
| Root relative squared error | 98.8001 % | |

Total Number of Instances 1381

=== Detailed Accuracy By Class ===

| | TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
|---|---|---|---|---|---|---|---|---|---|
| | 0.993 | 0.941 | 0.752 | 0.993 | 0.856 | 0.161 | 0.529 | 0.754 | n |
| | 0.059 | 0.007 | 0.750 | 0.059 | 0.109 | 0.161 | 0.529 | 0.311 | y |
| Weighted Avg. | 0.752 | 0.700 | 0.751 | 0.752 | 0.663 | 0.161 | 0.529 | 0.640 | |

=== Confusion Matrix ===

| a | b | <-- classified as |
|---|---|---|
| 1017 | 7 | a = n |
| 336 | 21 | b = y |

**Functions.SMO Normalized total cost**

=== Summary ===

| | | |
|---|---|---|
| Correctly Classified Instances | 1381 | 100 % |
| Incorrectly Classified Instances | 0 | 0 % |
| Kappa statistic | 1 | |
| Mean absolute error | 0.2222 | |
| Root mean squared error | 0.2722 | |
| Relative absolute error | 109.9536 % | |
| Root relative squared error | 85.7405 % | |
| Total Number of Instances | 1381 | |

=== Detailed Accuracy By Class ===

| | TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
|---|---|---|---|---|---|---|---|---|---|
| | 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | Technology |
| | 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | Natural |
| | 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | Conflict |
| Weighted Avg. | 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | |

=== Confusion Matrix ===

| a | b | c | <-- classified as |
|---|---|---|---|
| 220 | 0 | 0 | a = Technology |
| 0 | 1132 | 0 | b = Natural |

0 0 29 | c = Conflict

**Adaboost M1 Normalized total cost**

=== Summary ===

| | | |
|---|---|---|
| Correctly Classified Instances | 667 | 48.1588 % |
| Incorrectly Classified Instances | 718 | 51.8412 % |
| Kappa statistic | 0.4084 | |
| Mean absolute error | 0.0769 | |
| Root mean squared error | 0.255 | |
| Relative absolute error | 60.6678 % | |
| Root relative squared error | 101.3063 % | |
| Total Number of Instances | 1385 | |

=== Detailed Accuracy By Class ===

| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
|---|---|---|---|---|---|---|---|---|
| 0.609 | 0.171 | 0.468 | 0.609 | 0.529 | 0.399 | 0.773 | 0.468 | ON |
| 0.391 | 0.106 | 0.371 | 0.391 | 0.381 | 0.278 | 0.669 | 0.339 | QC |
| 0.570 | 0.085 | 0.498 | 0.570 | 0.531 | 0.458 | 0.801 | 0.428 | BC |
| 0.280 | 0.037 | 0.300 | 0.280 | 0.290 | 0.251 | 0.745 | 0.222 | NB |
| 0.454 | 0.055 | 0.481 | 0.454 | 0.467 | 0.409 | 0.755 | 0.382 | AB |
| 0.500 | 0.041 | 0.548 | 0.500 | 0.523 | 0.478 | 0.801 | 0.477 | MB |
| 0.296 | 0.029 | 0.387 | 0.296 | 0.336 | 0.303 | 0.699 | 0.237 | NS |
| 0.149 | 0.030 | 0.220 | 0.149 | 0.177 | 0.143 | 0.629 | 0.100 | SK |
| 0.991 | 0.000 | 1.000 | 0.991 | 0.996 | 0.995 | 0.998 | 0.993 | 0 |
| 0.278 | 0.027 | 0.357 | 0.278 | 0.313 | 0.282 | 0.701 | 0.199 | NL |
| 0.143 | 0.004 | 0.375 | 0.143 | 0.207 | 0.224 | 0.585 | 0.112 | NT |
| 0.111 | 0.001 | 0.500 | 0.111 | 0.182 | 0.234 | 0.467 | 0.064 | YT |
| 0.500 | 0.000 | 1.000 | 0.500 | 0.667 | 0.706 | 0.717 | 0.505 | NU |
| 0.048 | 0.007 | 0.100 | 0.048 | 0.065 | 0.059 | 0.659 | 0.055 | PE |
| Weighted Avg. 0.482 | 0.076 | 0.473 | 0.482 | 0.472 | 0.406 | 0.757 | 0.405 | |

=== Confusion Matrix ===

| a | b | c | d | e | f | g | h | i | j | k | l | m | n | <-- classified as |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 167 | 38 | 16 | 6 | 12 | 15 | 7 | 5 | 0 | 7 | 0 | 0 | 0 | 1 | a = ON |
| 50 | 75 | 24 | 11 | 7 | 7 | 4 | 4 | 0 | 9 | 0 | 0 | 0 | 1 | b = QC |
| 29 | 19 | 102 | 4 | 10 | 2 | 6 | 5 | 0 | 1 | 1 | 0 | 0 | 0 | c = BC |
| 9 | 12 | 8 | 21 | 5 | 3 | 8 | 0 | 0 | 6 | 0 | 0 | 0 | 3 | d = NB |

```
 22  14  15   4  64   9   1   9   0   3   0   0   0   0 |   e = AB
 22   9   5   1  11  63   1  12   0   1   1   0   0   0 |   f = MB
 17   6  12  13   0   0  24   2   0   4   1   0   0   2 |   g = NS
 17   9   4   2  14  13   0  11   0   3   1   0   0   0 |   h = SK
  0   0   1   0   0   0   0   0 111   0   0   0   0   0 |   i = 0
 13  14   4   6   3   2   5   1   0  20   1   1   0   2 |   j = NL
  5   2   6   0   3   1   0   1   0   0   3   0   0   0 |   k = NT
  3   0   4   0   1   0   0   0   0   0   0   1   0   0 |   l = YT
  0   1   2   0   1   0   0   0   0   0   0   0   4   0 |   m = NU
  3   3   2   2   2   0   6   0   0   2   0   0   0   1 |   n = PE
```

**Naive Bayes normalized Total Cost**

=== Summary ===

| | | |
|---|---|---|
| Correctly Classified Instances | 1351 | 97.5451 % |
| Incorrectly Classified Instances | 34 | 2.4549 % |
| Kappa statistic | 0.1482 | |
| Mean absolute error | 0.0019 | |
| Root mean squared error | 0.0428 | |
| Relative absolute error | 35.1261 % | |
| Root relative squared error | 96.3977 % | |
| Total Number of Instances | 1385 | |

=== Detailed Accuracy By Class ===

| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
|---|---|---|---|---|---|---|---|---|
| 1.000 | 0.919 | 0.975 | 1.000 | 0.988 | 0.281 | 0.865 | 0.994 | 0 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.984 | 0.062 | 48319.81 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.328 | 0.001 | 77457 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.258 | 0.001 | 83524 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.950 | 0.021 | 145954 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.658 | 0.003 | 247039 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.712 | 0.003 | 334404 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.998 | 0.514 | 439291 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.890 | 0.007 | 486784 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.886 | 0.006 | 500000 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.770 | 0.003 | 1000000 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.474 | 0.002 | 1194997 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.339 | 0.001 | 1668590 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.975 | 0.028 | 1766793 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.975 | 0.028 | 1850000 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.626 | 0.002 | 3036145 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.092 | 0.001 | 3189873 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.644 | 0.002 | 5788529 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.499 | 0.001 | 14234637 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.981 | 0.053 | 21496062 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.837 | 0.004 | 29751766 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.996 | 0.278 | 48480045 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.592 | 0.002 | 72000000 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.848 | 0.005 | 76670115 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.752 | 0.003 | 204494382 |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.624 | 0.002 | 496865521 |
| 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 2018484288 |
| Weighted Avg. 0.975 | 0.894 | ? | 0.975 | ? | ? | 0.862 | 0.972 | |

=== Confusion Matrix ===

| a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | aa | <-- classified as |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1348 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | a = 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | b = 48319.81 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | c = 77457 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | d = 83524 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | e = 145954 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | f = 247039 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | g = 334404 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | h = 439291 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | i = 486784 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | j = 500000 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | k = 1000000 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | l = 1194997 |

```
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   m = 1668590
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   n = 1766793
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   o = 1850000
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   p = 3036145
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   q = 3189873
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   r = 5788529
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   s = 14234637
   2   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   t = 21496062
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   u = 29751766
   3   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   v = 48480045
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   w = 72000000
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   x = 76670115
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   y = 204494382
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0 |   z = 496865521
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   3 |   aa = 2018484288
```

2. **Trees J48**

=== Summary ===

| | | |
|---|---|---|
| Correctly Classified Instances | 1149 | 82.9603 % |
| Incorrectly Classified Instances | 236 | 17.0397 % |
| Kappa statistic | 0.5991 | |
| Mean absolute error | 0.0011 | |
| Root mean squared error | 0.0288 | |
| Relative absolute error | 27.1225 % | |
| Root relative squared error | 66.9793 % | |

Total Number of Instances 1385

=== Detailed Accuracy By Class ===

| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
|---|---|---|---|---|---|---|---|---|
| 1.000 | 0.518 | 0.804 | 1.000 | 0.891 | 0.622 | 1.000 | 1.000 | 0 |

=== Confusion Matrix ===

J48 on Disaster Subgroup

=== Summary ===

| | | |
|---|---|---|
| Correctly Classified Instances | 1361 | 98.5518 % |
| Incorrectly Classified Instances | 20 | 1.4482 % |
| Kappa statistic | 0.9645 | |
| Mean absolute error | 0.0029 | |
| Root mean squared error | 0.0423 | |
| Relative absolute error | 4.6148 % | |
| Root relative squared error | 23.8487 % | |
| Total Number of Instances | 1381 | |

=== Detailed Accuracy By Class ===

| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
|---|---|---|---|---|---|---|---|---|
| 0.675 | 0.003 | 0.871 | 0.675 | 0.761 | 0.761 | 0.956 | 0.670 | Fire |
| 0.862 | 0.009 | 0.676 | 0.862 | 0.758 | 0.758 | 0.959 | 0.612 | Explosion |
| 0.950 | 0.000 | 1.000 | 0.950 | 0.974 | 0.974 | 1.000 | 0.998 | Infrastructure failure |
| 1.000 | 0.003 | 0.999 | 1.000 | 1.000 | 0.998 | 1.000 | 1.000 | Meteorological - Hydrological |
| 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | Geological |
| 1.000 | 0.001 | 0.983 | 1.000 | 0.991 | 0.991 | 1.000 | 0.982 | Transportation accident |
| 0.950 | 0.000 | 1.000 | 0.950 | 0.974 | 0.974 | 1.000 | 0.998 | Biological |
| 1.000 | 0.002 | 0.973 | 1.000 | 0.986 | 0.986 | 1.000 | 1.000 | Hazardous Chemicals |
| 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | Arson |
| 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | Civil Incident |
| 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | Terrorist |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1.000 | 0.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | Hijacking |
| 0.000 | 0.000 | ? | 0.000 | ? | ? | 0.500 | 0.001 | Space Event |
| Weighted Avg. 0.986 | 0.003 | ? | 0.986 | ? | ? | 0.997 | 0.981 | |

=== Confusion Matrix ===

| a | b | c | d | e | f | g | h | i | j | k | l | m | <-- classified as |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 12 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | a = Fire |
| 4 | 25 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | b = Explosion |
| 0 | 0 | 19 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | c = Infrastructure failure |
| 0 | 0 | 0 | 1055 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | d = Meteorological - Hydrological |
| 0 | 0 | 0 | 0 | 57 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | e = Geological |
| 0 | 0 | 0 | 0 | 0 | 57 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | f = Transportation accident |
| 0 | 0 | 0 | 1 | 0 | 0 | 19 | 0 | 0 | 0 | 0 | 0 | 0 | g = Biological |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 73 | 0 | 0 | 0 | 0 | 0 | h = Hazardous Chemicals |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | i = Arson |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | j = Civil Incident |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 0 | k = Terrorist |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | l = Hijacking |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | m = Space Event |

**J48 Disaster SubType**

=== Summary ===

| | | |
|---|---|---|
| Correctly Classified Instances | 782 | 56.6256 % |
| Incorrectly Classified Instances | 599 | 43.3744 % |
| Kappa statistic | 0.4834 | |
| Mean absolute error | 0.0251 | |
| Root mean squared error | 0.1188 | |
| Relative absolute error | 60.4091 % | |
| Root relative squared error | 82.5846 % | |
| Total Number of Instances | 1381 | |

=== Confusion Matrix ===

# Cluster Analysis:

**Hierarchical Clustering on Euclidean Distance**

=== Run information ===

Scheme: weka.clusterers.HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"
Relation: Weka_file-weka.filters.AllFilter-weka.filters.MultiFilter-Fweka.filters.AllFilter-weka.filters.AllFilter-weka.filters.unsupervised.instance.RemoveWithValues-S0.0-Clast-Lfirst-last-weka.filters.MultiFilter-Fweka.filters.AllFilter-Fweka.filters.unsupervised.instance.RemoveWithValues -S 0.0 -C last -L first-last-weka.filters.unsupervised.attribute.Reorder-R1,2,3,4,5,6,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,7-weka.filters.unsupervised.attribute.NumericToNominal-Rfirst-last-weka.filters.unsupervised.attribute.Remove-R1-weka.filters.unsupervised.attribute.InterquartileRange-Rfirst-last-O3.0-E6.0-weka.filters.unsupervised.attribute.Remove-R26-weka.filters.unsupervised.attribute.Remove-R26-weka.filters.unsupervised.attribute.InterquartileRange-Rfirst-last-O3.0-E6.0-weka.filters.unsupervised.attribute.Remove-R26-27-weka.filters.unsupervised.attribute.InterquartileRange-Rfirst-last-O3.0-E6.0
Instances: 1381
Attributes: 27

- day
- month
- year
- weekend
- city
- country
- canada
- disaster_type
- disaster_subgroup
- disaster_group
- disaster_category
- magnitude
- utility_people_affected
- estimated_total_cost
- normalized_total_cost
- federal_dfaa_payements
- provincal_dfaa_payement
- municipal_cost
- insurance_payements
- ogd_cost
- ngo_payement
- fatalities
- injured
- evacuated
- province

Outlier
Ignored:
ExtremeValue
Test mode: Classes to clusters evaluation on training data

=== Clustering model (full training set) ===

Time taken to build model (full training data) : 18.05 seconds

=== Model and evaluation on training set ===

Clustered Instances

0 1378 (100%)
1 3 ( 0%)

Class attribute: ExtremeValue
Classes to Clusters:

0 1 <-- assigned to cluster
1378 3 | no
0 0 | yes

Cluster 0 <-- no
Cluster 1 <-- No class

Incorrectly clustered instances : 3.0 0.2172 %

**Simple K-means Nominal Extreme Values Euclidean Distance**

=== Run information ===

Scheme: weka.clusterers.SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 4 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1 -S 10
Relation:
Weka_File-weka.filters.AllFilter-weka.filters.MultiFilter-Fweka.filters.AllFilter-weka.filters.AllFilter-weka.filters.unsupervised.instance.RemoveWithValues-S0.0-Clast-Lfirst-last-weka.filters.MultiFilter-Fweka.filters.AllFilter-Fweka.filters.unsupervised.instance.RemoveWithValues -S 0.0 -C last -L

first-last-weka.filters.unsupervised.attribute.Reorder-R1,2,3,4,5,6,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,7-weka.filters.unsupervised.attribute.NumericToNominal-Rfirst-last-weka.filters.unsupervised.attribute.Remove-R1-weka.filters.supervised.instance.SMOTE-C0-K5-P100.0-S1
Instances: 1385
Attributes: 25
day
month
year
weekend
city
country
canada
disaster_type
disaster_subgroup
disaster_group
disaster_category
magnitude
utility_people_affected
estimated_total_cost
federal_dfaa_payements
provincal_dfaa_payement
municipal_cost
insurance_payements
ogd_cost
ngo_payement
fatalities
injured
evacuated
province
Ignored:
normalized_total_cost
Test mode: Classes to clusters evaluation on training data

=== Clustering model (full training set) ===

kMeans
======

Number of iterations: 4
Within cluster sum of squared errors: 9416.0

Initial starting points (random):

Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0 901 ( 65%)
1 202 ( 15%)
2 230 ( 17%)
3 52 ( 4%)

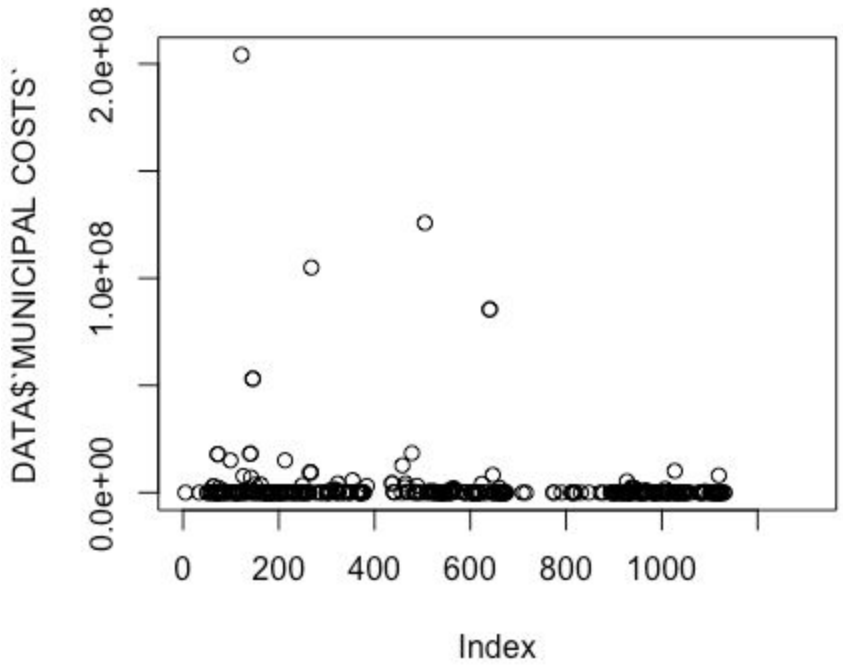
Class attribute: normalized_total_cost
Classes to Clusters:

Cluster 0 <-- 0
Cluster 1 <-- 7699216
Cluster 2 <-- 418551.2
Cluster 3 <-- 460047.9

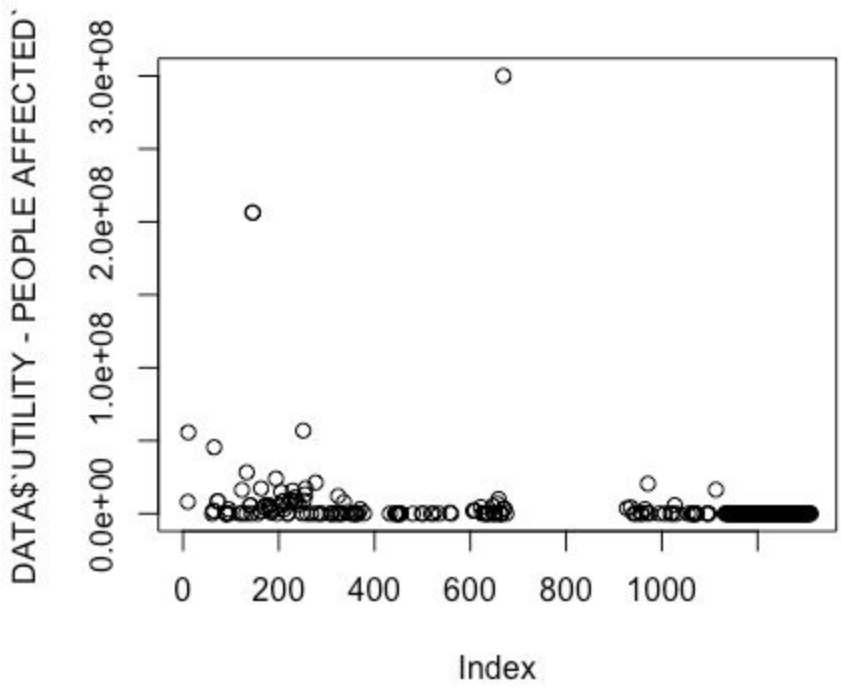
Incorrectly clustered instances : 707.0 51.0469 %

**Anomaly Detection:**
We used both weka and R for anomaly detection. In weka we did a cluster analysis and took a cluster that had minimal results as an outlier. In R we imported the CSV then ran a plot on several of the columns data in order to find outliers. The r file is titled outlier.r Here are a sample of some of the outliers we found.

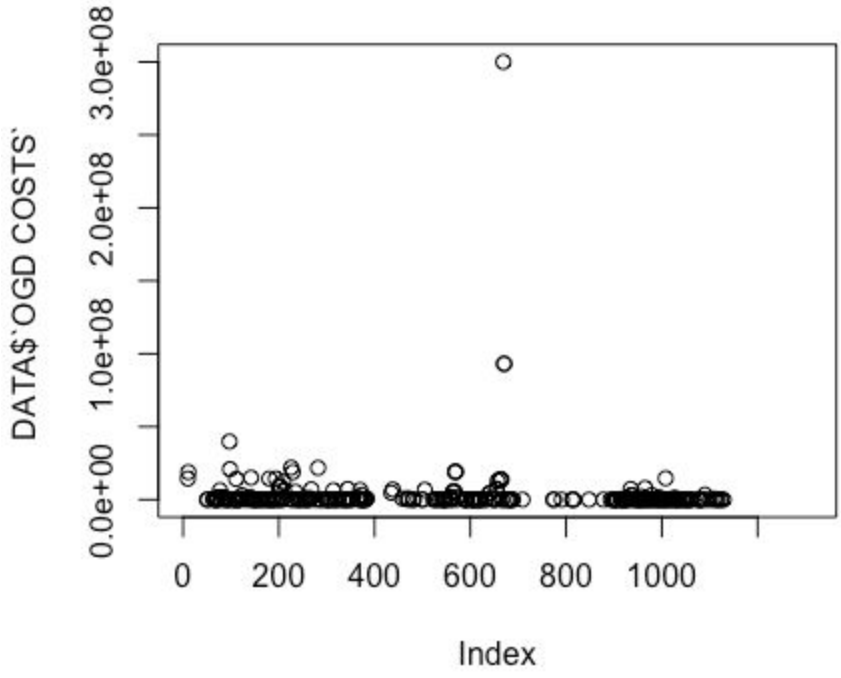
Outlier of Municipal costs:




Outlier of Municipal People affected:




Outlier of OGD Cost:



Missing values globally replaced with mean/mode

Final cluster centroids:

| | | Cluster# | | | |
|---|---|---|---|---|---|
| Attribute | Full Data | 0 | 1 | 2 | 3 |
| | (1381.0) | (294.0) | (1021.0) | (57.0) | (9.0) |
| utility_people_affected | 0 | 0 | 0 | 0 | 0 |
| municipal_cost | 0 | 0 | 0 | 0 | 0 |
| fatalities | 0 | 2 | 0 | 1 | 3 |
| injured | 0 | 0 | 0 | 0 | 13 |

Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0 294 ( 21%)
1 1021 ( 74%)
2 57 ( 4%)
3 9 ( 1%)

Class attribute: normalized_total_cost

Final cluster centroids:

| | | Cluster# | | | |
|---|---|---|---|---|---|
| Attribute | Full Data | 0 | 1 | 2 | 3 |
| | (1381.0) | (294.0) | (1021.0) | (57.0) | (9.0) |
| utility_people_affected | 0 | 0 | 0 | 0 | 0 |
| normalized_total_cost | 0 | 0 | 0 | 0 | 0 |
| municipal_cost | 0 | 0 | 0 | 0 | 0 |
| fatalities | 0 | 2 | 0 | 1 | 3 |

injured 0 0 0 0 13

Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0 294 ( 21%)
1 1021 ( 74%)
2 57 ( 4%)
3 9 ( 1%)

Class attribute: insurance_payements

kMeans
======

Number of iterations: 4
Within cluster sum of squared errors: 1948.0

Initial starting points (random):

Cluster 0: 0,0,2815608.2,0,0,0,43595000,0,0,1,4,200
Cluster 1: 0,0,0,0,0,0,0,0,0,0,0,0
Cluster 2: 0,0,0,0,0,0,0,0,0,0,0,1500
Cluster 3: 0,0,0,0,0,0,0,0,0,1,0,0

Missing values globally replaced with mean/mode

Final cluster centroids:

| | Cluster# | | | | |
|---|---|---|---|---|---|
| Attribute | Full Data | 0 | 1 | 2 | 3 |
| | (1381.0) | (3.0) | (1317.0) | (4.0) | (57.0) |
| magnitude | 0 | 0 | 0 | 0 | 0 |
| utility_people_affected | 0 | 4828750 | 0 | 0 | 0 |

| | | | | | |
|---|---|---|---|---|---|
| normalized_total_cost | 0 | 304647564.1 | 0 | 0 | 0 |
| federal_dfaa_payements | 0 | 665387416 | 0 | 0 | 0 |
| provincal_dfaa_payement | 0 | 60222575 | 0 | 0 | 0 |
| municipal_cost | 0 | 2018484288 | 0 | 0 | 0 |
| insurance_payements | 0 | 1712248000 | 0 | 0 | 0 |
| ogd_cost | 0 | 166306876 | 0 | 0 | 0 |
| ngo_payement | 0 | 13071278 | 0 | 0 | 0 |
| fatalities | 0 | 35 | 0 | 0 | 1 |
| injured | 0 | 945 | 0 | 0 | 0 |
| evacuated | 0 | 17800 | 0 | 1500 | 0 |

Time taken to build model (full training data) : 0.11 seconds

=== Model and evaluation on training set ===

Clustered Instances

| | |
|---|---|
| 0 | 3 ( 0%) |
| 1 | 1317 ( 95%) |
| 2 | 4 ( 0%) |
| 3 | 57 ( 4%) |

Class attribute: estimated_total_cost
Classes to Clusters:

**Discussion - Machine learning**

J48 is an open source Java implementation of simple C4.5 decision tree algorithm. We saw that when we ran J48 on province(nominal) we got 48% of the results as incorrectly classified, whereas only 51% were correctly classified. Those results are definitely not good. We realize province is not a measure that can be correctly classified based on attributes like disaster_group or disaster subgroup. We also realized running the algorithm on pruned data, gave us better results than reduced error pruning. While running the J48 algorithm on Country gave us amazing results with more than 95% correctly classified instances whereas only 4% of incorrectly classified partly because having the city attribute makes it quite predictive even if disaster subgroup, disaster_group aren't that good for classifying disasters to country. Data was

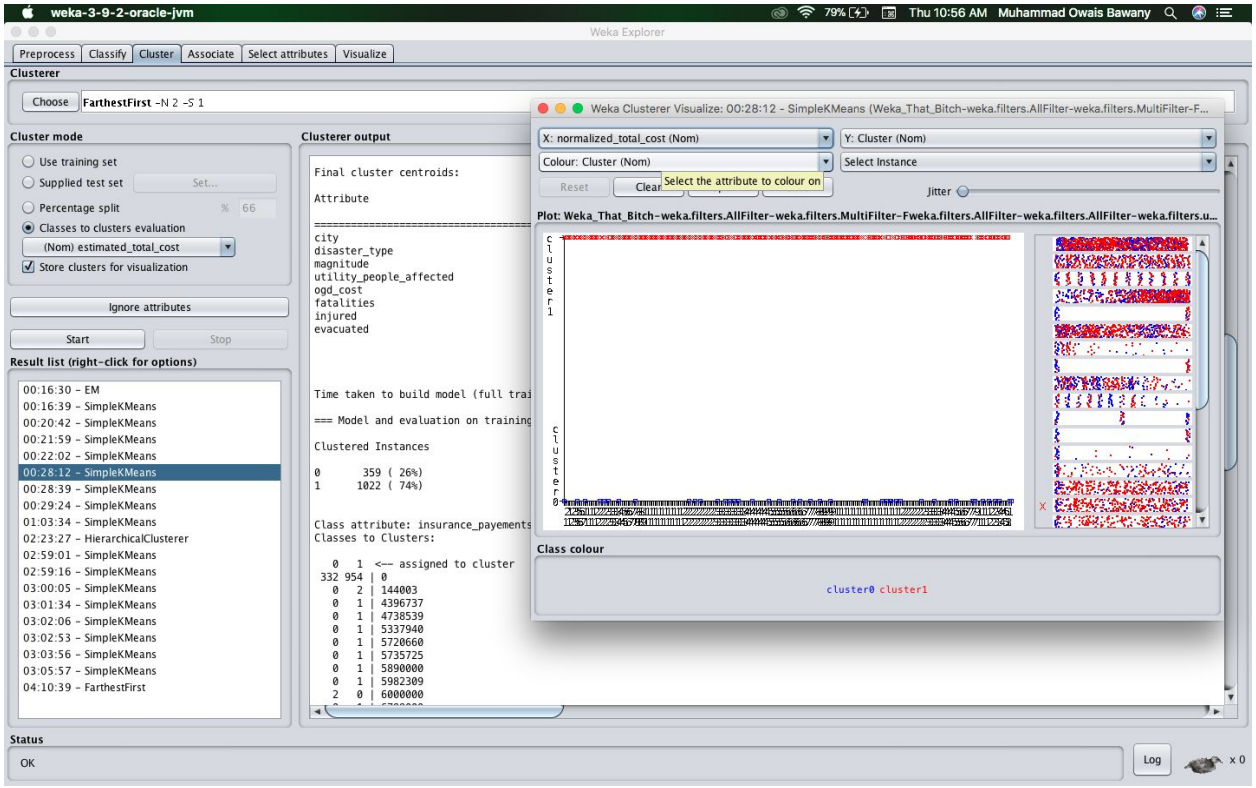
highly imbalanced too with most of the events in Canada, probably overfitting the data. A very interesting result was for J48 on Canada (y/n) which gave us more than 99% of correctly classified instances well not surprising enough given that our classification attribute was also country which is a clear indicative if the country is Canada or not. To do some realistic classifications we ran J48 for disaster_subgroup since disaster group has some correlation to disaster subgroup, the results were still not the best with 61% correctly classified instances and only 38% of incorrectly classified instances. We realize disaster subgroup is a subset of disaster group and as we'd expect classifying for disaster group when the disaster subgroup is known will give us very accurate results, which it did. Disaster group on J48 gave us more than 98% of correctly classified instances.

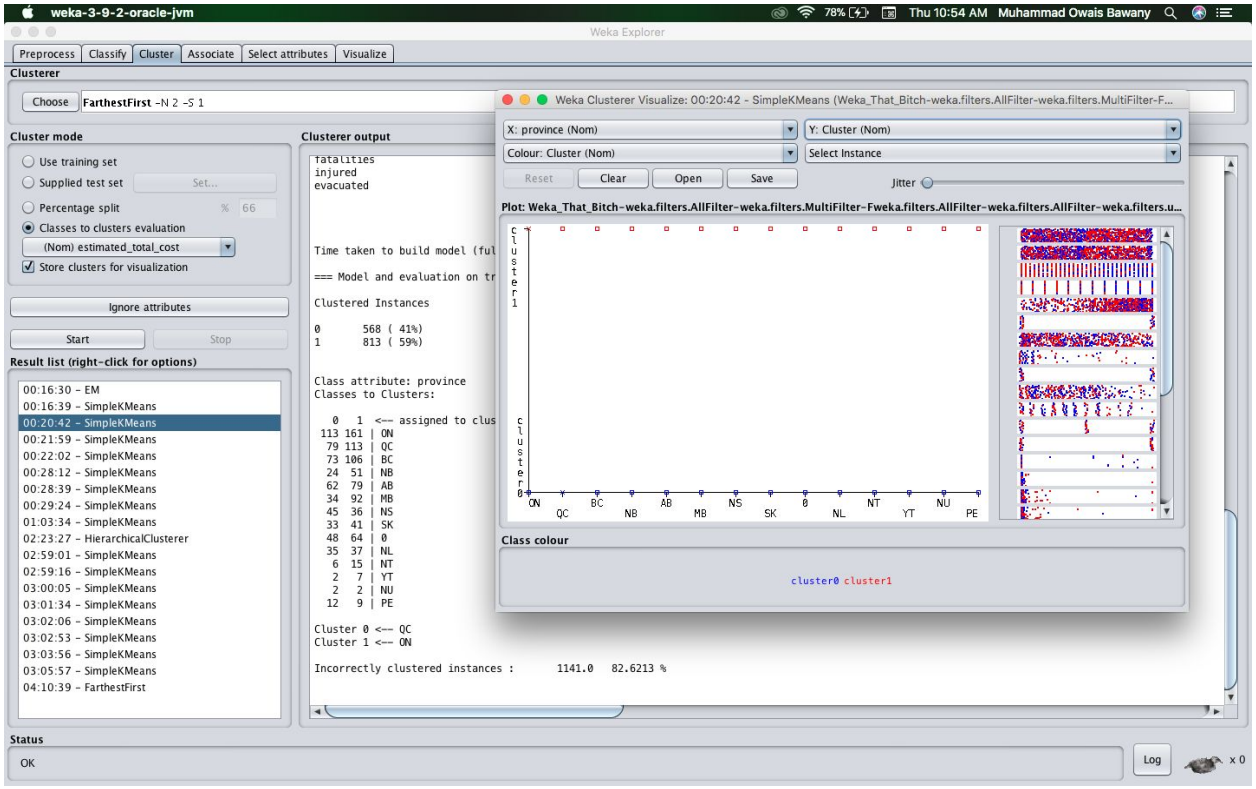
While J48 is one of the classification algorithms for accuracy, SMO is a good choice to run some tests on. From the very first test run on SMO for Canada (y/n) we saw how good J48 was compared to SMO. SMO only classified 72% of the instances while 27% of them incorrectly. Comparing to J48 that gave us 99% of correctly classified instances and less than a percent of incorrectly classified instances.

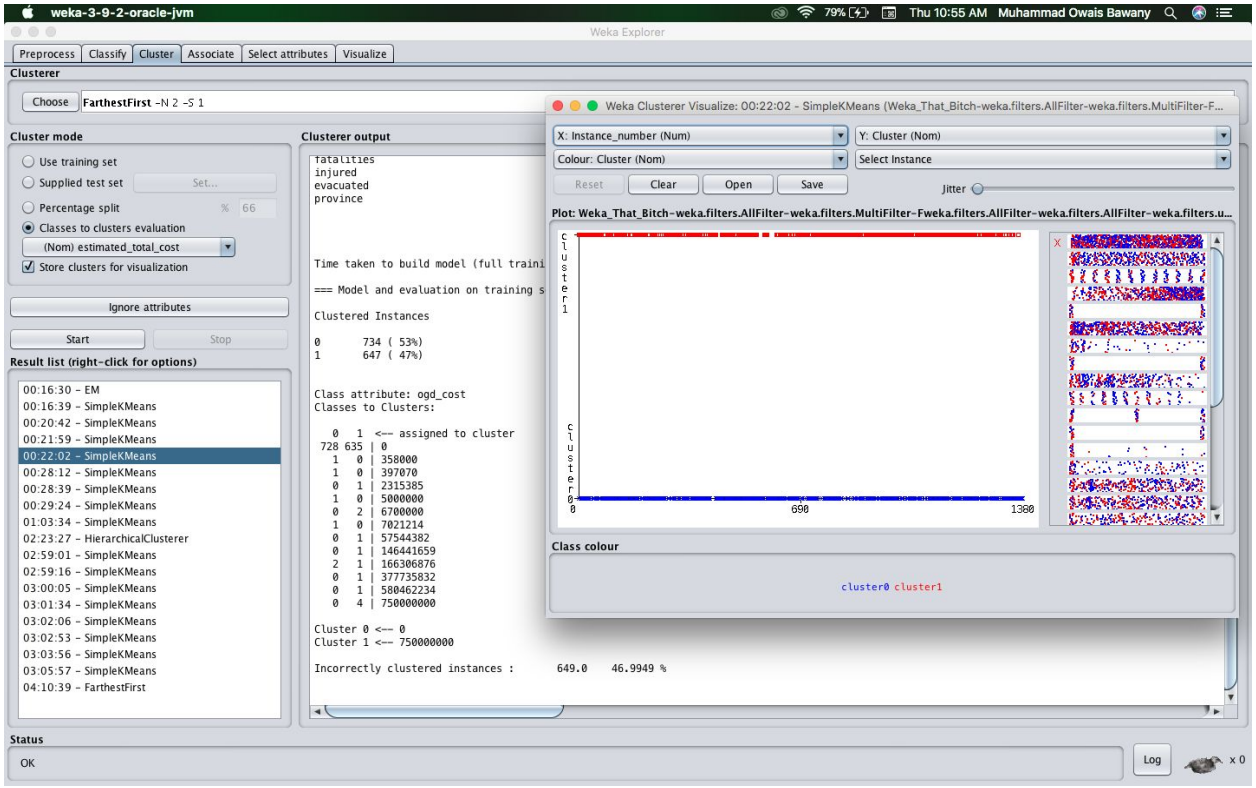
Then we used an ensemble, ADABOOST1, which should do better but surprisingly it does worse than J48. Normally we expect bagging, boosting and ensemble to improve our results. Initially with adaboost for provinces we only got 46% of correctly classified instances but after preprocessing with SMOTE, our results improved to 48% (not a bid difference) which must've balanced the data a bit.

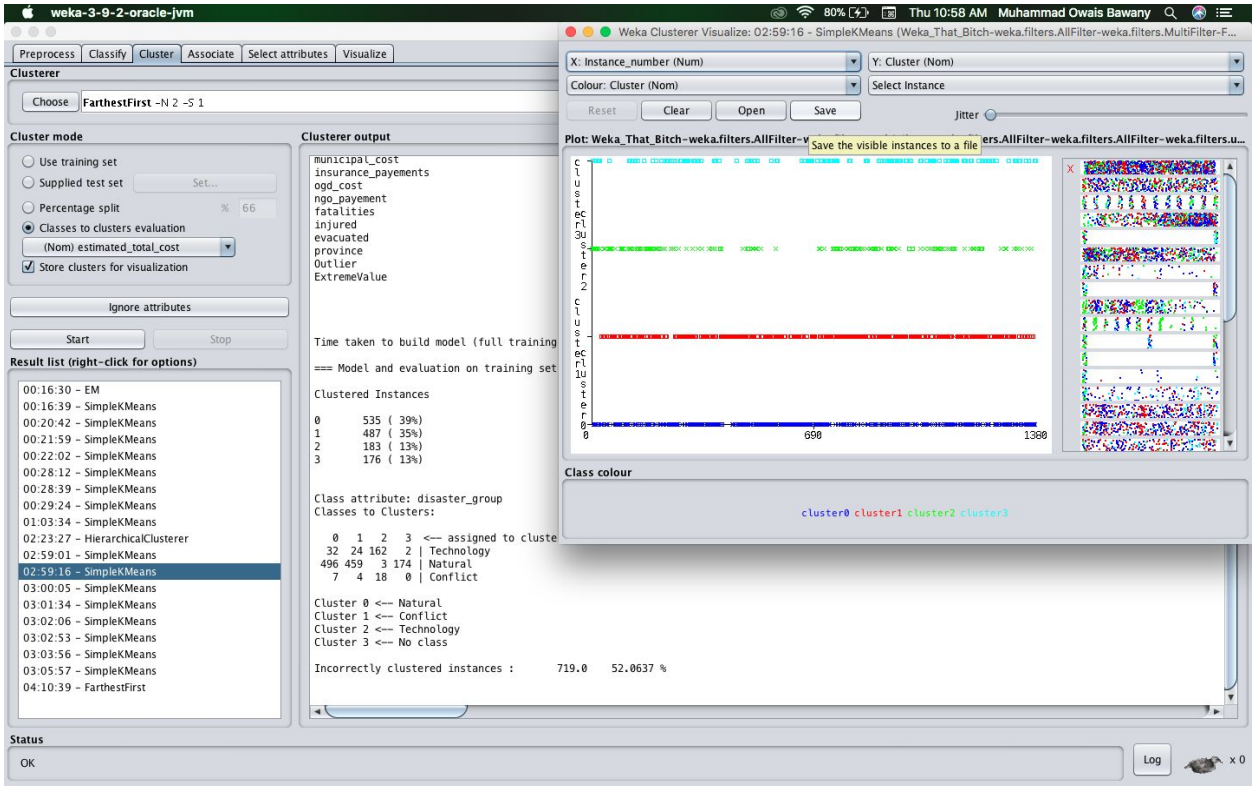
NaiveBayes Algorithm works with decision trees, and running it on disaster type gave poor results because decision trees are the not the best at determining disaster subgroup occurs so little it often gets pruned out. Hence we get more than 70% of incorrectly classified instances.Decision trees are neat because they tell you what inputs are the best predictors of the outputs so often decision trees can guide you to find if there is a statistical relationship between a given input to the output and how strong that relationship is.

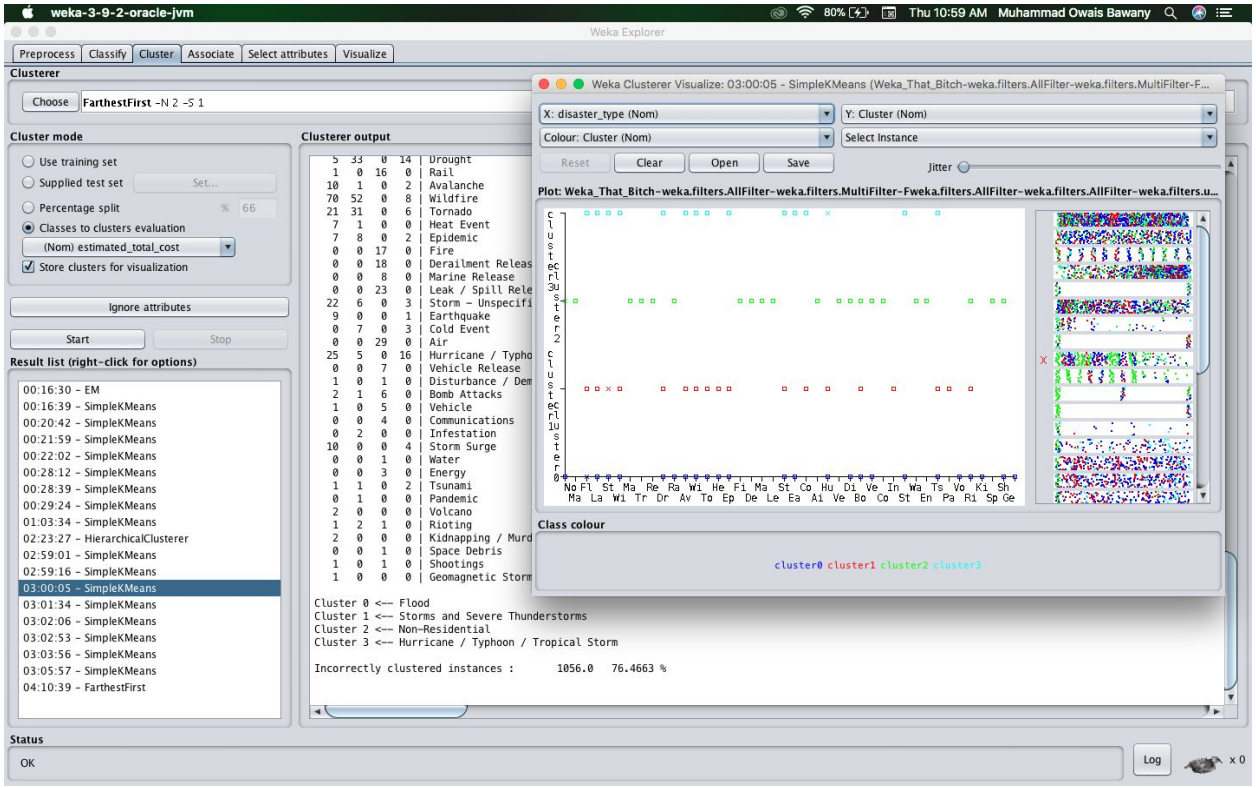
## Clustering results




weka-3-9-2-oracle-jvm
Thu 10:55 AM Muhammad Owais Bawany
Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Clusterer
Choose FarthestFirst -N 2 -S 1
Cluster mode
Use training set
Supplied test set Set...
Percentage split % 66
Classes to clusters evaluation
(Nom) estimated_total_cost
Store clusters for visualization
Ignore attributes
Start Stop
Result list (right-click for options)
00:16:30 - EM
00:16:39 - SimpleKMeans
00:20:42 - SimpleKMeans
00:21:59 - SimpleKMeans
00:22:02 - SimpleKMeans
00:28:12 - SimpleKMeans
00:28:39 - SimpleKMeans
00:29:24 - SimpleKMeans
01:03:34 - SimpleKMeans
02:23:27 - HierarchicalClusterer
02:59:01 - SimpleKMeans
02:59:16 - SimpleKMeans
03:00:05 - SimpleKMeans
03:01:34 - SimpleKMeans
03:02:06 - SimpleKMeans
03:02:53 - SimpleKMeans
03:03:56 - SimpleKMeans
03:05:57 - SimpleKMeans
04:10:39 - FarthestFirst
Clusterer output
fatalities
injured
evacuated
province
Time taken to build model (full traini
=== Model and evaluation on training s
Clustered Instances
0 734 ( 53%)
1 647 ( 47%)
Class attribute: ogd_cost
Classes to Clusters:
0 1 <-- assigned to cluster
728 635 | 0
1 0 | 358000
1 0 | 397070
0 1 | 2315385
1 0 | 5000000
0 2 | 6700000
1 0 | 7021214
0 1 | 57544382
0 1 | 146441659
2 1 | 166306876
0 1 | 377735832
0 1 | 580462234
0 4 | 750000000
Cluster 0 <-- 0
Cluster 1 <-- 750000000
Incorrectly clustered instances : 649.0 46.9949 %
Weka Clusterer Visualize: 00:22:02 - SimpleKMeans (Weka_That_Bitch-weka.filters.AllFilter-weka.filters.MultiFilter-F...
X: Instance_number (Num)
Y: Cluster (Nom)
Colour: Cluster (Nom)
Select Instance
Reset Clear Open Save
Jitter
Plot: Weka_That_Bitch-weka.filters.AllFilter-weka.filters.MultiFilter-Fweka.filters.AllFilter-weka.filters.AllFilter-weka.filters.u...
Cluster1
Cluster0
0 690 1380
X
Class colour
cluster0 cluster1
Status
OK
Log
x 0


weka-3-9-2-oracle-jvm
Thu 10:58 AM Muhammad Owais Bawany
Weka Clusterer Visualize: 02:59:16 - SimpleKMeans (Weka_That_Bitch-weka.filters.AllFilter-weka.filters.MultiFilter-F...
Preprocess Classify Cluster Associate Select attributes Visualize
Clusterer
Choose FarthestFirst -N 2 -S 1
Cluster mode
Use training set
Supplied test set Set...
Percentage split % 66
Classes to clusters evaluation
(Nom) estimated_total_cost
Store clusters for visualization
Ignore attributes
Start Stop
Result list (right-click for options)
00:16:30 - EM
00:16:39 - SimpleKMeans
00:20:42 - SimpleKMeans
00:21:59 - SimpleKMeans
00:22:02 - SimpleKMeans
00:28:12 - SimpleKMeans
00:28:39 - SimpleKMeans
00:29:24 - SimpleKMeans
01:03:34 - SimpleKMeans
02:23:27 - HierarchicalClusterer
02:59:01 - SimpleKMeans
02:59:16 - SimpleKMeans
03:00:05 - SimpleKMeans
03:01:34 - SimpleKMeans
03:02:06 - SimpleKMeans
03:02:53 - SimpleKMeans
03:03:56 - SimpleKMeans
03:05:57 - SimpleKMeans
04:10:39 - FarthestFirst
Clusterer output
municipal_cost
insurance_payements
ogd_cost
ngo_payement
fatalities
injured
evacuated
province
Outlier
ExtremeValue
Time taken to build model (full training
=== Model and evaluation on training set
Clustered Instances
0 535 ( 39%)
1 487 ( 35%)
2 183 ( 13%)
3 176 ( 13%)
Class attribute: disaster_group
Classes to Clusters:
0 1 2 3 <-- assigned to cluste
32 24 162 2 | Technology
496 459 3 174 | Natural
7 4 18 0 | Conflict
Cluster 0 <-- Natural
Cluster 1 <-- Conflict
Cluster 2 <-- Technology
Cluster 3 <-- No class
Incorrectly clustered instances : 719.0 52.0637 %
X: Instance_number (Num)
Y: Cluster (Nom)
Colour: Cluster (Nom)
Select Instance
Reset Clear Open Save
Jitter
Save the visible instances to a file
Cluster3
Cluster2
Cluster1
Cluster0
0 690 1380
X
Class colour
cluster0 cluster1 cluster2 cluster3
Status
OK
Log
x 0


weka-3-9-2-oracle-jvm
Thu 10:59 AM Muhammad Owais Bawany
Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Clusterer
Choose FarthestFirst -N 2 -S 1
Cluster mode
Use training set
Supplied test set Set...
Percentage split % 66
Classes to clusters evaluation
(Nom) estimated_total_cost
Store clusters for visualization
Ignore attributes
Start Stop
Result list (right-click for options)
00:16:30 - EM
00:16:39 - SimpleKMeans
00:20:42 - SimpleKMeans
00:21:59 - SimpleKMeans
00:22:02 - SimpleKMeans
00:28:12 - SimpleKMeans
00:28:39 - SimpleKMeans
00:29:24 - SimpleKMeans
01:03:34 - SimpleKMeans
02:23:27 - HierarchicalClusterer
02:59:01 - SimpleKMeans
02:59:16 - SimpleKMeans
03:00:05 - SimpleKMeans
03:01:34 - SimpleKMeans
03:02:06 - SimpleKMeans
03:02:53 - SimpleKMeans
03:03:56 - SimpleKMeans
03:05:57 - SimpleKMeans
04:10:39 - FarthestFirst
Clusterer output
5 33 0 14 | Drought
1 0 16 0 | Rail
10 1 0 2 | Avalanche
70 52 0 8 | Wildfire
21 31 0 6 | Tornado
7 1 0 0 | Heat Event
7 8 0 2 | Epidemic
0 0 17 0 | Fire
0 0 18 0 | Derailment Releas
0 0 8 0 | Marine Release
0 0 23 0 | Leak / Spill Rele
22 6 0 3 | Storm - Unspecifi
9 0 0 1 | Earthquake
0 7 0 3 | Cold Event
0 0 29 0 | Air
25 5 0 16 | Hurricane / Typho
0 0 7 0 | Vehicle Release
1 0 1 0 | Disturbance / Dem
2 1 6 0 | Bomb Attacks
1 0 5 0 | Vehicle
0 0 4 0 | Communications
0 2 0 0 | Infestation
10 0 0 4 | Storm Surge
0 0 1 0 | Water
0 0 3 0 | Energy
1 1 0 2 | Tsunami
0 1 0 0 | Pandemic
2 0 0 0 | Volcano
1 2 1 0 | Rioting
2 0 0 0 | Kidnapping / Murd
0 0 1 0 | Space Debris
1 0 1 0 | Shootings
1 0 0 0 | Geomagnetic Storm
Cluster 0 <-- Flood
Cluster 1 <-- Storms and Severe Thunderstorms
Cluster 2 <-- Non-Residential
Cluster 3 <-- Hurricane / Typhoon / Tropical Storm
Incorrectly clustered instances : 1056.0 76.4663 %
Weka Clusterer Visualize: 03:00:05 - SimpleKMeans (Weka_That_Bitch-weka.filters.AllFilter-weka.filters.MultiFilter-F...
X: disaster_type (Nom)
Y: Cluster (Nom)
Colour: Cluster (Nom)
Select Instance
Reset Clear Open Save
Jitter
Plot: Weka_That_Bitch-weka.filters.AllFilter-weka.filters.MultiFilter-Fweka.filters.AllFilter-weka.filters.AllFilter-weka.filters.u...
Class colour
cluster0 cluster1 cluster2 cluster3
Status
OK
Log
x 0

# Appendix


Figure 1: High-level schematic

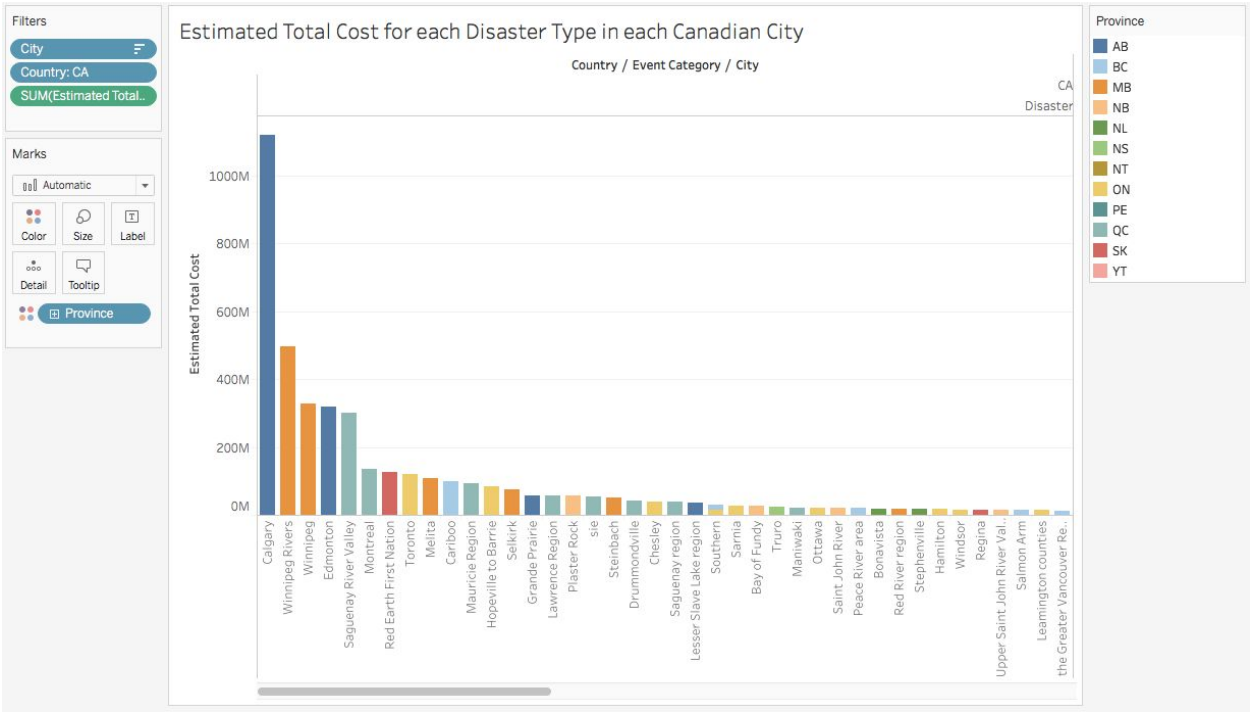



Figure 2: Example of visualizing a roll-up query in Tableau using a simple bar graph. This shows the estimated total cost of both disaster categories. OLAP query: determine total estimated costs of each disaster category IN each canadian city

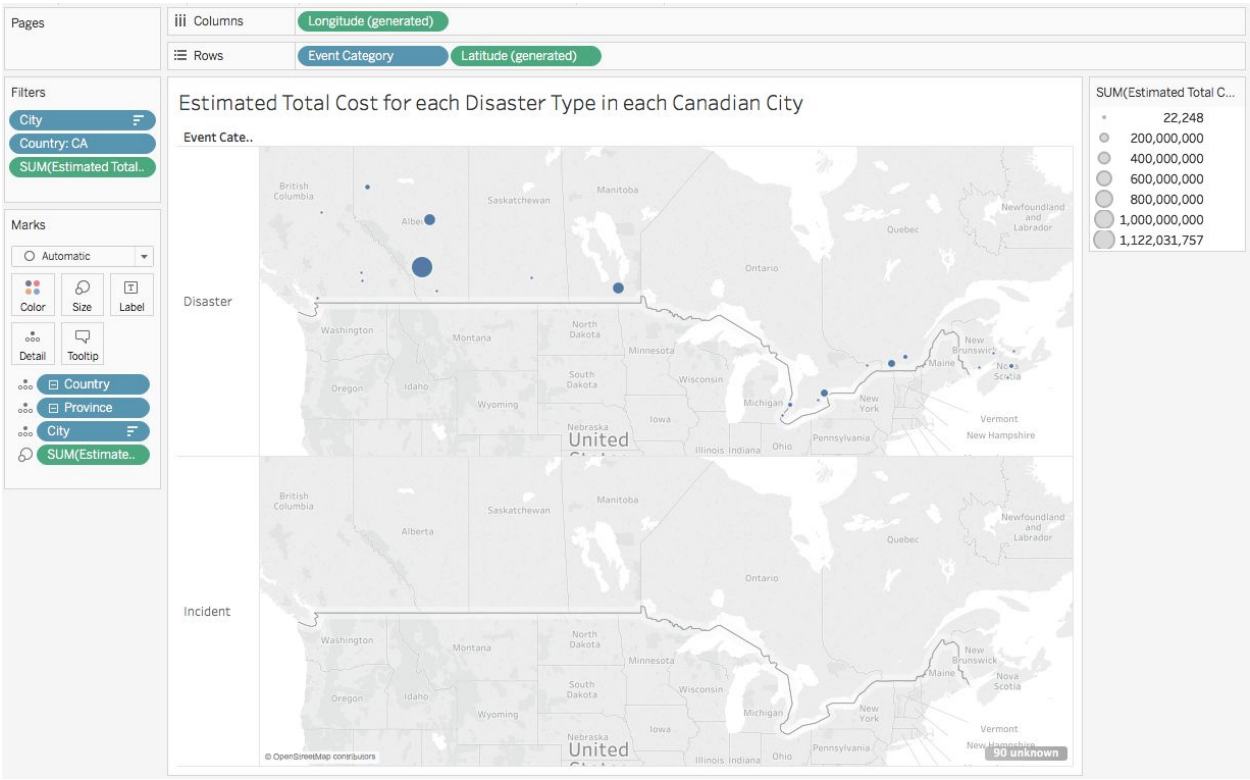


Figure 3: Example of visualizing a roll-up query in Tableau using a geographical representation of each city. This shows the estimated total cost of both disaster categories. OLAP query: determine total estimated costs of each disaster category IN each canadian city

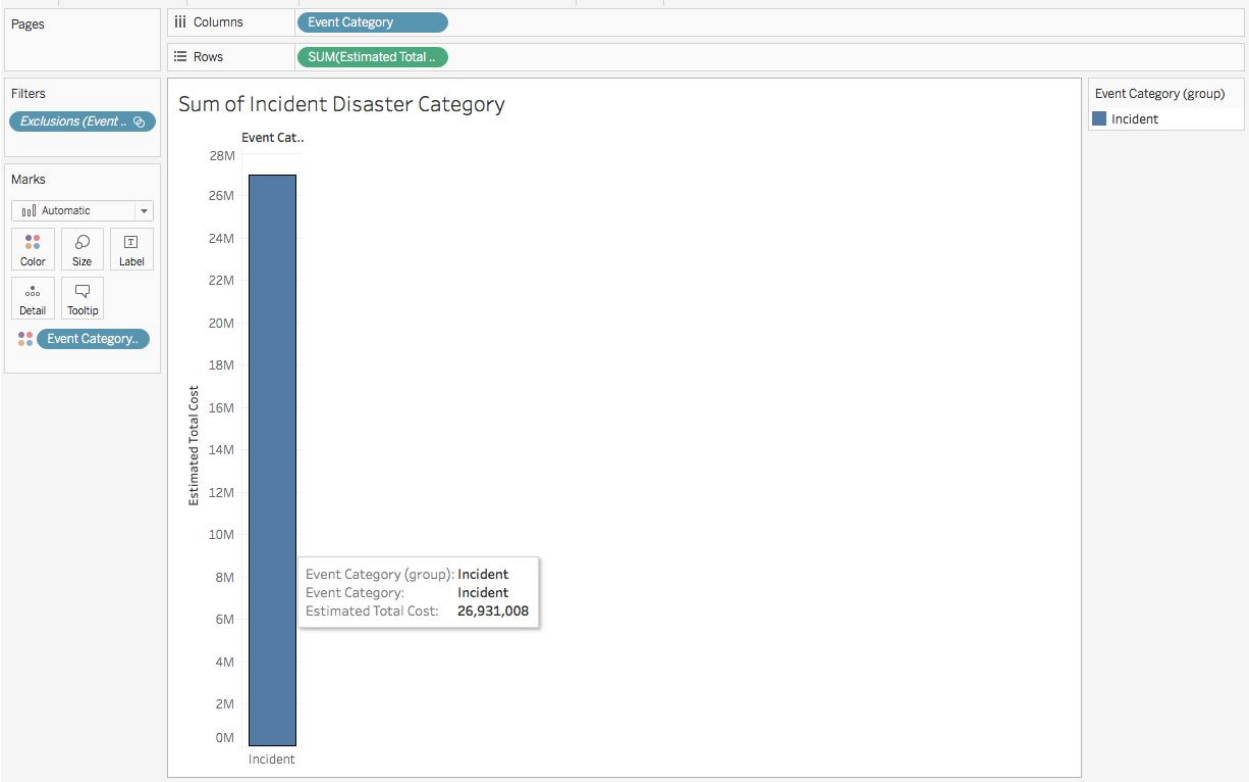



Figure 4: Example of visualizing a slice query in Tableau using a simple bar graph displaying the total estimated cost of incident disasters. OLAP query: determining the estimated total cost of incident disasters IN Canada

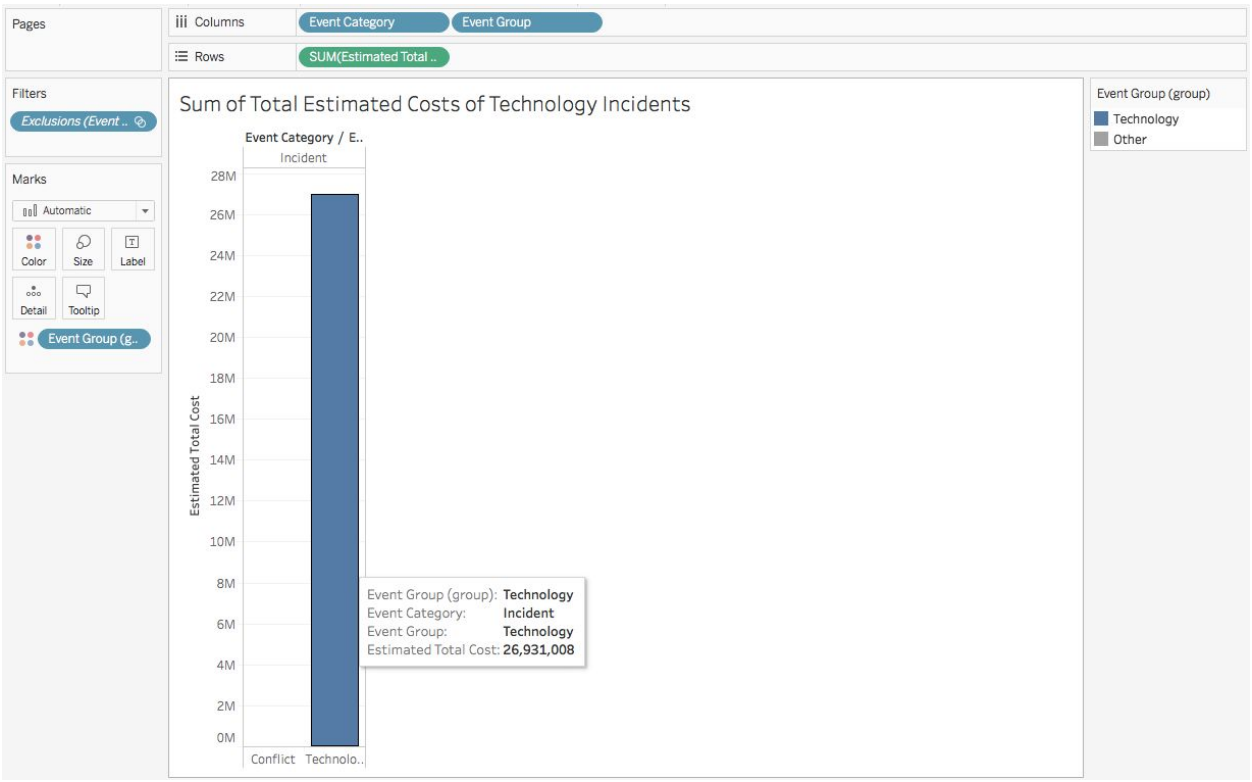



Figure 5: Example of visualizing a dice query in Tableau using a simple bar graph displaying the total estimated cost of incident technology disasters. OLAP query: determining the total cost of Technology related Incidences IN Canada

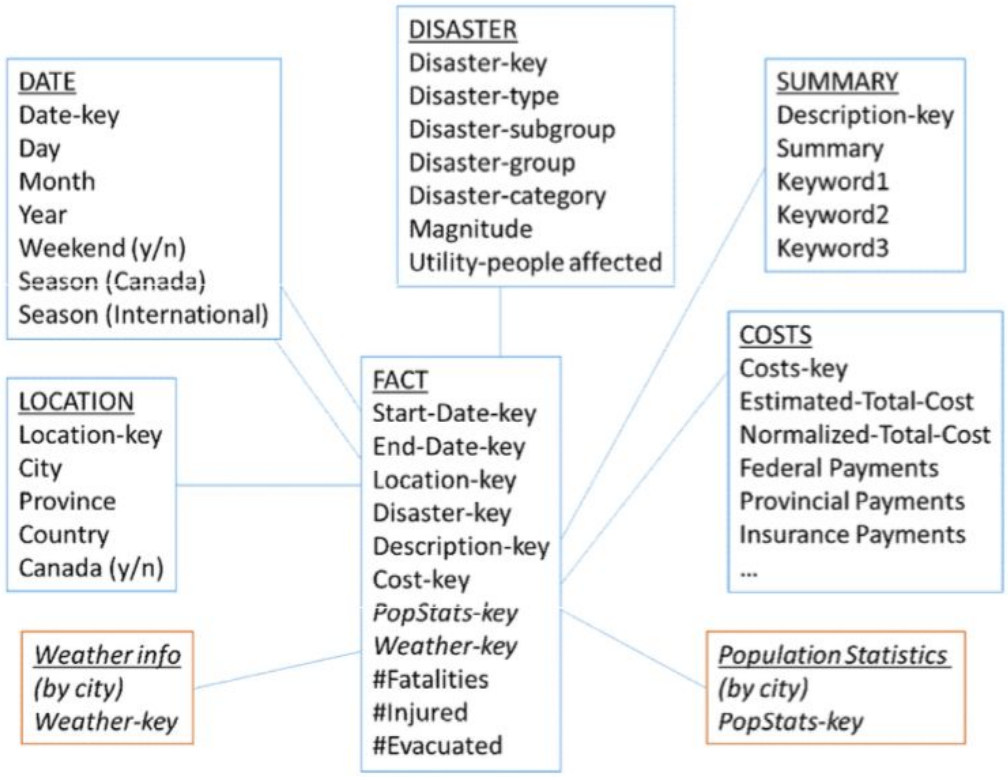


Figure 6: Data mart star schema guideline based on the manual

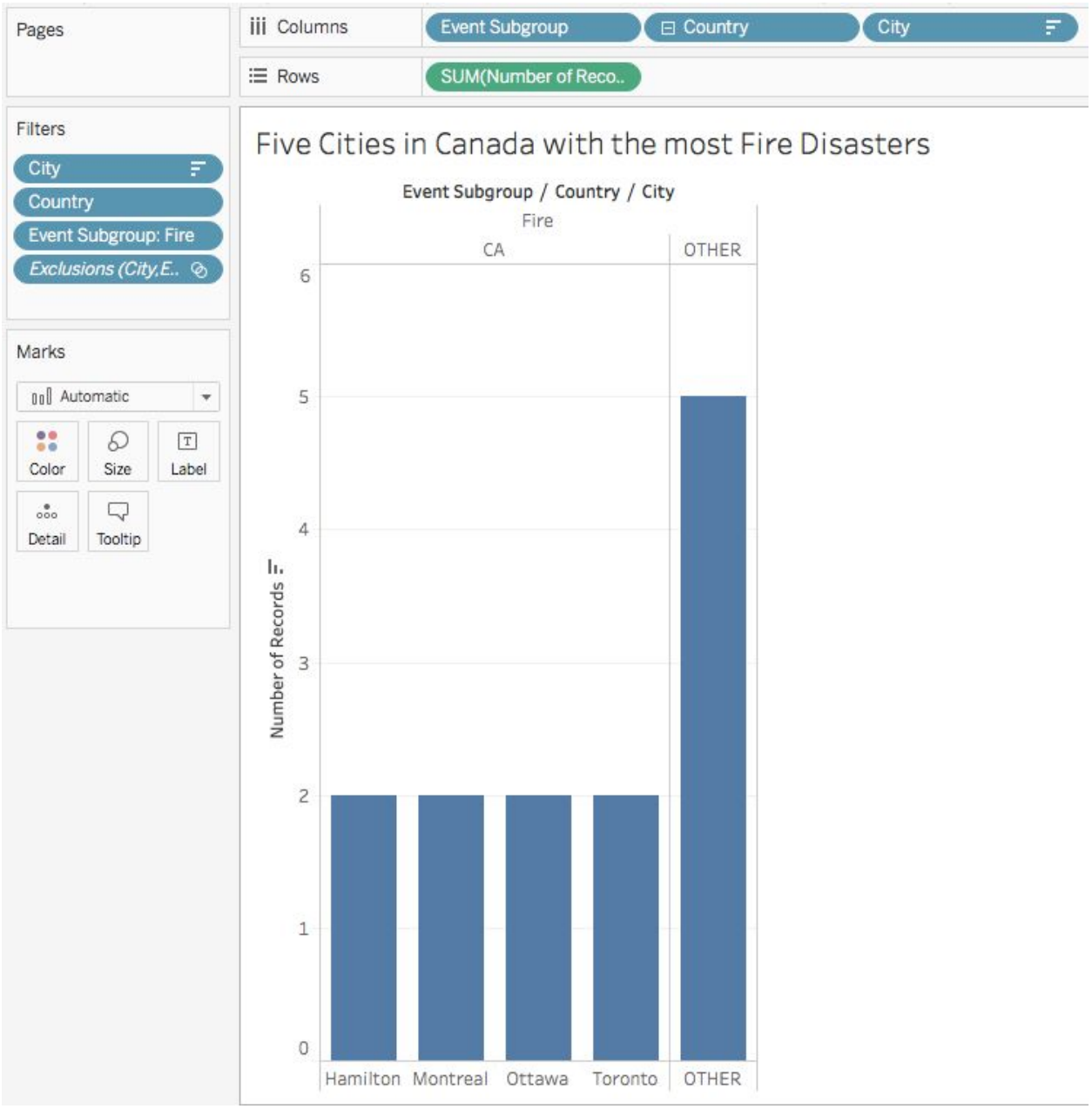



Figure 7: Example of visualizing a dice query in Tableau using a simple bar graph displaying the five cities in Canada with the most fire disasters. OLAP query: determine the 5 cities IN Canada with the most fires

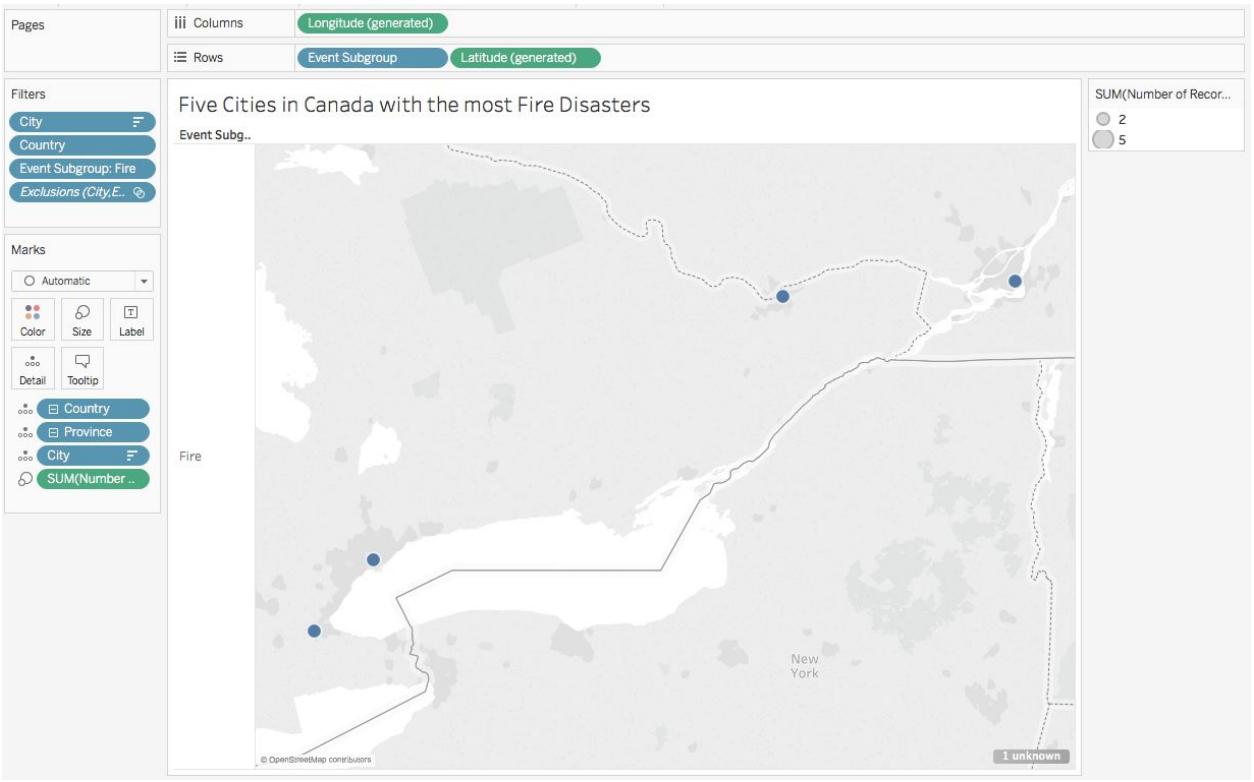



Figure 8: Example of visualizing a dice query in Tableau using a geographical representation of the five cities in Canada with the most fire disasters. OLAP query: determine the 5 cities IN Canada with the most fires

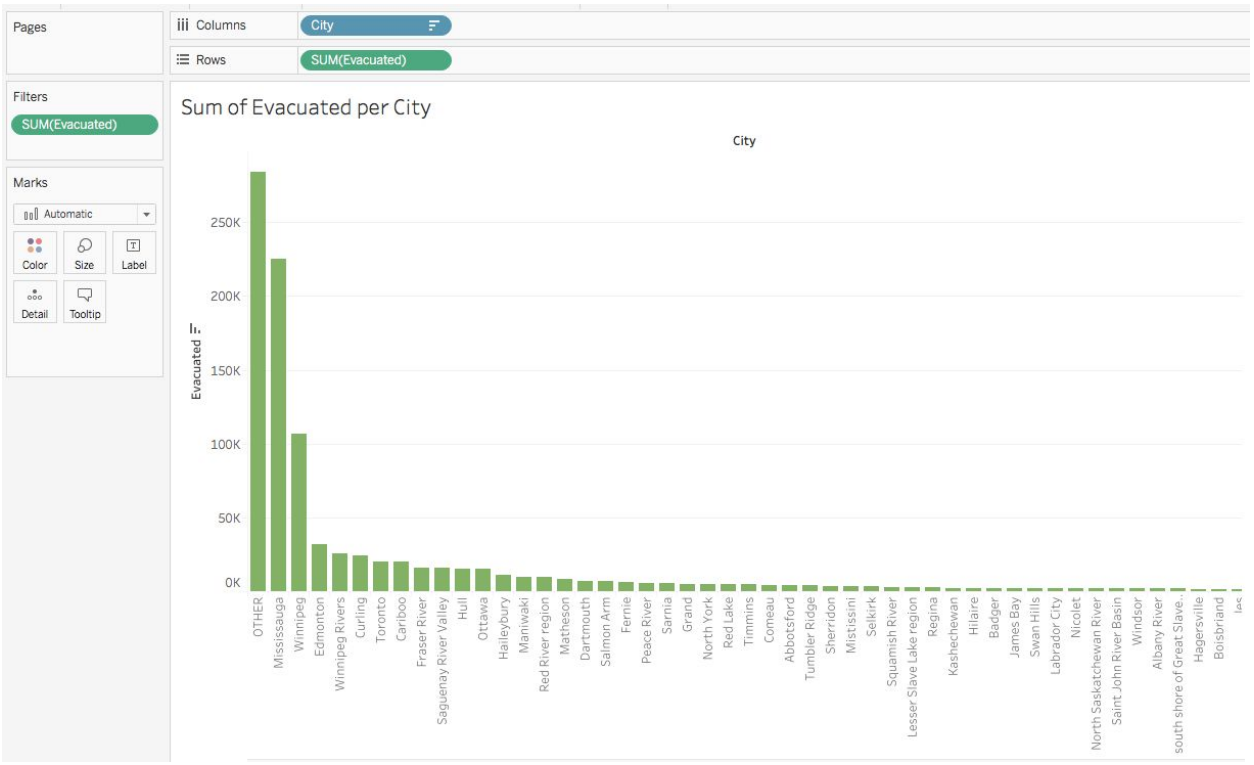


Figure 9: Example of visualizing a sum of evacuated people per Canadian cities

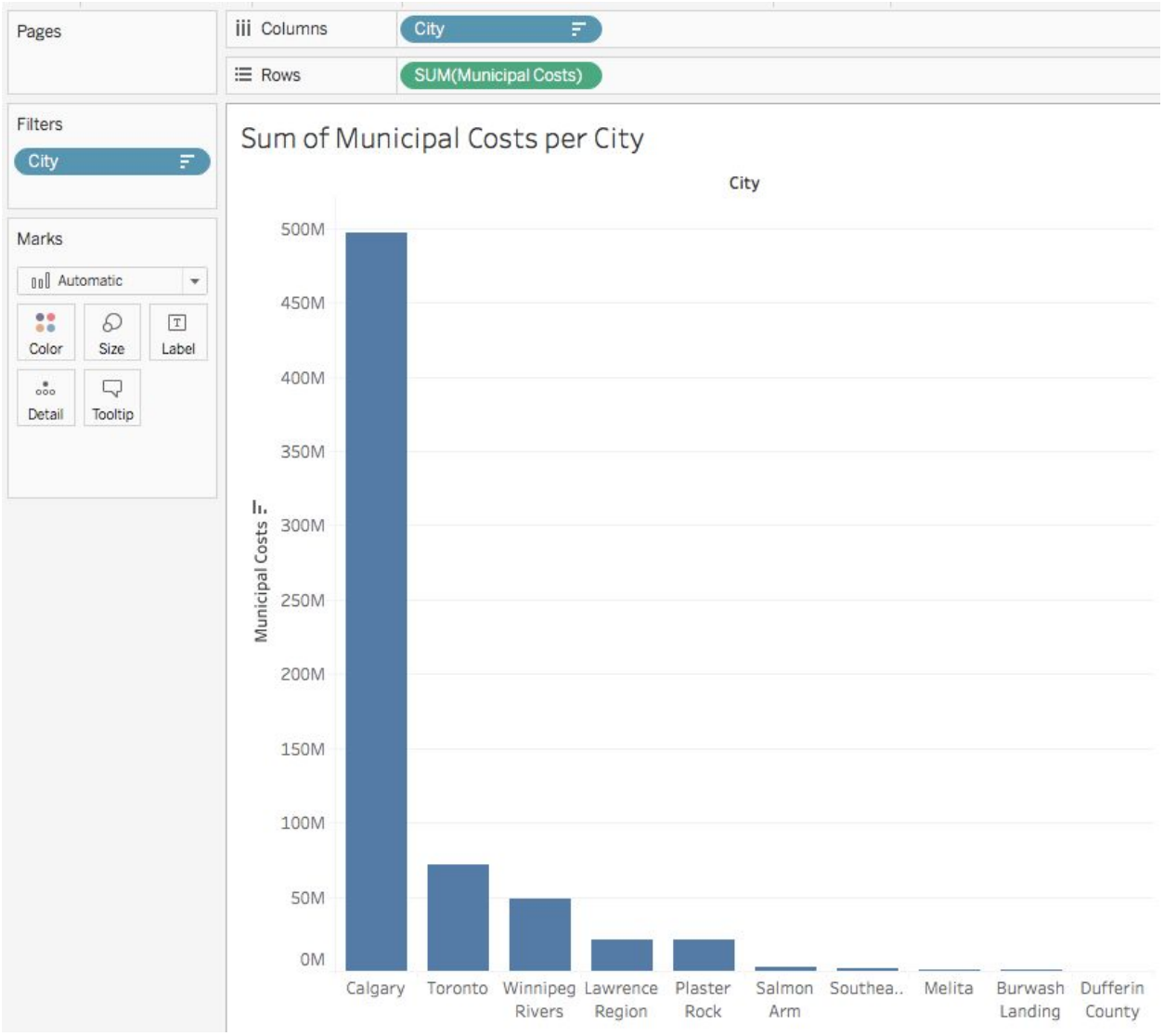



Figure 10: Example of visualizing a sum of municipal costs per city

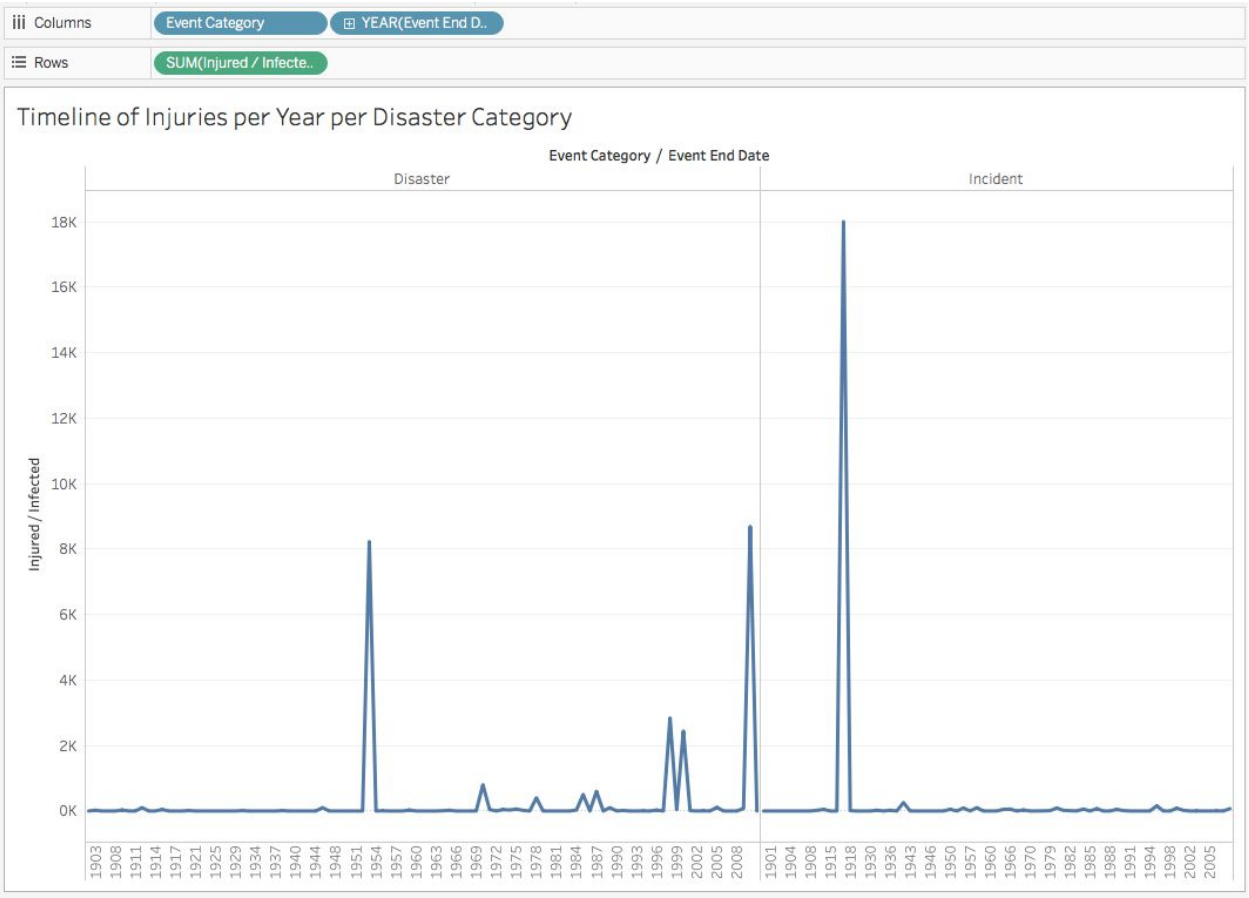



Figure 11: Example of visualizing a timeline of injuries per year per disaster category

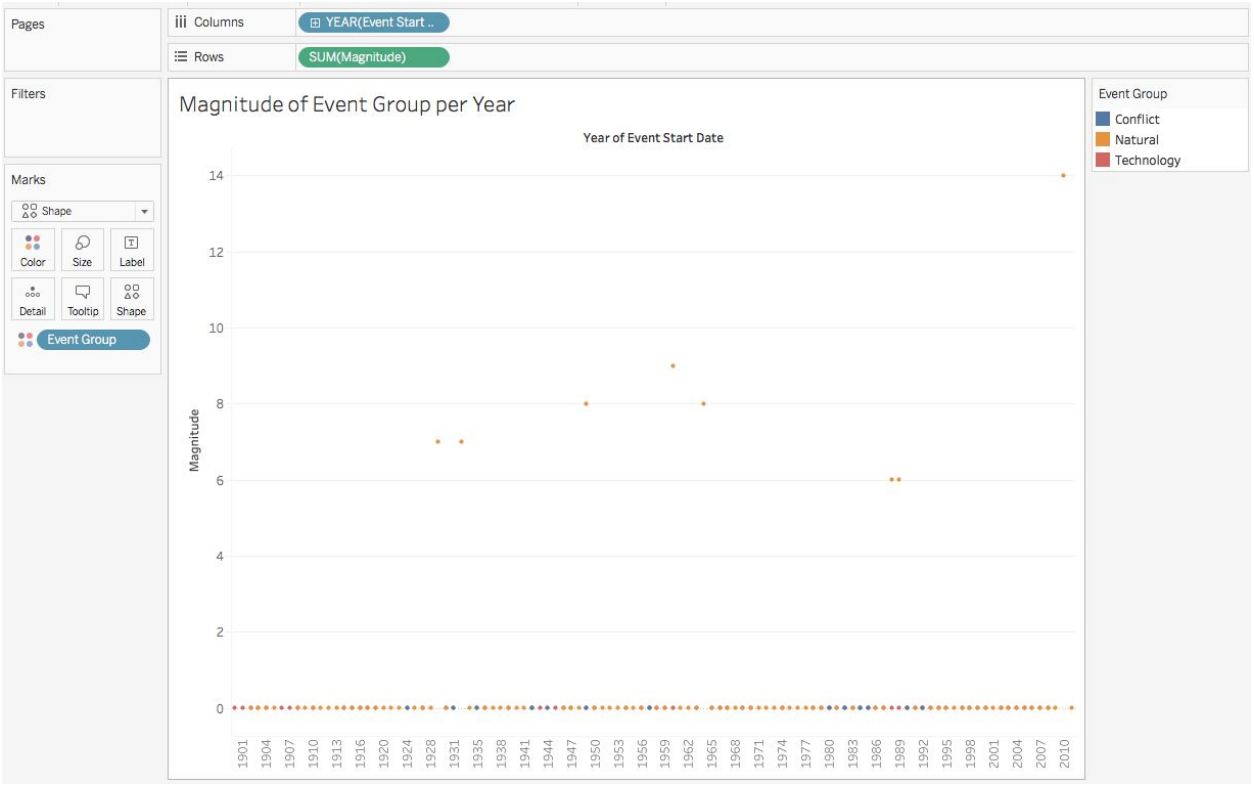



Figure 11: Example of visualizing the magnitude of event groups per year

# References


1. "Meteorological Forecast - Open Government - Canada.ca." https://open.canada.ca/data/en/dataset/4a5ff290-e27d-42f3-9b34-380599546120. Accessed 22 Mar. 2018.
2. "Historical Climate Data." http://climate.weather.gc.ca/. Accessed 22 Mar. 2018.
3. "Population and dwelling counts, for Canada and census subdivisions ...." http://www12.statcan.gc.ca/census-recensement/2011/dp-pd/hlt-fst/pd-pl/Table-Tableau.cfm?LANG=Eng&T=301&S=3&O=D. Accessed 24 Mar. 2018.
4. "NLTK's list of english stopwords - GitHub Gist." https://gist.github.com/sebleier/554280. Accessed 25 Mar. 2018.
5. "The Canadian Disaster Database." 15 Dec. 2015, https://www.publicsafety.gc.ca/cnt/rsrcs/cndn-dsstr-dtbs/index-en.aspx. Accessed 20 Mar. 2018.