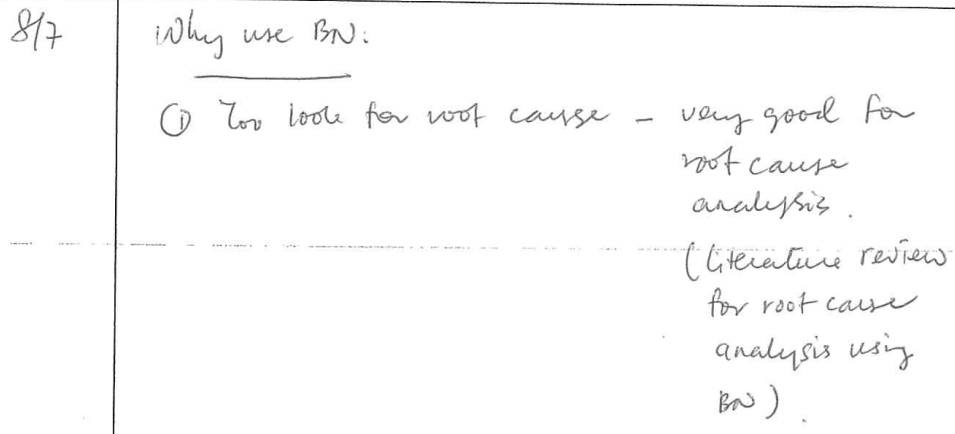
PHD - Progress update 14/10/2015

Fauzy Bin Che Yayah

October 14, 2015

* Why use Bayesian Net ?

1. To look for the root cause - the dataset is in factor type which is status not a number.
2. To find out the probabilistic relationship between the symptom error code and the resolution



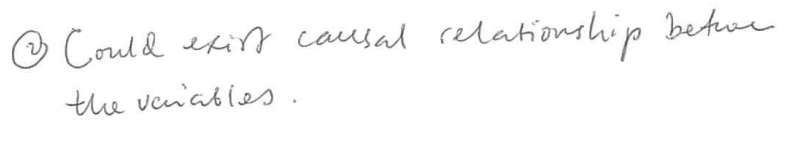
|  |  |  |  |
| --- | --- | --- | --- |
| Citation | Method | Output | Conclusions |
| TroubleMiner: Mining network trouble tickets Medem, A. ; Akodjenou, M.-I ; Teixeira, R. 20091 | Trouble tickets classification | \* Automation process on clustering the free text inside the description of the trouble tickets. \* Choosing the correct keywords for the analysis | Using term frequency distance between trouble tickets and similarity between clusters |
| Knowledge Discovery from Trouble Ticketing Reports in a Large Telecommunication Company Temprado, Y. ; Garcia, C. ; Molinero, F.J. 2009 | Data Mining , Text Mining and Machine Learning , Bayes Net, Naïve Bayes | Prediction on the next action of trouble tickets ,Different snapshots were added to the machine learning algorithm for training | Combination of multiple method to construct the recommendation , Using Bayesian for prediction |
| A Bayesian Approach To Stochastic Root Finding 2011 | x | x | x |
| A Fully Bayesian Approach For Unit Root Testing 2011 | x | x | x |
| Online Root-Cause Analysis Of Alarms In Discrete Bayesian 2014 | x | x | x |
| Documents Categorization Based On Bayesian Spanning Tree 2006 | x | x | x |
| Benefits of a Bayesian Approach to Anomaly and Failure 2009 | x | x | x |

List of literature review regarding Bayesian Net :-

1.A real-life application of multi-agent systems for fault diagnosis in the provision of an Internet business service

2.A Bayesian Network approach to diagnosing the root cause of failure

3.sss



# Process on gathering the dataset

* Acquiring dataset for 100 records, for each zone , randomize , selective year ; ie . 2015
* Rules :-

|  |  |
| --- | --- |
| Rules | Description |
| status = 'Closed' | Dataset must be closed for complete information |
| network\_tt\_id is NULL | Dataset must be not related to Network Trouble Ticket |
| trouble ticket type <> PASSIVE | Trouble Ticket must related to the Active elements such as routers, switches , modem , etc |
| installed\_date is NOT NULL | This field must have value |
| created\_date is NOT NULL | This field must have value |
| closed\_date is NOT NULL | This field must have value |
| closed\_date is NOT NULL | This field must have value |
| product is NOT NULL | This field must have value |
| sub\_product is NOT NULL | This field must have value |
| length description > 10 | This field is useful for text analysis |
| rand() | Record selection is in random mode |
| zone | Should selective from different zone , sparse |

For sample purpose - selecting dataset from ZONE KEPONG for the analysis due to this zone has the highest records inside the Trouble Ticket dataset.

* Using Impala for the data retrieval :-

Documentation - <https://github.com/piersharding/dplyrimpaladb>

* Data processing using DplyrImpalaDb
* Package installation manual below :-

install.packages(c("RJDBC", "devtools", "dplyr"))  
devtools::install\_github("jwills/dplyrimpaladb")  
install.packages("dplyrimpaladb")

* Basic notes why choosing Impala.

1. Cloudera 'Impala', which is a massively parallel processing (MPP) SQL query engine runs natively in Apache Hadoop
2. Impala's Place in the Big Data Ecosystem
3. Flexibility for Big Data Workflow
4. High-Performance Analytics

# Connection to Impala

Basic Impala drivers can be downloaded from <https://github.com/Mu-Sigma/RImpala/blob/master/impala-jdbc-cdh5.zip>

Below is the components required and how to set the class path for the Impala drivers , RJava , RJDBC and dplyr

suppressWarnings(suppressMessages(library("RJDBC")))  
suppressWarnings(suppressMessages(library("dplyr")))  
suppressWarnings(suppressMessages(library("caret")))  
suppressWarnings(suppressMessages(library("corrplot")))  
suppressWarnings(suppressMessages(library("lazy")))  
suppressWarnings(suppressMessages(library("dplyrimpaladb")))  
suppressWarnings(suppressMessages(library("rpart")))  
  
.jaddClassPath(c(list.files("C:\\Users\\R10154\\Google Drive\\lib\\impala\\new",pattern="jar$",full.names=T)))  
  
.jinit(classpath = c(list.files("C:\\Users\\R10154\\Google Drive\\lib\\impala\\new",pattern="jar$",full.names=T)))  
  
dplyr.jdbc.classpath = c(list.files("C:\\Users\\R10154\\Google Drive\\lib\\impala\\new",pattern="jar$",full.names=T))  
  
conn <- src\_impaladb(dbname='nova', host='10.54.1.151')

## Loading required package: testthat

## Warning: package 'testthat' was built under R version 3.2.1

## [1] "here:"  
## [1] FALSE

* Zone list

result <- tbl(conn, sql("select zone from nova.nova\_trouble\_ticket where zone <> 'null' group by zone order by zone limit 1000"))  
as.data.frame(result)

## zone  
## 1 ZONE AIR ITAM  
## 2 ZONE BANGI  
## 3 ZONE BANGSAR  
## 4 ZONE BANTING  
## 5 ZONE BATU  
## 6 ZONE BATU PAHAT  
## 7 ZONE BAYAN BARU  
## 8 ZONE BINTULU  
## 9 ZONE BUKIT ANGGERIK  
## 10 ZONE BUKIT MERTAJAM  
## 11 ZONE BUKIT RAJA  
## 12 ZONE BUTTERWORTH  
## 13 ZONE CYBERJAYA  
## 14 ZONE GOMBAK  
## 15 ZONE IPOH  
## 16 ZONE KAJANG  
## 17 ZONE KEPONG  
## 18 ZONE KERAMAT  
## 19 ZONE KINRARA  
## 20 ZONE KL CENTRAL  
## 21 ZONE KLANG  
## 22 ZONE KOTA KINABALU SELATAN  
## 23 ZONE KOTA KINABALU UTARA  
## 24 ZONE KUCHING  
## 25 ZONE KULIM  
## 26 ZONE LANGKAWI  
## 27 ZONE MALURI  
## 28 ZONE MELAKA UTARA  
## 29 ZONE MIRI  
## 30 ZONE N. SEMBILAN UTARA  
## 31 ZONE PANDAN  
## 32 ZONE PELANGI  
## 33 ZONE PERLIS  
## 34 ZONE PETALING JAYA  
## 35 ZONE PUCHONG  
## 36 ZONE SEBERANG JAYA  
## 37 ZONE SENAI  
## 38 ZONE SG PETANI  
## 39 ZONE SHAH ALAM  
## 40 ZONE SIBU  
## 41 ZONE SKUDAI PONTIAN  
## 42 ZONE STAMPIN  
## 43 ZONE SUBANG JAYA  
## 44 ZONE TAMAN PETALING  
## 45 ZONE TAMPOI  
## 46 ZONE TAR  
## 47 ZONE TASEK  
## 48 ZONE TASIK AMPANG  
## 49 ZONE TDI  
## 50 ZONE TELUK INTAN  
## 51 ZONE TERENGGANU SELATAN  
## 52 ZONE TERUNTUM

* Trouble Ticket Data Dictionary

result <- tbl(conn, sql("select \* from nova\_trouble\_ticket where zone <> 'null' limit 1"))  
as.data.frame(apply(as.data.frame(result),2,class))

## apply(as.data.frame(result), 2, class)  
## tt\_row\_id character  
## tt\_num character  
## tt\_type character  
## tt\_sub\_type character  
## status character  
## severity character  
## important\_message character  
## appointment\_flag character  
## nova\_account\_name character  
## nova\_subscriber\_num character  
## nova\_account\_num character  
## package\_row\_id character  
## created\_by character  
## category character  
## symptom\_error\_code character  
## priority character  
## product character  
## sub\_product character  
## package\_name character  
## network\_tt\_id character  
## swap\_order\_num character  
## cause\_category character  
## cause\_code character  
## resolution\_code character  
## closure\_category character  
## resolution\_team character  
## service\_affected character  
## service\_order\_num character  
## btu\_type character  
## owner character  
## owner\_name character  
## group\_owner character  
## owner\_position character  
## btu\_platform character  
## dp\_location character  
## created\_date character  
## pending\_verify\_date character  
## closed\_by character  
## closed\_date character  
## source character  
## installed\_date character  
## description character  
## repeat\_ticket\_count character  
## follow\_up\_ticket\_count character  
## fdp\_device\_name character  
## fdp\_site\_name character  
## olt\_site\_name character  
## exchange character  
## timestamp character  
## contact\_id character  
## contact\_name character  
## contact\_office\_phone character  
## contact\_mobile\_phone character  
## contact\_home\_phone character  
## contact\_email\_addr character  
## due\_date character  
## part\_num character  
## network\_layer character  
## network\_row\_id character  
## asset\_id character  
## ptt character  
## zone character  
## service\_point\_id character

# Getting the dataset from Impala

Sample dataset - Selection trouble tickets only from Zone Kepong. The SQL is define by :-

|  |  |
| --- | --- |
| Rules | Description |
| a.status like '%Closed%' | Dataset must be closed for complete information |
| network\_tt\_id = 'null' | Dataset must be not related to Network Trouble Ticket |
| trouble ticket type <> PASSIVE | Trouble Ticket must related to the Active elements such as routers, switches , modem , etc. Excluding for now if related to the 3rd party causes , customer behavior and Passive elements |
| installed\_date is NOT NULL | This field must have value |
| created\_date is NOT NULL | This field must have value |
| closed\_date is NOT NULL | This field must have value |
| closed\_date is NOT NULL | This field must have value |
| product is NOT NULL | This field must have value |
| sub\_product is NOT NULL | This field must have value |
| length description > 10 | This field is useful for text analysis |
| rand() | Record selection is in random mode |
| zone | Should selective from different zone , sparse control |

Generated SQL :-

# result <- tbl(conn, sql("select \* from nova\_trouble\_ticket a join active\_code b on (trim(a.cause\_code) = trim(b.cause\_code)) join exchange\_zone c ON (trim(a.exchange)=trim(c.building\_id)) and (b.code <> 'PASSIVE' ) where c.zone\_name like '%ZONE KEPONG%' and a.status like '%Closed%' and length(a.cause\_category) > 1 and length(a.created\_date) > 6 and length(a.closed\_date) > 6 and length(a.installed\_date) > 6 and a.package\_name not like '%null%' and a.product not like '%null%' and a.sub\_product not like '%null%' and length(a.description) > 10 and network\_tt\_id = 'null' order by rand() limit 10000 "))  
# names(result)

Removing non-related fields such as trouble ticket key , trouble ticket number , trouble ticket date etc.

# result$`tt\_row\_id` <- NULL  
# result$`tt\_num` <- NULL  
# result$`created\_date` <- NULL  
# result$`closed\_date` <- NULL  
# result$`installed\_date` <- NULL  
# num <- as.numeric(result$status)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "status1"  
# num <- as.numeric(result$tt\_sub\_type)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "tt\_sub\_type1"  
# num <- as.numeric(result$category)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "category1"  
# num <- as.numeric(result$symptom\_error\_code)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "symptom\_error\_code1"  
# num <- as.numeric(result$product)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "product1"  
# num <- as.numeric(result$package\_name)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "package\_name1"  
# num <- as.numeric(result$sub\_product)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "sub\_product1"  
# num <- as.numeric(result$cause\_category)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "cause\_category1"  
# num <- as.numeric(result$cause\_code)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "cause\_code1"  
# num <- as.numeric(result$resolution\_code)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "resolution\_code1"  
# num <- as.numeric(result$closure\_category)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "closure\_category1"  
# num <- as.numeric(result$btu\_platform)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "btu\_platform1"  
# num <- as.numeric(result$btu\_type)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "btu\_type1"  
# num <- as.numeric(result$dp\_location)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "dp\_location1"  
# num <- as.numeric(result$zone\_name)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "zone\_name1"  
# num <- as.numeric(result$exchange)-1  
# result <- cbind(result,num)  
# names(result)[names(result)=="num"] <- "exchange1"  
# result <- result[,c(66:81)]  
# head(result)

Remove again the predictors column which might have one unique value which can leads to zero variance result

# result <- result[,-nearZeroVar(result)]   
# head(result)

Find the correlation between the variables using Pearson.

# correlations <- cor(result, use="pairwise.complete.obs", method="pearson")  
# print(correlations)

Find the highest correlated variables.

# highlyCorrelated <- findCorrelation(correlations, 0.5 ,verbose = TRUE,names = TRUE)  
# highlyCorrelated

Summary of the correlated variables.

# summary(correlations)

Plot correlated variables.

# corrplot(correlations, method = "circle",tl.cex = 0.9)

Feature selection process to confirm which variable does become the independent and resolution code is the dependent variable via GBM (Stochastic Gradient Boosting).

List of other available model - <http://topepo.github.io/caret/modelList.html>

# set.seed(999)  
# suppressWarnings(library(mlbench))  
# suppressWarnings(library(caret))  
# control <- trainControl(method="repeatedcv", number=10, repeats=3)  
# model <- train(resolution\_code1~., data=result, method="gbm", preProcess="scale", trControl=control , verbose = FALSE)  
# importance <- varImp(model, scale=TRUE)  
# print(importance)  
# plot(importance)