Capstone Milestone Report

Ivy Chiang

# Predicting Network fault severity

### Introduction

In the telephony network, telephone exchanges and nodes are connected by wire, microwaves or via satellite to provide communication service for people to communicate across long distances. However service disruption can arise from physical damage and network congestion from increased call traffic. When these disruptions occur, the matter could be life and death in emergency situations or delays and losses for businesses. So to ensure network availability to service customers, Telstra is constantly monitoring the network to ensure the network is available to service customers for their communication needs.

Since these disruptions are done by nature or human activity, their forces can at times can be hard to predict but is acknowledged and managed as part of the system. When there a disruption is present, a log of the error is recorded and alternative paths is applied to cater for repairs. Some faults will only disservice customers from a decrease service capability whilst other faults can be a complete shutdown that will need immediate attention.Thus in a business point of view the dilemma is how to allocate resources effectively to reduce repair costs and proactively fix faults before they occur and avoid unnecessary costs.

### Dataset

Data has been sourced with Telstra's kaggle competition at <https://www.kaggle.com/c/telstra-recruiting-network>. There are 6 data tables about the following:

\*train.csv - the training set for fault severity

\*test.csv - the test set for fault severity

\*event\_type.csv - event type related to the main dataset

\*log\_feature.csv - features extracted from log files

\*resource\_type.csv - type of resource related to the main dataset

\*severity\_type.csv - severity type of a warning message coming from the log

#### Important fields include:

: train fault\_severity - discrete | id | location | fault\_severity | |-----:|-----------:|-----------------:|

: test | id | location | |-----:|-----------:|

: log\_feature | id | log\_feature | volume | |-----:|--------------:|:--------:|

: resource\_type - discrete | id | resource\_type | |-----:|----------------:|

: event\_type - discrete | id | event\_type | |-----:|-------------:|

: severity\_type - discrete | id | severity\_type | |-----:|----------------:|

#### Limitation

From this data, the description of the log features is unknown, also the resource type is assumed to be a category due to the repeated use of values

#### Data Cleaning and wrangling

In this data, most data did not require tidying because all tables presented columns of categorised variables and not data. Also the data itself is relatively clean, where most values are discrete values. However in loading from the log\_feature dataset, all values were joined as one variable, so these values have been extracted into 3 columns (id, log\_feature & volume ) using separate().

During the loading of data, all data was extracted unfactorised so as for manual conversion for better understanding and control of data types. All discrete variables are also converted to numbers so as to ease analysis as numbers.

Then, data is ordered by id to join tables together as one set to uncover and remove missing data ready for analysis.

#loading csv files into dataframes named - train, logfeat, resourcetype, eventtype, sevtype, test  
train <-  
 read.csv("~/Foundations/DataSources/Telstra/train.csv", stringsAsFactors = FALSE ) %>% tbl\_df  
  
log\_feature <-  
 read.csv2("~/Foundations/DataSources/Telstra/log\_feature.csv", stringsAsFactors = FALSE) %>% tbl\_df  
  
resource\_type <-  
 read.csv("~/Foundations/DataSources/Telstra/resource\_type.csv", stringsAsFactors = FALSE) %>% tbl\_df  
  
event\_type <-  
 read.csv("~/Foundations/DataSources/Telstra/event\_type.csv", stringsAsFactors = FALSE) %>% tbl\_df  
  
sev\_type <-  
 read.csv("~/Foundations/DataSources/Telstra/severity\_type.csv", stringsAsFactors = FALSE ) %>% tbl\_df  
  
test <-  
 read.csv("~/Foundations/DataSources/Telstra/test.csv", stringsAsFactors = FALSE ) %>% tbl\_df

##### Tidy log\_feature

str(log\_feature)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 58671 obs. of 1 variable:  
## $ id.log\_feature.volume: chr "6597,feature 68,6" "8011,feature 68,7" "2597,feature 68,1" "5022,feature 172,2" ...

log\_feature <-   
 log\_feature %>%   
 separate(id.log\_feature.volume , c("id", "log\_feature", "volume"), sep = ",")

##### Clean types - log\_feature

str(log\_feature) # view Structure

## Classes 'tbl\_df', 'tbl' and 'data.frame': 58671 obs. of 3 variables:  
## $ id : chr "6597" "8011" "2597" "5022" ...  
## $ log\_feature: chr "feature 68" "feature 68" "feature 68" "feature 172" ...  
## $ volume : chr "6" "7" "1" "2" ...

summary(log\_feature) # Check for missing values

## id log\_feature volume   
## Length:58671 Length:58671 Length:58671   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character

log\_feature$log\_feature <- sub("feature ", "", log\_feature$log\_feature)  
  
log\_feature$id <- as.integer(log\_feature$id)  
log\_feature$volume <- as.integer(log\_feature$volume)  
log\_feature$log\_feature <- as.integer(log\_feature$log\_feature)  
  
head(log\_feature)

## # A tibble: 6 × 3  
## id log\_feature volume  
## <int> <int> <int>  
## 1 6597 68 6  
## 2 8011 68 7  
## 3 2597 68 1  
## 4 5022 172 2  
## 5 5022 56 1  
## 6 5022 193 4

##### Clean types - event\_type

str(event\_type)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 31170 obs. of 2 variables:  
## $ id : int 6597 8011 2597 5022 5022 6852 6852 5611 14838 14838 ...  
## $ event\_type: chr "event\_type 11" "event\_type 15" "event\_type 15" "event\_type 15" ...

summary(log\_feature)

## id log\_feature volume   
## Min. : 1 Min. : 1.0 Min. : 1.000   
## 1st Qu.: 4658 1st Qu.:134.0 1st Qu.: 1.000   
## Median : 9275 Median :227.0 Median : 2.000   
## Mean : 9271 Mean :209.1 Mean : 9.685   
## 3rd Qu.:13903 3rd Qu.:307.0 3rd Qu.: 7.000   
## Max. :18552 Max. :386.0 Max. :1310.000

event\_type$event\_type <- sub("event\_type ", "", event\_type$event\_type)  
event\_type$event\_type <- as.integer(event\_type$event\_type)  
  
head(event\_type)

## # A tibble: 6 × 2  
## id event\_type  
## <int> <int>  
## 1 6597 11  
## 2 8011 15  
## 3 2597 15  
## 4 5022 15  
## 5 5022 11  
## 6 6852 11

##### Clean types - resource\_type

str(resource\_type)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 21076 obs. of 2 variables:  
## $ id : int 6597 8011 2597 5022 6852 5611 14838 2588 4848 6914 ...  
## $ resource\_type: chr "resource\_type 8" "resource\_type 8" "resource\_type 8" "resource\_type 8" ...

resource\_type$resource\_type <- sub("resource\_type ","", resource\_type$resource\_type)  
resource\_type$resource\_type <- as.integer(resource\_type$resource\_type)  
  
head(resource\_type)

## # A tibble: 6 × 2  
## id resource\_type  
## <int> <int>  
## 1 6597 8  
## 2 8011 8  
## 3 2597 8  
## 4 5022 8  
## 5 6852 8  
## 6 5611 8

##### Clean types - sev\_type

str(sev\_type)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 18552 obs. of 2 variables:  
## $ id : int 6597 8011 2597 5022 6852 5611 14838 2588 4848 6914 ...  
## $ severity\_type: chr "severity\_type 2" "severity\_type 2" "severity\_type 2" "severity\_type 1" ...

sev\_type$severity\_type <- sub("severity\_type ", "", sev\_type$severity\_type)  
sev\_type$severity\_type <- as.integer(sev\_type$severity\_type)  
  
head(sev\_type)

## # A tibble: 6 × 2  
## id severity\_type  
## <int> <int>  
## 1 6597 2  
## 2 8011 2  
## 3 2597 2  
## 4 5022 1  
## 5 6852 1  
## 6 5611 2

##### Clean types - train

str(train)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 7381 obs. of 3 variables:  
## $ id : int 14121 9320 14394 8218 14804 1080 9731 15505 3443 13300 ...  
## $ location : chr "location 118" "location 91" "location 152" "location 931" ...  
## $ fault\_severity: int 1 0 1 1 0 0 0 0 1 1 ...

train$location <- sub("location ","",train$location)  
  
train$location <- as.integer(train$location)  
train$fault\_severity <- as.integer(train$fault\_severity)  
  
head(train)

## # A tibble: 6 × 3  
## id location fault\_severity  
## <int> <int> <int>  
## 1 14121 118 1  
## 2 9320 91 0  
## 3 14394 152 1  
## 4 8218 931 1  
## 5 14804 120 0  
## 6 1080 664 0

##### Clean types - test

str(test)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 11171 obs. of 2 variables:  
## $ id : int 11066 18000 16964 4795 3392 3795 2881 1903 5245 6726 ...  
## $ location: chr "location 481" "location 962" "location 491" "location 532" ...

test$location <- sub("location ","",test$location)  
test$location <- as.integer(test$location)  
  
head(test)

## # A tibble: 6 × 2  
## id location  
## <int> <int>  
## 1 11066 481  
## 2 18000 962  
## 3 16964 491  
## 4 4795 532  
## 5 3392 600  
## 6 3795 794

Order tables to prepare Joining

arrange(train, id)

## # A tibble: 7,381 × 3  
## id location fault\_severity  
## <int> <int> <int>  
## 1 1 601 1  
## 2 5 460 0  
## 3 6 332 1  
## 4 8 243 0  
## 5 13 418 0  
## 6 19 644 1  
## 7 20 79 0  
## 8 23 257 0  
## 9 24 367 0  
## 10 26 238 0  
## # ... with 7,371 more rows

arrange(log\_feature, id)

## # A tibble: 58,671 × 3  
## id log\_feature volume  
## <int> <int> <int>  
## 1 1 68 2  
## 2 1 345 2  
## 3 1 179 1  
## 4 2 315 1  
## 5 2 312 1  
## 6 2 235 1  
## 7 2 233 1  
## 8 2 313 1  
## 9 3 171 2  
## 10 4 370 3  
## # ... with 58,661 more rows

arrange(resource\_type, id)

## # A tibble: 21,076 × 2  
## id resource\_type  
## <int> <int>  
## 1 1 8  
## 2 1 6  
## 3 2 2  
## 4 3 8  
## 5 4 2  
## 6 5 2  
## 7 6 2  
## 8 7 2  
## 9 8 2  
## 10 9 8  
## # ... with 21,066 more rows

arrange(event\_type, id)

## # A tibble: 31,170 × 2  
## id event\_type  
## <int> <int>  
## 1 1 11  
## 2 1 13  
## 3 2 35  
## 4 2 34  
## 5 3 11  
## 6 4 47  
## 7 5 34  
## 8 5 35  
## 9 6 34  
## 10 7 44  
## # ... with 31,160 more rows

arrange(sev\_type, id)

## # A tibble: 18,552 × 2  
## id severity\_type  
## <int> <int>  
## 1 1 1  
## 2 2 2  
## 3 3 1  
## 4 4 4  
## 5 5 2  
## 6 6 2  
## 7 7 1  
## 8 8 2  
## 9 9 1  
## 10 10 1  
## # ... with 18,542 more rows

arrange(test, id)

## # A tibble: 11,171 × 2  
## id location  
## <int> <int>  
## 1 2 474  
## 2 3 64  
## 3 4 645  
## 4 7 638  
## 5 9 1100  
## 6 10 878  
## 7 11 398  
## 8 12 899  
## 9 14 1009  
## 10 15 159  
## # ... with 11,161 more rows

Full join

network <- full\_join(train, log\_feature, by = c("id"="id"))  
network <- full\_join(network, resource\_type, by = c("id"="id"))  
network <- full\_join(network, event\_type, by = c("id"="id"))  
network <- full\_join(network, sev\_type, by = c("id"="id"))  
  
head(network)

## # A tibble: 6 × 8  
## id location fault\_severity log\_feature volume resource\_type  
## <int> <int> <int> <int> <int> <int>  
## 1 14121 118 1 312 19 2  
## 2 14121 118 1 312 19 2  
## 3 14121 118 1 232 19 2  
## 4 14121 118 1 232 19 2  
## 5 9320 91 0 315 200 2  
## 6 9320 91 0 315 200 2  
## # ... with 2 more variables: event\_type <int>, severity\_type <int>

summary(network) # There are some logs with no location details

## id location fault\_severity log\_feature   
## Min. : 1 Min. : 1.0 Min. :0.00 Min. : 1.0   
## 1st Qu.: 4467 1st Qu.: 304.0 1st Qu.:0.00 1st Qu.:134.0   
## Median : 9165 Median : 607.0 Median :0.00 Median :227.0   
## Mean : 9150 Mean : 580.2 Mean :0.55 Mean :211.4   
## 3rd Qu.:13824 3rd Qu.: 834.0 3rd Qu.:1.00 3rd Qu.:306.0   
## Max. :18552 Max. :1126.0 Max. :2.00 Max. :386.0   
## NA's :84584 NA's :84584   
## volume resource\_type event\_type severity\_type   
## Min. : 1.000 Min. : 1.000 Min. : 1.00 Min. :1.000   
## 1st Qu.: 1.000 1st Qu.: 2.000 1st Qu.:13.00 1st Qu.:1.000   
## Median : 2.000 Median : 2.000 Median :23.00 Median :1.000   
## Mean : 8.341 Mean : 4.566 Mean :25.15 Mean :1.451   
## 3rd Qu.: 6.000 3rd Qu.: 8.000 3rd Qu.:35.00 3rd Qu.:2.000   
## Max. :1310.000 Max. :10.000 Max. :54.00 Max. :5.000   
##

str(network)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 146423 obs. of 8 variables:  
## $ id : int 14121 14121 14121 14121 9320 9320 9320 9320 14394 14394 ...  
## $ location : int 118 118 118 118 91 91 91 91 152 152 ...  
## $ fault\_severity: int 1 1 1 1 0 0 0 0 1 1 ...  
## $ log\_feature : int 312 312 232 232 315 315 235 235 221 221 ...  
## $ volume : int 19 19 19 19 200 200 116 116 1 1 ...  
## $ resource\_type : int 2 2 2 2 2 2 2 2 2 2 ...  
## $ event\_type : int 34 35 34 35 34 35 34 35 35 34 ...  
## $ severity\_type : int 2 2 2 2 2 2 2 2 2 2 ...

#view na entries  
network %>% filter(is.na(fault\_severity)) %>% head(10)

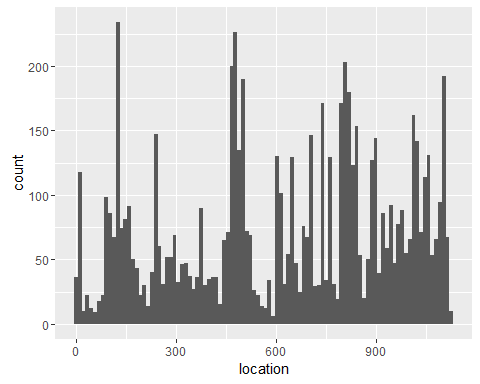
## # A tibble: 10 × 8  
## id location fault\_severity log\_feature volume resource\_type  
## <int> <int> <int> <int> <int> <int>  
## 1 6597 NA NA 68 6 8  
## 2 2597 NA NA 68 1 8  
## 3 5022 NA NA 172 2 8  
## 4 5022 NA NA 172 2 8  
## 5 5022 NA NA 56 1 8  
## 6 5022 NA NA 56 1 8  
## 7 5022 NA NA 193 4 8  
## 8 5022 NA NA 193 4 8  
## 9 5022 NA NA 71 3 8  
## 10 5022 NA NA 71 3 8  
## # ... with 2 more variables: event\_type <int>, severity\_type <int>

### Preliminary exploration

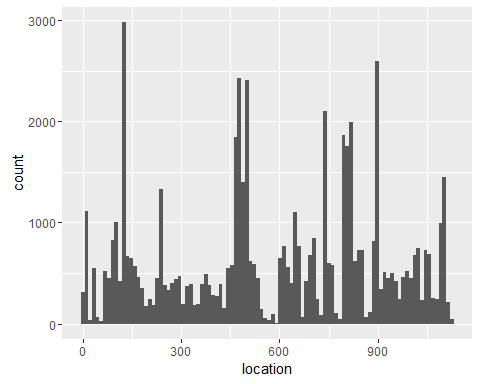
Firstly, let's examine each dataset

##### train

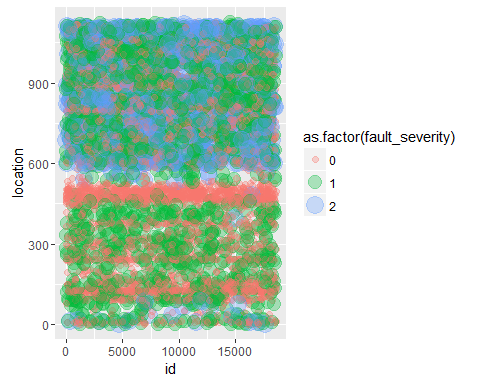
## frequency at each location  
ggplot(train, aes(x = location)) + geom\_histogram(bins = 100)



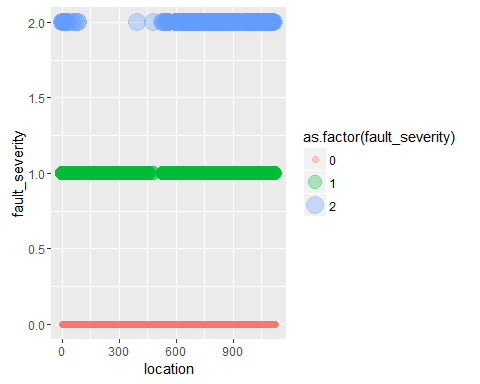
ggplot(network, aes(x = location)) + geom\_histogram(bins = 100)



## Removed 84584 rows containing non-finite values (stat\_bin).  
##-Conclusion: some ids from joining other tables together have missing logs.  
##-Some locations have more frequent fault reporting (any correlation with resource type?)  
  
## relationship of location with id  
ggplot(train, aes(x = id, y = location, col = as.factor(fault\_severity), size = as.factor(fault\_severity) )) + geom\_point(alpha = 0.3)

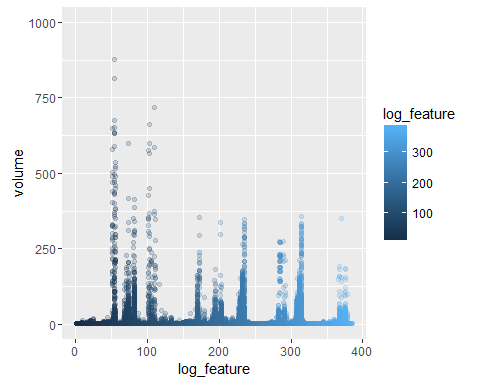


##-There is a pattern of faults to have the same severity type hence the linear repetition across the id scale  
  
##ggplot(train, aes(x = id, y = location)) + geom\_point() + facet\_grid( . ~ faulty\_severity)  
##Error in layout\_base(data, cols, drop = drop) :   
## At least one layer must contain all variables used for facetting  
  
##relationship of feature with fault\_severity  
ggplot(train, aes(x = location, y = fault\_severity, col = as.factor(fault\_severity), size = as.factor(fault\_severity) )) + geom\_point(alpha = 0.3)

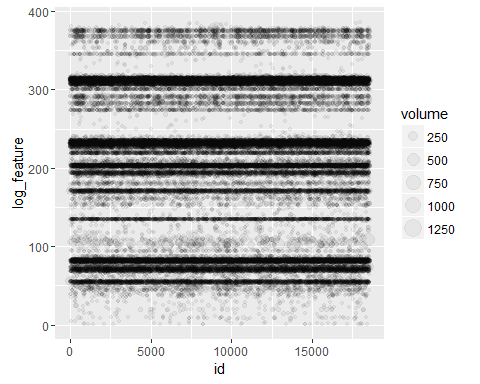


##### log\_feature

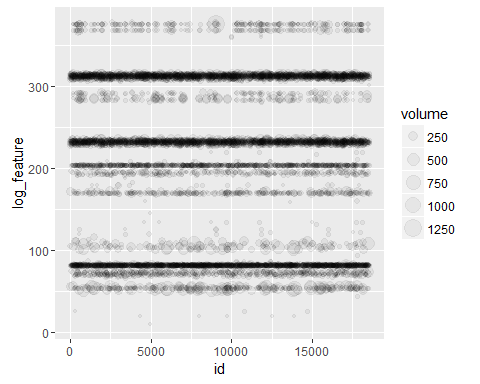
## frequency at each location  
ggplot(log\_feature, aes(x = log\_feature, y = volume, col = log\_feature)) + geom\_point(alpha = 0.2) + coord\_cartesian(ylim = c(0, 1000)) # removed outlier



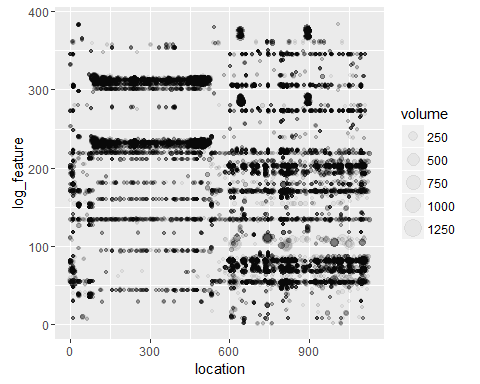
##-The Volume variable here is a continuous measure with some log\_feature more frequent than others  
  
## relationship of log\_feature with id  
ggplot(log\_feature, aes(x = id, y = log\_feature, size = volume)) + geom\_point(alpha = 0.05)



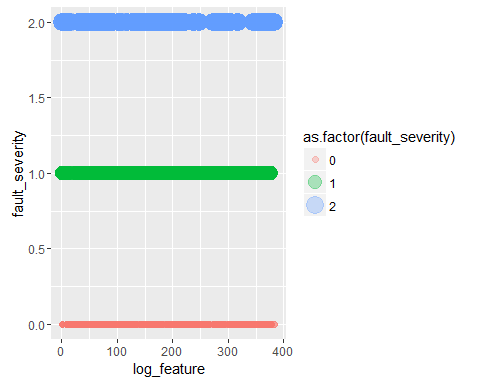
#-There are no missing values in this dataset, but reveals some common error log\_features logged regularly  
  
## Volume of the faults   
volume\_above10 <- log\_feature %>% filter(volume > 10)  
ggplot(volume\_above10, aes(x = id, y = log\_feature, size = volume)) + geom\_point(alpha = 0.05)



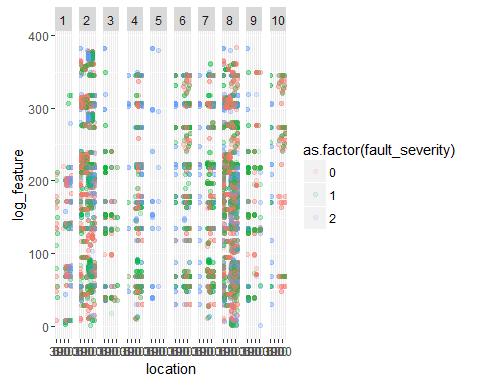
##-The Volume depicted is fairly uniform to the type of feature the fault being recorded with, showing a uniform match with the fault feature recorded  
  
##-Question is of there is a relation of these log\_features with location.  
ggplot(network, aes(x = location, y = log\_feature, size = volume)) + geom\_jitter(alpha = 0.05)



##-Log features appear distinctly at grouped location numbers, showing where the features regularly appear like a map  
  
  
##relationship of log\_feature with fault\_severity  
ggplot(network, aes(x = log\_feature, y = fault\_severity, col = as.factor(fault\_severity), size = as.factor(fault\_severity) )) + geom\_point(alpha = 0.3)

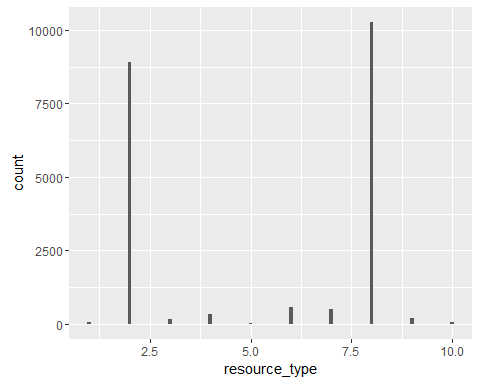


##relationship of log\_feature X location with fault\_severity  
ggplot(network, aes(x = location, y = log\_feature, col = as.factor(fault\_severity))) + geom\_point(alpha = 0.1) + facet\_grid( . ~ resource\_type)

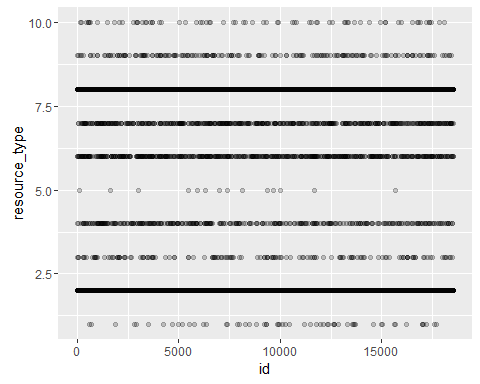


##### resource\_type

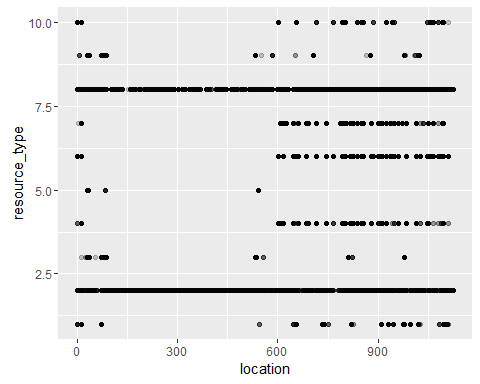
## frequency at each location  
ggplot(resource\_type, aes(x= resource\_type)) + geom\_histogram(bins = 100)



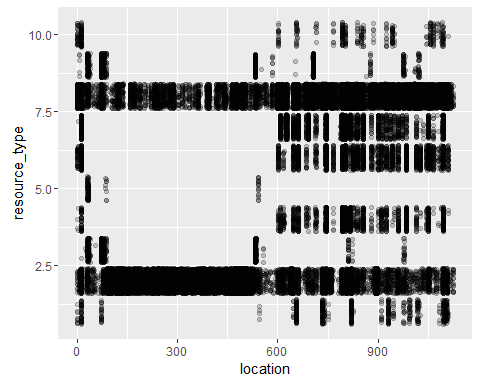
##There are 10 distinct resource types  
  
## relationship of resource\_type with id  
ggplot(resource\_type, aes(x = id, y = resource\_type)) + geom\_point(alpha = 0.2)



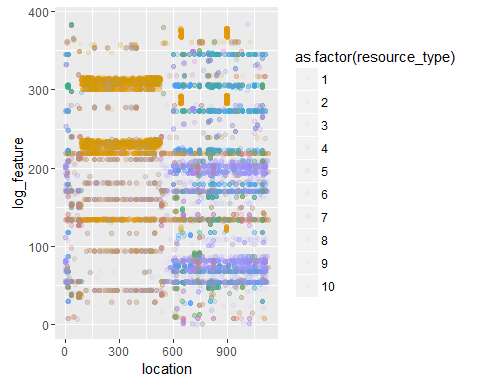
##-Shows some resource types are more likely to be logged more frequently  
  
## relationship of resource\_type with location  
ggplot(network, aes(x = location, y = resource\_type)) + geom\_point(alpha = 0.2)



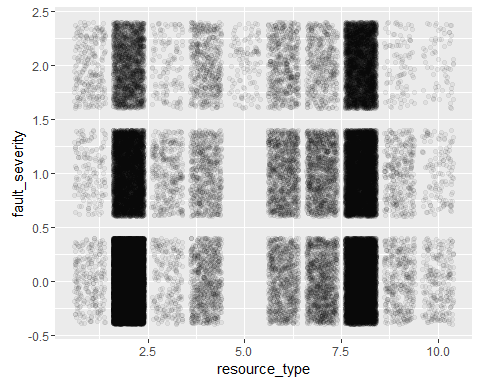
ggplot(network, aes(x = location, y = resource\_type)) + geom\_jitter(alpha = 0.2)



##-In comparing the locations, there is clustering on certain resource type by location number  
##-As the dataset records errors, it is unclear whether all locations have all resource types   
##-but some location's equipment may be more prone to fault then others, this may be due to the  
##-location's position in a high traffic hub or that there particular resource is due for replacement  
  
## relationship of resource\_type with log\_feature, as the resource type is a discrete variable  
ggplot(network, aes(x = location, y = log\_feature, col= as.factor(resource\_type)) ) + geom\_jitter(alpha = 0.05)

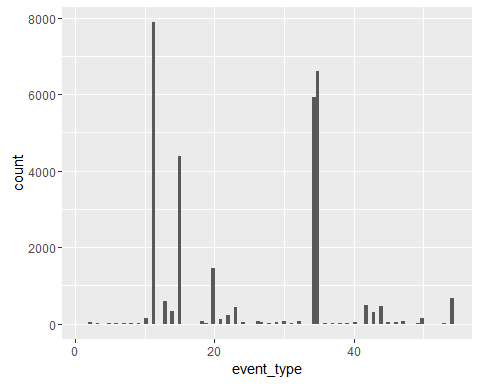


##-Here we can tell that by location, error features logged correspond to certain resource type   
##-at that location, with each resource having distinct log features.  
  
##-Question here will be is there any correlation of the resource type, log feature to   
##-severity\_type to produce the level of fault\_severity.  
##-In plain english this maybe the pinpoint of a particular machine that location that   
##-works as a main artery of the network to publish repair urgency.  
  
##- "location" > "resource" > "log\_feature" > "Volume" ~ fault occurance  
  
## compare resource\_type with fault\_severity  
ggplot(network, aes(x = resource\_type, y = fault\_severity)) + geom\_jitter(alpha = 0.05)

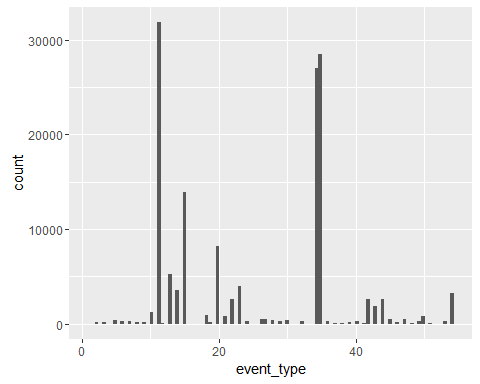


##### event\_type

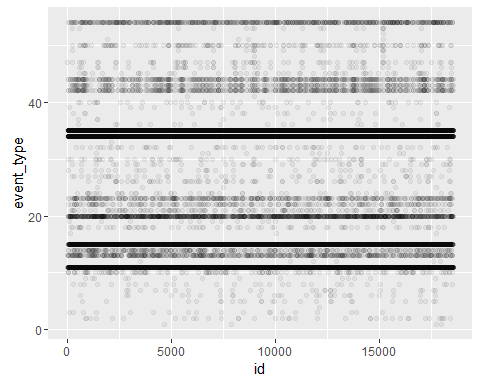
## frequency at each location  
ggplot(event\_type, aes(event\_type)) + geom\_histogram(bins = 100)



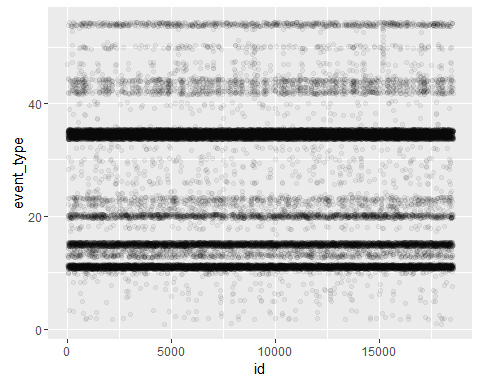
ggplot(network, aes(x = event\_type)) + geom\_histogram(bins = 100) # There are no missing values



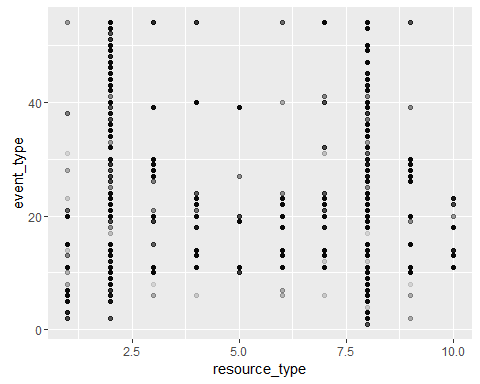
##-There appears to be distinct but also popular events.  
  
##-Question is, do these events correlation with particular resource (machine) that can   
##-interpret level of fault severity?  
  
## relationship of event\_type with id  
ggplot(event\_type, aes(x = id, y = event\_type)) + geom\_point(alpha = 0.05)



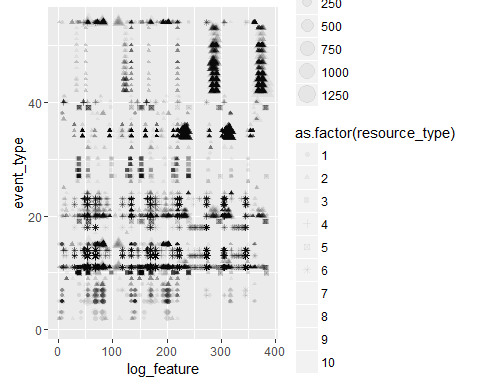
ggplot(event\_type, aes(x = id, y = event\_type)) + geom\_jitter(alpha = 0.05)



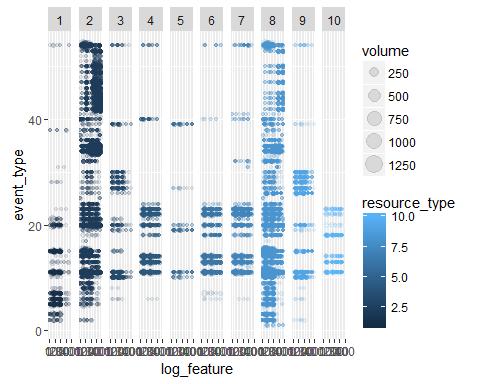
##-There are certain event types that are frequently occuring or being recorded. Reviewing with jitter does not show much gaps in frequency  
  
## relationship of event\_type with resource\_type  
ggplot(network, aes(x = resource\_type, y = event\_type)) + geom\_point(alpha = 0.05)



## Some resource share the same event\_type but the plot further shows that resource\_type is a discrete variable of range 10.  
  
## relationship of event\_type with log\_feature where Volume is displayed differentiated by resource\_type  
ggplot(network, aes(x = log\_feature, y = event\_type, size = volume, shape = as.factor(resource\_type))) + geom\_point(alpha = 0.05)



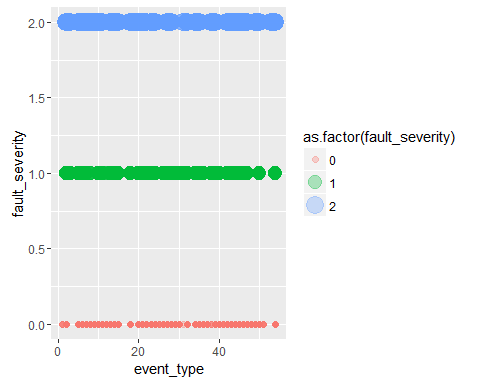
## The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes  
## difficult to discriminate; you have 10. Consider specifying shapes manually if you must have  
## them.  
  
##-There are some event types carried over to multiple log\_feature, with various event types   
##-have varying degrees of log\_feature volumes, hence there may not be any correction there,   
##-certain fault events are more highly probable with particular resources as evidence of clusters  
  
##- At this point, volume in reference with log\_feature may be telling of the quantity of equipment  
##- and not severity  
  
## further look at relationship of event\_type with log\_feature, sized in Volume, faceted by resource\_type  
ggplot(network, aes(x = log\_feature, y = event\_type, size = volume, col = resource\_type)) + geom\_point(alpha = 0.1) + facet\_grid( . ~ resource\_type )



##-Resources 4 & 6 have very similar patterns  
  
## relationship of event\_type with location  
ggplot(network, aes(x = location, y = event\_type, col = as.factor(resource\_type))) + geom\_point(alpha = 0.05)

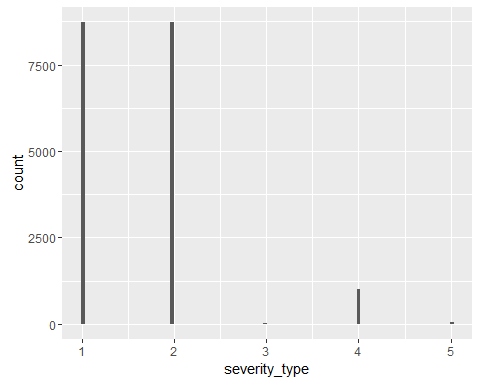


##-There is a similarity of pattern with "locationXresource\_type" and "locationXlog\_feature".  
  
##relationship of event\_type with fault\_severity  
ggplot(network, aes(x = event\_type, y = fault\_severity, col = as.factor(fault\_severity), size = as.factor(fault\_severity) )) + geom\_point(alpha = 0.3)

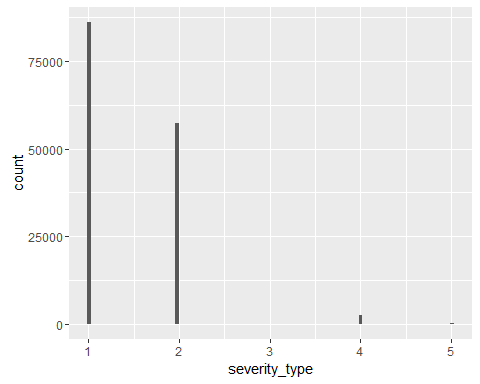


###### severity\_type

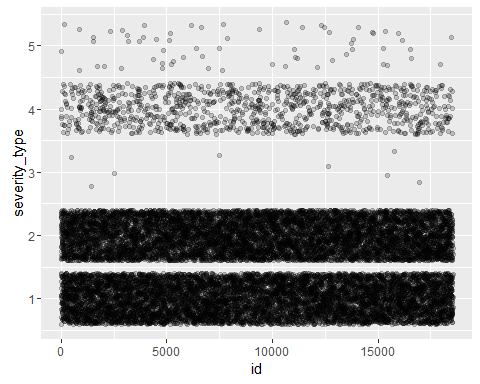
## frequency at each location  
ggplot(sev\_type, aes(severity\_type)) + geom\_histogram(bins = 100)



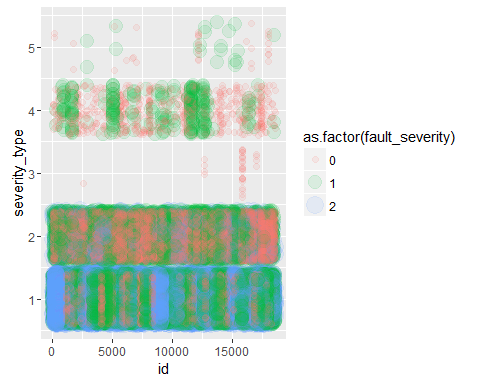
ggplot(network, aes(x = severity\_type)) + geom\_histogram(bins = 100) # There are no missing values



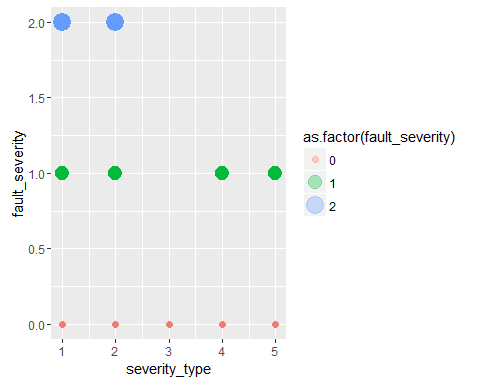
##-There appears to be 5 distinct types.  
  
## relationship of feature with id  
ggplot(sev\_type, aes(x = id, y = severity\_type)) + geom\_jitter(alpha = 0.2)



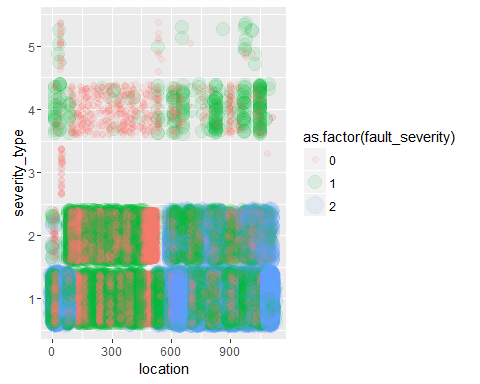
## The distinct categories have clear proportion clear distributed in the logs  
  
## relationship of feature with id highlighted by fault\_severity  
ggplot(network, aes(x = id, y = severity\_type, col = as.factor(fault\_severity), size = as.factor(fault\_severity))) + geom\_jitter(alpha = 0.1)



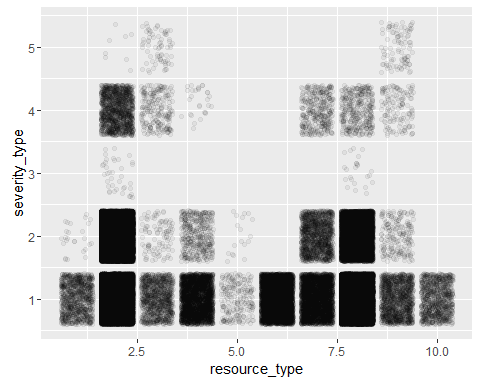
##-Most severe faults are categorized by severity\_type no.1 .  
##-Severity\_type no.3 only attracts fault level of 1 .  
  
##relationship of severity\_type with fault\_severity  
ggplot(network, aes(x = severity\_type, y = fault\_severity, col = as.factor(fault\_severity), size = as.factor(fault\_severity) )) + geom\_point(alpha = 0.3)



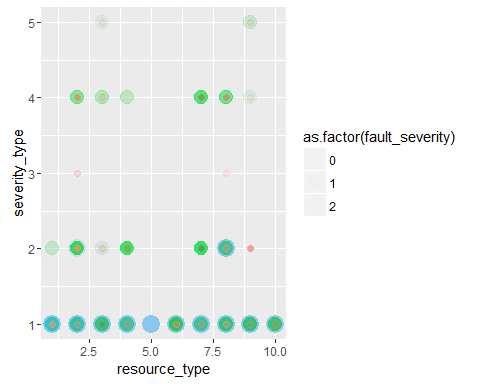
##-Severity\_type no.3 only attracts fault level of 1 confirmed.  
  
##relationship of feature with location highlighted with fault\_severity  
ggplot(network, aes(x = location, y = severity\_type, col = as.factor(fault\_severity), size = as.factor(fault\_severity))) + geom\_jitter(alpha = 0.1)



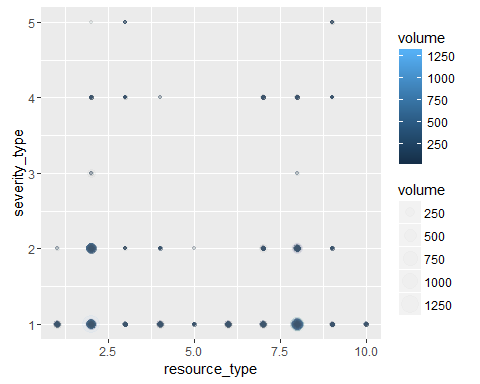
##-High levels of faults appear at low number locations or locations numbered greater than 500  
  
#relationship of feature with resource\_type  
ggplot(network, aes(x = resource\_type, y = severity\_type)) + geom\_jitter(alpha = 0.05)



##relationship of feature with resource\_type with fault\_severity highlights  
ggplot(network, aes(x = resource\_type, y = severity\_type, col = as.factor(fault\_severity), size = as.factor(fault\_severity))) + geom\_point(alpha = 0.01)

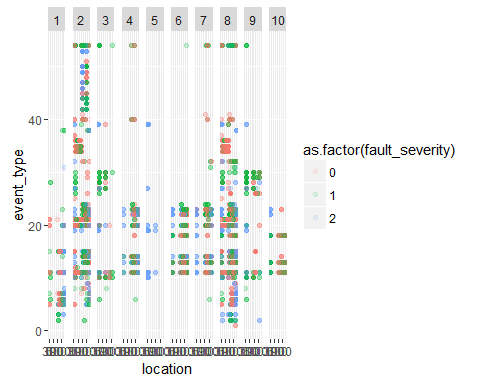


##-all resources can display the first severity type with high fault level  
  
## compare faulty severity highlights with volume  
ggplot(network, aes(x = resource\_type, y = severity\_type, col = volume, size = volume)) + geom\_point(alpha = 0.01)

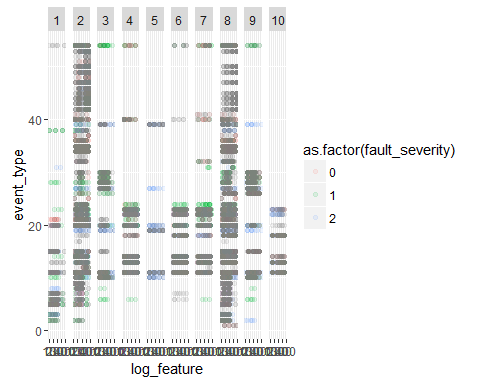


##### Other exploration with fault\_severity

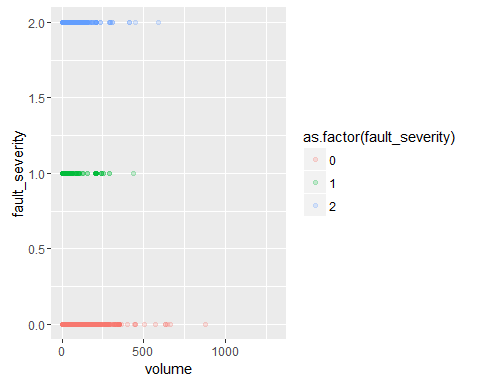
##relationship of event\_type X location with fault\_severity  
ggplot(network, aes(x = location, y = event\_type, col = as.factor(fault\_severity))) + geom\_point(alpha = 0.1) + facet\_grid( . ~ resource\_type)



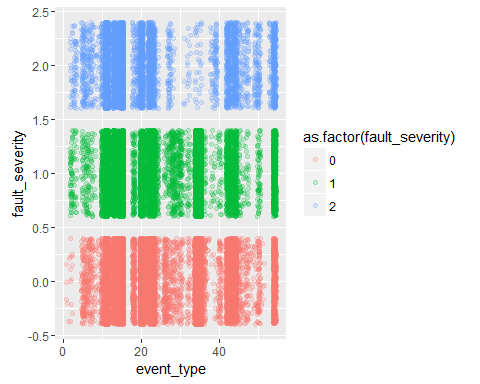
##relationship of event\_type X log\_feature with fault\_severity  
ggplot(network, aes(x = log\_feature, y = event\_type, col = as.factor(fault\_severity))) + geom\_point(alpha = 0.1) + facet\_grid( . ~ resource\_type)



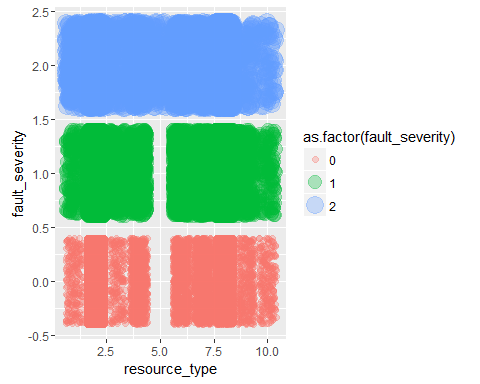
## compare volume with fault\_severity  
ggplot(network, aes(x = volume, y = fault\_severity, col = as.factor(fault\_severity))) + geom\_point(alpha = 0.2)



## compare event\_type with fault\_severity  
ggplot(network, aes(x = event\_type, y = fault\_severity,col = as.factor(fault\_severity))) + geom\_jitter(alpha = 0.2)



## compare resource\_type with fault\_severity  
ggplot(network, aes(x = resource\_type, y = fault\_severity, col = as.factor(fault\_severity), size = as.factor(fault\_severity))) + geom\_jitter(alpha = 0.3)



##-Useful to interpret end prediction outcomes

### Initial Findings

The fault\_severity and volume reported seems to be highly correlated and looks to be descriptors from resource’s fault event.

This is because each resource type seems to adopt with particular fault events and these faulty events are also sprung from particular log\_features. Thus the data tells us which log is has certain circumstances and issues develop fault.

Since each resource\_type has particular locations it often logs issues with, it is under these certain events that a prediction can be made regarding fault\_severity. There are 4 distinct characters arising from the log features where each has different volume demands at different events. Some resource types require regular attention yet the has little impact on the network.

Meanwhile severity\_types are selective depending on the type of resource, with types 1 & 2 yielding higher probability to highly network disruption.

The trend found is that towards fault\_severity, volume and severity is deterministic to it. Meanwhile with location, particular log\_features describe resource\_type

To determine fault\_severity = 2 the following events gives a high chance of outcome: 1. Location > resource\_type >log\_feature / volume > severity\_type (1 or 2) or 2. Resource\_type = 5

### Approach to the problem

From this analysis, different combinations will present different fault\_severity predictions. The data is highly structural with many discrete variables.

The approach to this problem is to determine the different event combinations and work out the probability to predict the fault\_severity. By category and walking through the tree to obtain an estimate.

Steps 1. Define the resource types logged at each location. 2. Make another table using all datasets but the train table to understand how feature or particular event type can derive which resource is at fault thus if able to reveal severity level. This then uses all values with no location log attached. 3. Determine the probability levels of fault\_severity to resource/location to conclude a prediction. 4. Use Random Tree or XGBoost to create the model. 5. Test the model using the test data.