Text classification - Heathcare project
By Shwet Prakash

Contents:

Problem Statement

To build a classification model based on the Text in the Summary and Description of the call to classify the ticket to appropriate Category (out of 5 Categories) and Subcategories (Out of 20 Sub Categories).

Case Study Explanation and Domain Knowledge

XYZ Health Services is a top ranked Health care provider in USA with stellar credentials and provides high quality-care with focus on end-to-end Health care services. The Heath Care Services range from basic medical diagnostics to critical emergency services. The provider follows a ticketing system for all the telephonic calls received across all the departments. Calls to the provider can be for New Appointment, Cancellation, Lab Queries, Medical Refills, Insurance Related, General Doctor Advise etc. The Tickets have the details of Summary of the call and description of the calls written by various staff members with no standard text guidelines.The challenge is, based on the Text in the Summary and Description of the call, the ticket isto be classified to Appropriate Category (out of 5 Categories) and Subcategories (Out of 20Sub Categories).

Pain and Gain Analysis

The Pain and Gain analysis of this project needs very subtlety in Perception and Understanding of the communication usually varies from Tone, Body language, Vocabulary and Absurdity levels while communicating.In real world scenarios, a person needs to understand the subtlety of this usage, requires second-order interpretation of the speaker's or writer's intentions; different parts of the brain must work together to understand purpose.
Using Analytics to identify the purpose of the call will be beneficial. This approach will reduce Time Consumption of text classification. Applications are such as analyzing healthcare calls helps reduce the time to classify and escalate to concerned team.
Examples: Optum Labs, an US research collaborative, has collected EHRs of over 30 million patients to create a database for predictive analytics tools that will improve the delivery of care.

Loading and Preparing the Data for Analysis (Cleaning and processing the data)

- Used data.table library to load the data using fread().
```
given_data = fread("TextClassification_Data.csv")
```
- data.table provides blazing fast speed when it comes to loading data.
- Understanding the types, summary, dimensions and structure of the data from dim(given_data), str(given_data), summary(given_data) commands.
- fread() simply reads everything as character so that's why from str(given_data), I came to know some columns like categories, sub_categories and previous_appointment needs to be converted as factor type.
```
given_data$categories = as.factor(given_data$categories)
given_data$sub_categories = as.factor(given_data$sub_categories)
given_data$previous_appointment = as.factor(given_data$previous_appointment)
```
- The Given dataset contains **57280 Observations** and **7 Variables** - fileid, summary, data,nprevious appointment, categories, sub categories and ID.
- From str(given_data), we can see that the DATA column is in RTF format which needs to be converted to text format.
- I used qdap library which involves replace_contaction() and replace_abbreviation() on the DATA column .
```
given_data[,3] = gsub("[[:punct:]]¥¥w+ *","",given_data[,3])
given_data[,3] = gsub("[[:punct:]]","",given_data[,3]) #removing punctuation
given_data[,3] = gsub("x¥¥w+ *","",given_data[,3])
given_data[,3] = replace_contraction(given_data[,3])
```

```
given_data[,3] = replace_abbreviation(given_data[,3],abbreviation =
qdapDictionaries::abbreviations,replace = NULL, ignore.case = TRUE)
```

- This is how **DATA column** looks after the above code on it:

```
 head(given_data[,3],n = 2)
## [1] "ArialNormalDefault Paragraph FontPhone Note Call patient back atCellPhoneFROM PATIENTCaller Name Caller
PatientCall For NurseOther Patient is returning nurse call He is unable to make appt without talking to fin service dept
However he needs medication and worried that he will have issue without medication Please call patient to discuss
Call Taken by 26 2015 5PMCall backFollowDetails Pt returned phone call Please call back to advise May 27 2015
8AMAdditional FollowDetails What is the problem Is he without insurance He has been nonwith instructions to come
in for a followappt and cannot have refills without oneAdditional Followby David 27 2015 8AMAdditional FollowDetails
RN spoke with pt and relayed the above to him he requested to speak with financialservices RN transferred him to the
business office RN requested Business office to call once matter has been completedFollowby Hollie Saltis RN May 27
2015 11AMAdditional FollowDetails OkAdditional Followby David 27 2015 5PM"
## [2] "ArialNormalDefault Paragraph FontPhone Note Call patient back atCellPhoneFROM PATIENTCaller Name Caller
PatientOther Ptschool teacher is reporting pt is not able to sit still mom wants to know if the Focalin needs some
dosage adjusting andis there something pt could take that the school staff couldadminister Please call back to discuss
Call Taken by May 12 2015 1PMFollowDetails Mom sts patient is having alot of issues with meds and effectiveness
LOV1 3rescheduled to 4and that was a No Show Mom apologized for the no show Shests they disconnected the home
number and only use the cell Appt scheduled for 5at 10 Advd to be here at 945Action Taken Phone Call Completed
Appt ScheduledFollowby Marcia Richardson LPN May 12 2015 4PM"
```

- From str(given_data), summary(given_data), I came to know, it is better to ignore fileid, ID variables as they are unique for every patient and they can disrupt the performance of the machine learning model.
- Using dplyr package to select the required columns and escaping fileid and ID variables
  data_req = select(given_data,SUMMARY:previous_appointment).
- Used table command to check the unique values of categorical variables.
  table(data_req$categories);table(data_req$sub_categories);table(data_req$previous_appointment)
- From the above table output, we can see there are multiple factors because of case sence. So removing such noice using mapvalues from plyr package to correctly change these multiple factors.

```
data_req$categories =  mapvalues(data_req$categories,from = c("asK_A_DOCTOR",  "mISCELLANEOUS",
 "JUNK"), to = c("ASK_A_DOCTOR","MISCELLANEOUS","MISCELLANEOUS"))
data_req$sub_categories = mapvalues(data_req$sub_categories,
            from = c("mEDICATION RELATED","JUNK"),to = c("MEDICATION RELATED","OTHERS"))
data_req$previous_appointment = mapvalues(data_req$previous_appointment,
               from = c("","No","NO","yes","Yes","YES"), to = c(NA,0,0,1,1,1))
```

- I have created a new data set data_req with the noise free target variables and without fileid and ID variables.
  Dimensions of data_req: 57280   5

```
 SUMMARY                  DATA                         categories
 Length:57280         Length:57280                    APPOINTMENTS :13872
Class :character      Class :character                ASK_A_DOCTOR :11800
Mode :character      Mode :character                   MISCELLANEOUS:12191
                                                       LAB : 4321
                                                       PRESCRIPTION :15096
```

```
sub_categories                                              previous_appointment
MEDICATION RELATED :10599
0:57085
NEW APPOINTMENT :10478                                                           1:
195
REFILL : 9819
OTHERS : 7377
SHARING OF HEALTH RECORDS (FAX, E-MAIL, ETC.): 3550
LAB RESULTS : 2650
(Other) :12807
```

- It was turn to find any missing values in the data.
```
table(is.na(data_req))
sapply(data_req,function(x)sum(is.na(x)))
which(is.na(data_req$previous_appointment))
```
- I find that row no 14129, 16570 has missing values for column previous_appointment
```
data_req$previous_appointment[c(14129,16570)] = 0
```
- Imputing the missing values with 0 because >90% are 0

Data Cleaning and Text preprocessing

Before we dive into analyzing text, we need to preprocess it. Text data contains white spaces, punctuations, stop words etc. These characters do not convey much information and are hard to process. For example, English stop words like “the”, “is” etc. do not tell you much information about the sentiment of the text, entities mentioned in the text, or relationships between those entities. Depending upon the task at hand, we deal with such characters differently. This will help isolate text mining in R on important words.

Case Folding

The first preprocessing step is Case folding. Here, we are converting all the letters in the Corpus to lowercase using R’s base function tolower.

Remove Numbers

In this step, we are freeing corpus from numbers. Here, we use tm’s removeNumbers function.

Removing Stop Words

This step is about eliminating words that doesn’t make any meaning. Stopwords of English would be enough, but since the dataset contains several short words in the form of short forms which are of no meaning to use. I used stopwords(“en”) and stopwords(“SMART”).

removing Punctuation

We have use tm’s removePunctuation function to remove all punctuation marks such as comma, full stop, parenthesis, various brackets etc.., from the corpus

Stemming

For grammatical reasons, document contains different inflectional forms like tense forms and derivational forms, we are performing stemming to reduce all those words to their root word. We are using tm's stemDocument function to do this. Stemming greatly help in reducing total number of terms and increase weighting.

Stripping White Spaces

The above performed preprocessing steps left our corpus with many leading and trailing whitespaces within documents. We are cleaning all of them in one go using tm's stripWhitespace function. With this step our basic preprocessing is completed.

A user defined function clean_corpus is created to do preprocessing and cleaning of the corpus. This function takes in a vector with all the text in it and convert it to a corpus and cleans it.

```
#function for cleaning the corpus by tm library
clean_corpus <- function(data){
  data_corpus <- Corpus(VectorSource(data)) #forming a corpus
  data_corpus <- tm_map(data_corpus,removePunctuation) #removing punctuations
  data_corpus <- tm_map(data_corpus,removeNumbers) #removing numbers
  data_corpus <- tm_map(data_corpus,tolower) #converting to lowercase
  data_corpus <- tm_map(data_corpus,removeWords,stopwords("English")) #removing english stopwords
  data_corpus <- tm_map(data_corpus,removeWords,stopwords("SMART"))
  data_corpus <- tm_map(data_corpus,stemDocument) #performing stemming
  data_corpus <- tm_map(data_corpus,stripWhitespace) #removing the whitespaces
 }
```

After cleaning the DATA and SUMMARY columns using clean_corpus, form a Document Term Matrix for each corpus.

A document term matrix is an important representation for text mining in R tasks and an important concept in text analytics.
Each row of the matrix is a document vector, with one column for every term in the entire corpus.
tm makes it very easy to create the term-document matrix.

```
dtm_SUMMARY <- DocumentTermMatrix(data_corpus_SUMMARY,control = list(weighting = weightTf))
dtm_DATA <- DocumentTermMatrix(data_corpus_DATA,control = list(weighting = weightTf))
```

```
## <<DocumentTermMatrix (documents: 57280, terms: 9750)>>
## Non-/sparse entries: 160261/558319739
## Sparsity : 100%
## Maximal term length: 36
## Weighting : term frequency (tf)

## <<DocumentTermMatrix (documents: 57280, terms: 73656)>>
## Non-/sparse entries: 2077704/4216937976
## Sparsity : 100%
## Maximal term length: 75
## Weighting : term frequency (tf)
```

A Document Term Matrix (DTM) is created from the corpus. Term Frequency is considered as weighting to create Document term matrix to keep DTM simple. DATA DTM has 57280 documents and 73656 terms with 100% sparsity and SUMMARY DTM has 57280 documents and 9750 terms with 100% sparsity.

```
#not able to convert Document matrix of SUMMARY and DATA to a matrix
SUMMARY_data_dtm <- as.matrix(dtm_SUMMARY)
DATA_data_dtm <- as.matrix(dtm_DATA)
Error: cannot allocate vector of size 4.2 Gb
```

Now I m removing those terms which don't appear too often in this data(dtm . We will remove any element that doesn't appear in atleast 1% of the entries (or documents). Relating to the above created DTM we are basically removing those columns whose entries are 1 in least number of documents.

```
 #Removing sparse terms
sparsed_dtm_SUMMARY = removeSparseTerms(dtm_SUMMARY,0.999)
sparsed_dtm_DATA = removeSparseTerms(dtm_DATA,0.99)
```

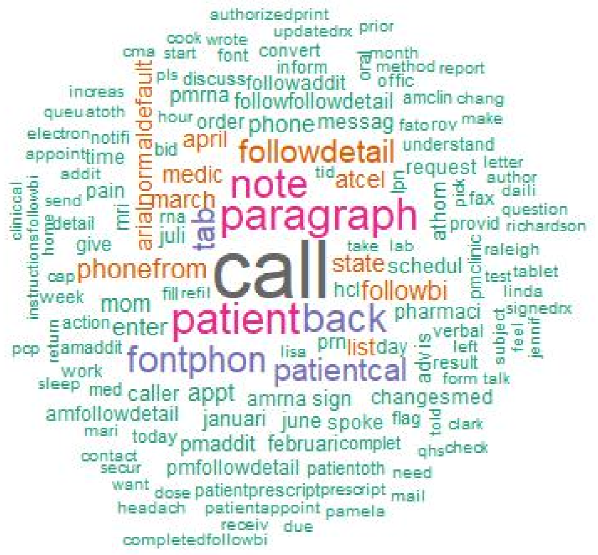
Sparsity is reduced and we made 73656+9750 terms to more relevant 510+403 terms. We try to get a balance between number of terms and vector size which R can allocate while processing. This Document term matrices is then converted to Data frame for Feature engineering.

```
## <<DocumentTermMatrix (documents: 57280, terms: 403)>>
## Non-/sparse entries: 124114/22959726
## Sparsity : 99%
## Maximal term length: 12
## Weighting : term frequency (tf)
## <<DocumentTermMatrix (documents: 57280, terms: 510)>>
## Non-/sparse entries: 1505072/27707728
## Sparsity : 95%
## Maximal term length: 36
## Weighting : term frequency (tf)
## [1] 57280 403 #dimensions of data set formed by SUMMARY DTM
## [1] 57280 510 #dimensions of data set formed by DATA DTM
## [1] 57280 913 #dimension of data set by combining SUMMARY and DATA
```

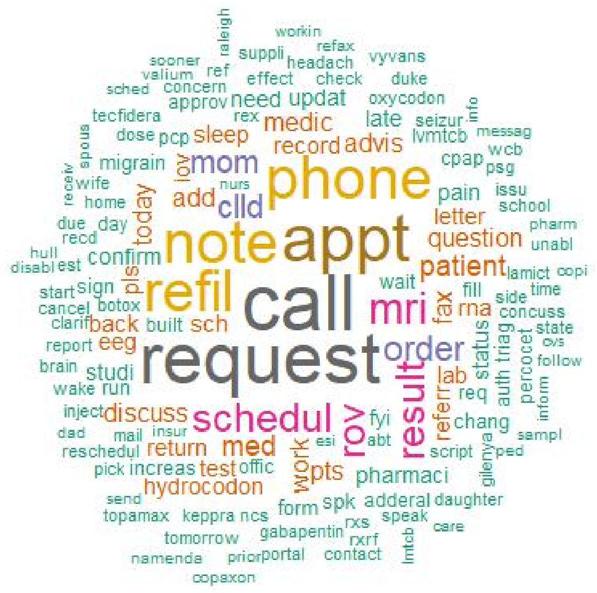
Combined Data frame has 57280 observations and 913 features (excluding target class).

Data Visualization (Exploratory Data Analysis)

wordcloud for corpus data column before applyng dtm and sparse



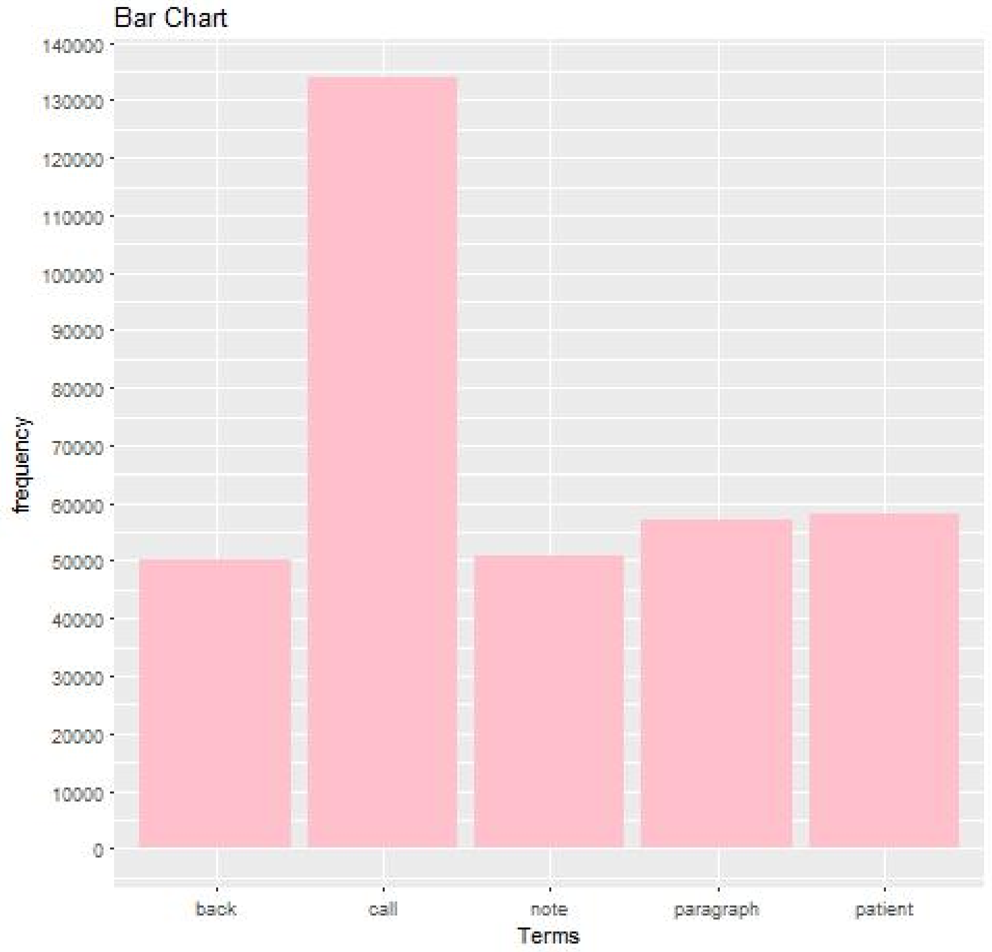
Word cloud for corpus summary column before applying dtm and sparse



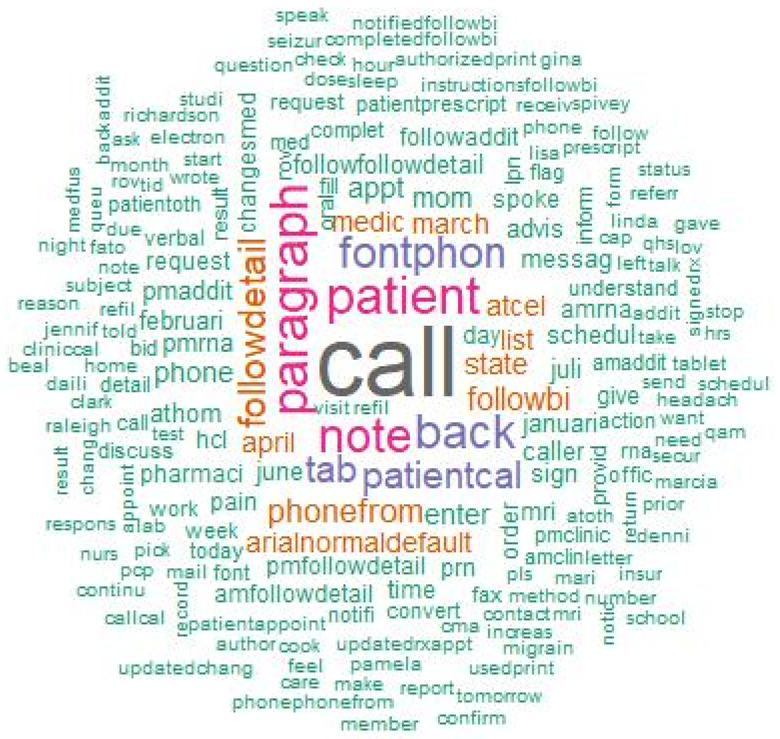
Combined_ data (SUMMARY + DATA)

	Terms	frequency
1:	call	134005
2:	patient	58324
3:	paragraph	57237
4:	note	50724
5:	back	50197

Equivalent Bar chart for the combined data



Word cloud for the combined data (SUMMARY + DATA column after creating dtm and applying sparse

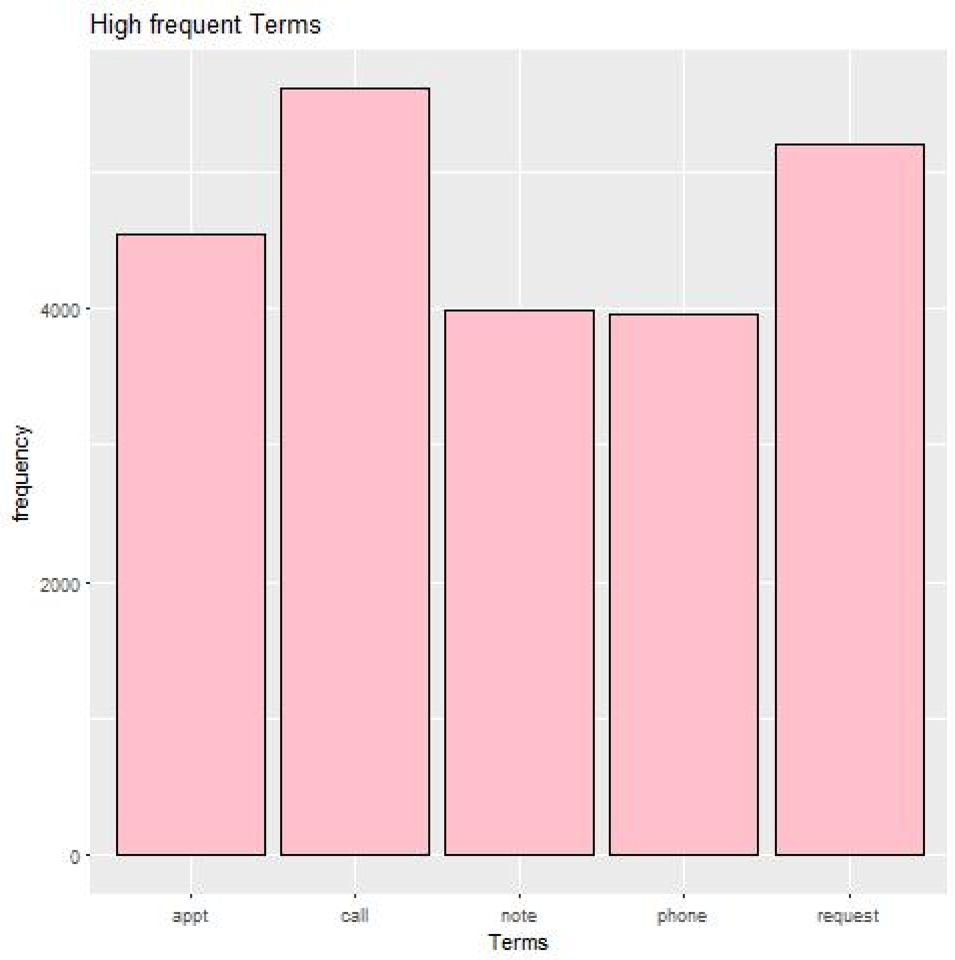


Terms frequency

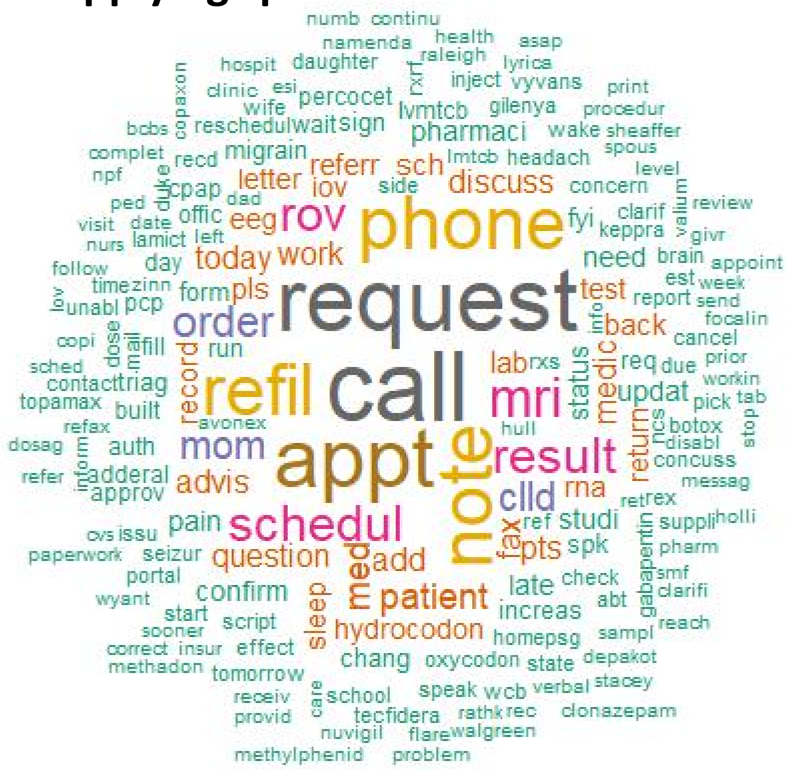
1:	call	5613
2:	request	5202
3:	appt	4535
4:	note	3982
5:	phone	3963

SUMMARY_data

Equivalent Bar chart for the combined data



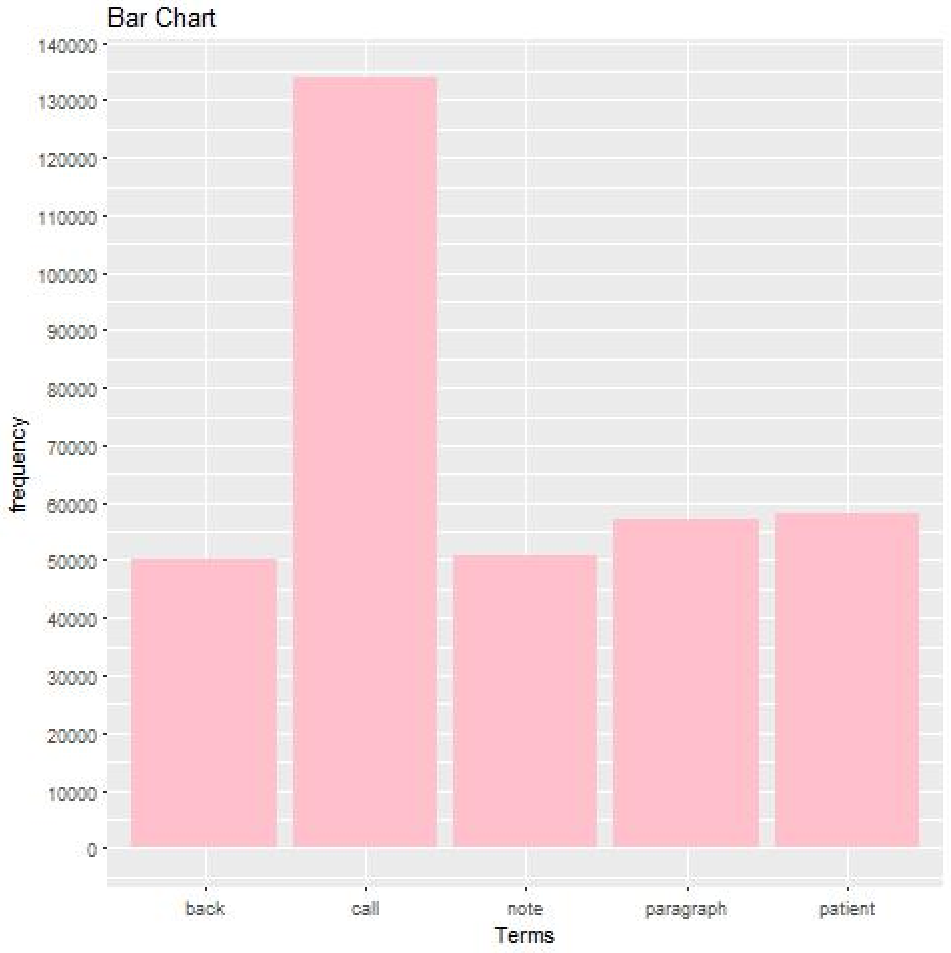
Word cloud for the SUMMARY column after creating dtm and applying sparse



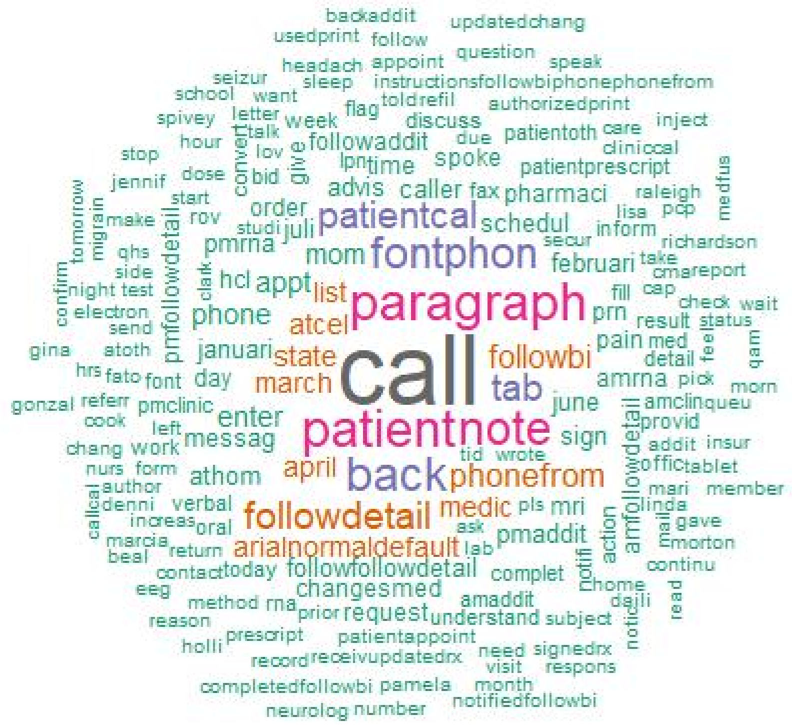
DATA_data

	Terms	frequency
1:	call	134005
2:	patient	58324
3:	paragraph	57237
4:	note	50724
5:	back	50197

Equivalent Bar chart for the combined data



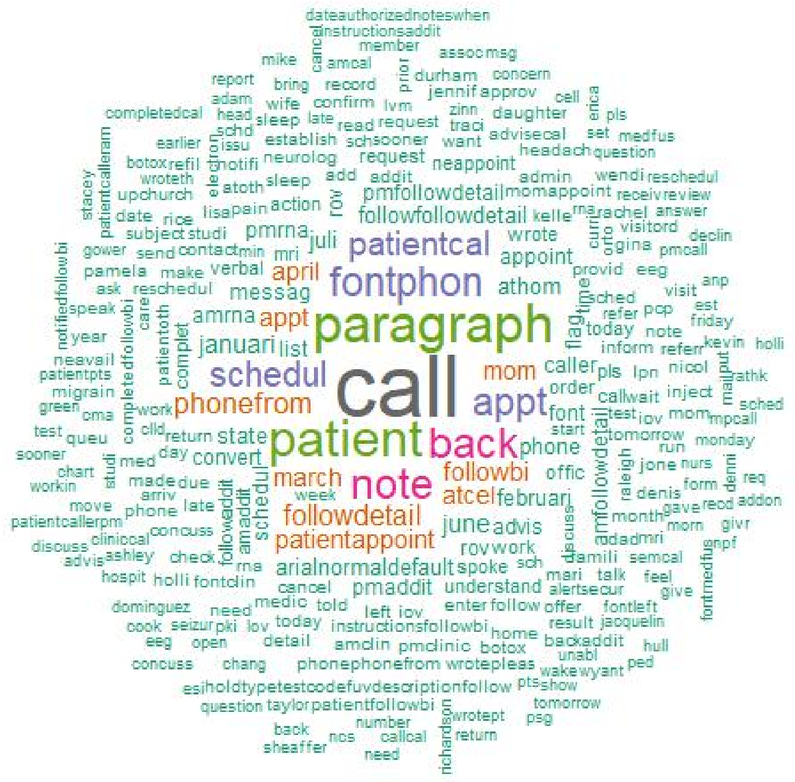
Word cloud for the DATA column after creating dtm and applying sparse



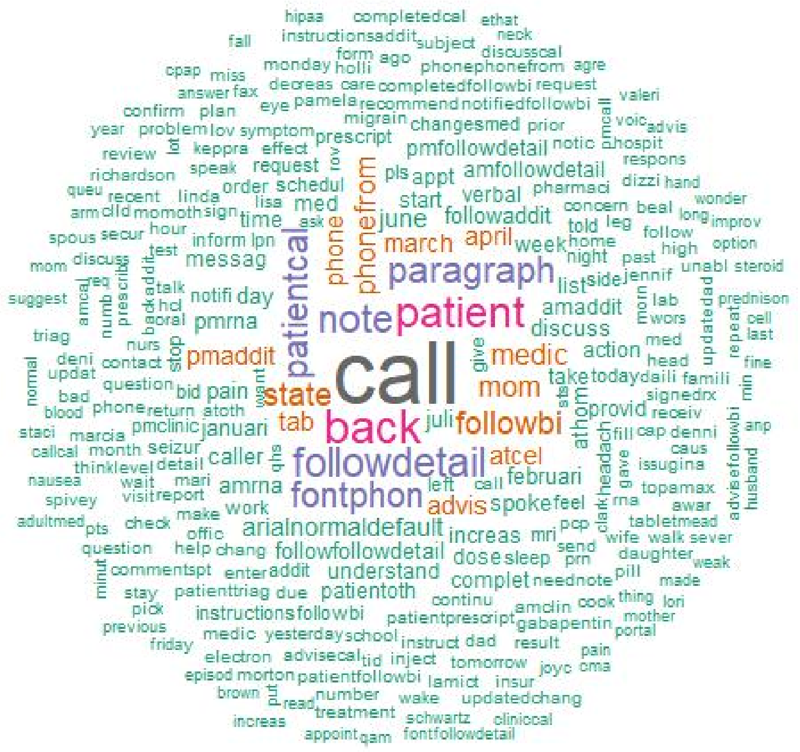
- We can clearly see ‘call’, ‘patient’, ‘paragraph’, ‘note’ and ‘back’ are top 5 most frequent words for the combined data(SUMMARY + DATA column). We further ensure this hypothesis from a bar plot.
- We can clearly see ‘call’, ‘request’, ‘apt, ‘note’ and ‘phone’ are top 5 most frequent words for the SUMMARY column. We further ensure this hypothesis from a bar plot.
- We can clearly see ‘call’, ‘patient’, ‘paragraph’, ‘note’ and ‘back’ are top 5 most frequent words for the DATA column. We further ensure this hypothesis from a bar plot.

Let’s explore category wise worclouds to understand data more clearly for the combined_data (SUMMARY + DATA)

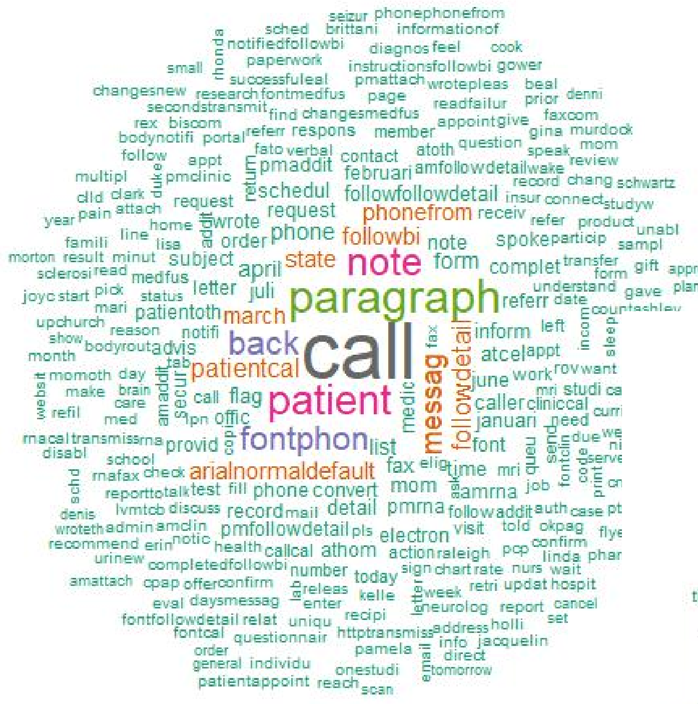
APPOINTMENT CLOUD



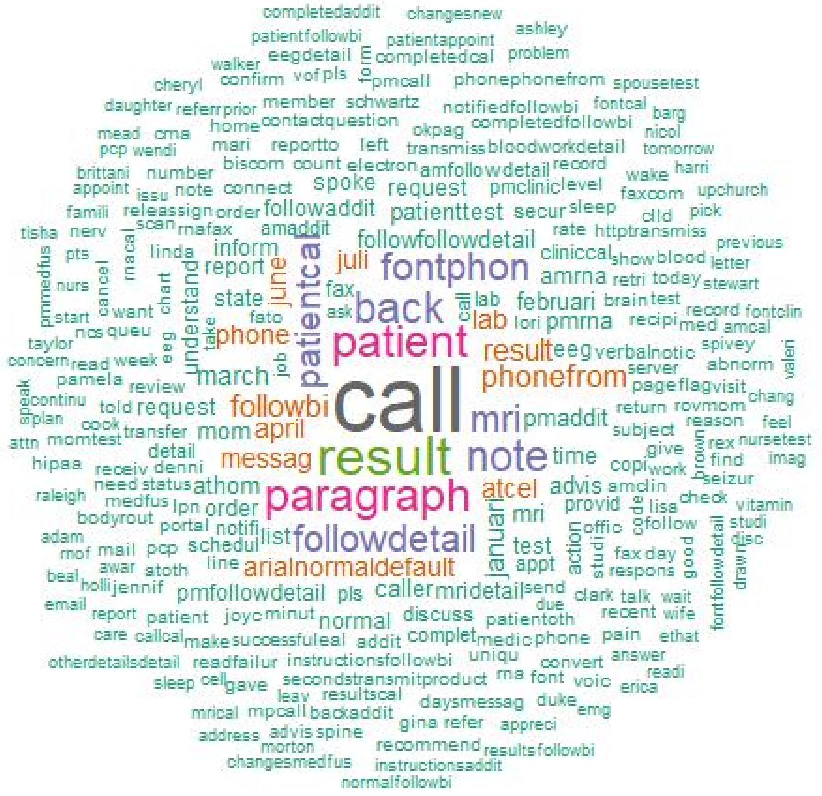
ASK A DOCTOR CLOUD



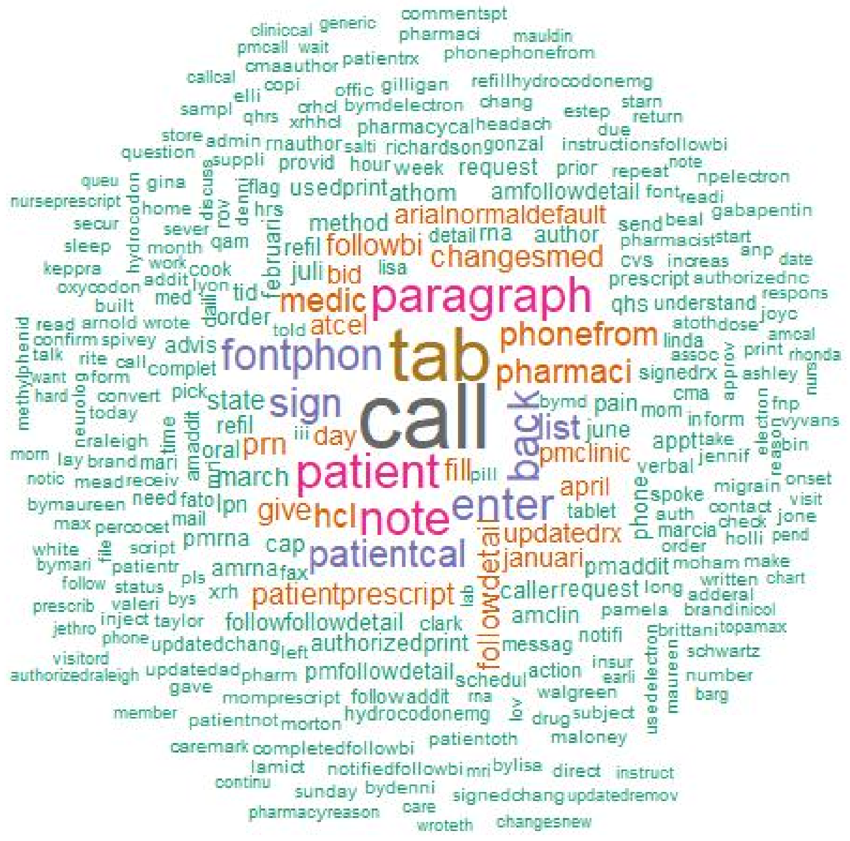
MISCELLANEOUS CLOUD



LAB CLOUD

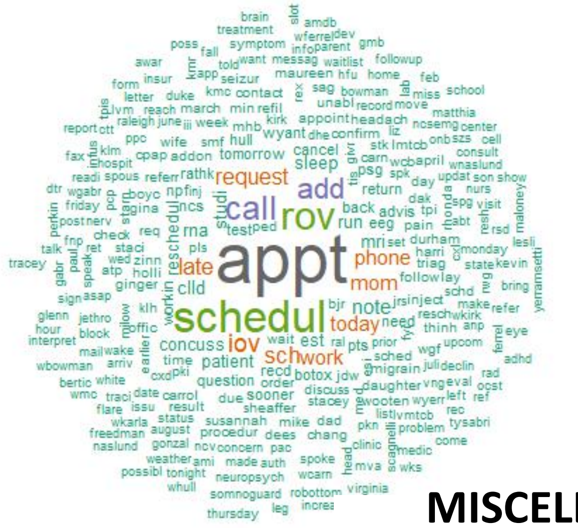


PRESCRIPTION CLOUD

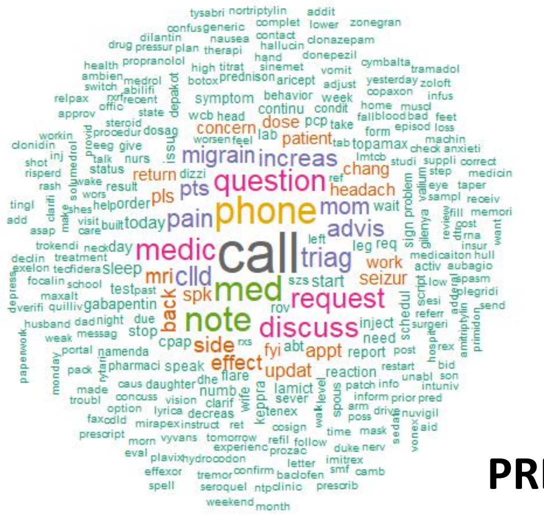


Let’s explore category wise worclouds to understand data more clearly for the combined data SUMMARY_data

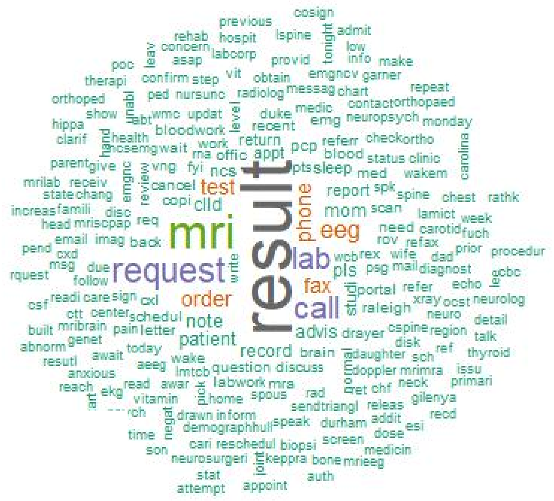
APPOINTMENT CLOUD



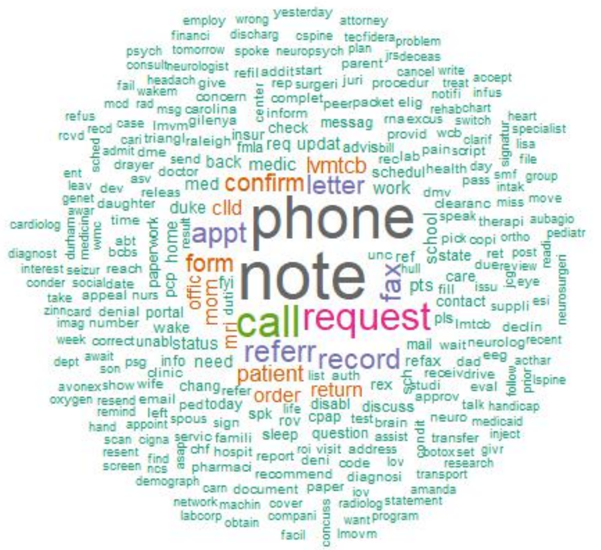
ASK A DOCTOR CLOUD



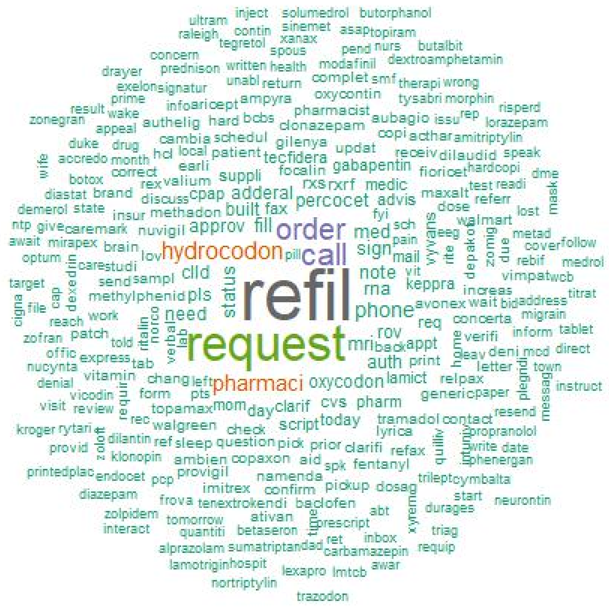
LAB CLOUD



MISCELLANEOUS CLOUD

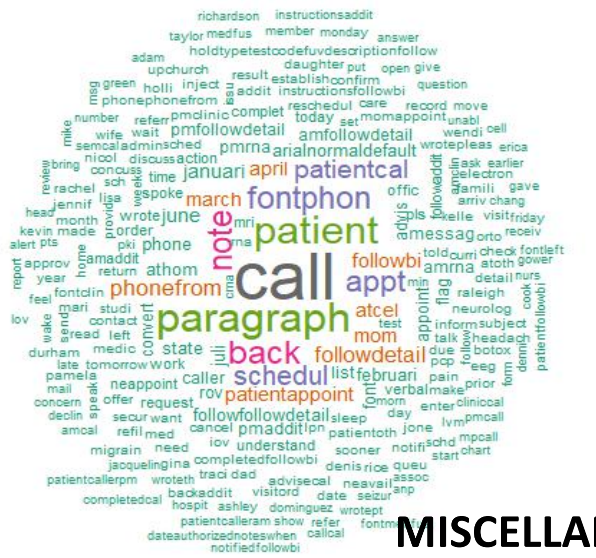


PRESCRIPTION CLOUD

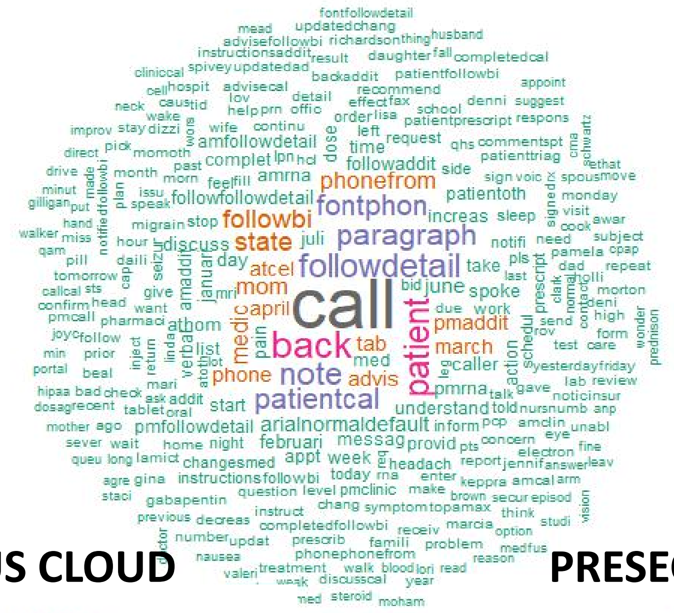


Let’s explore category wise worclouds to understand data more clearly for the DATA_data

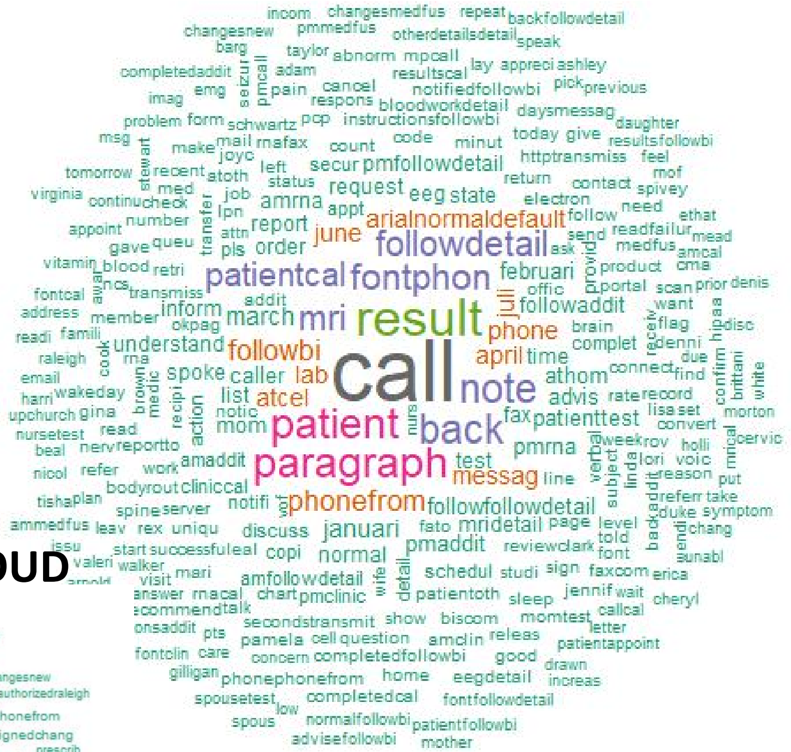
APPOINTMENT CLOUD



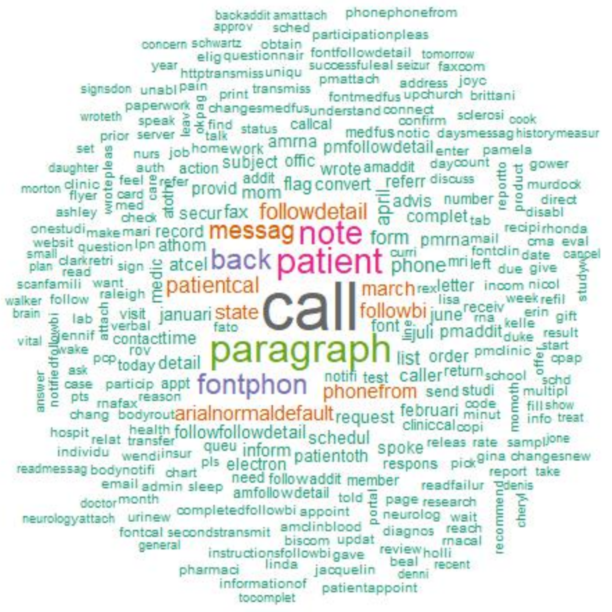
ASK A DOCTOR CLOUD



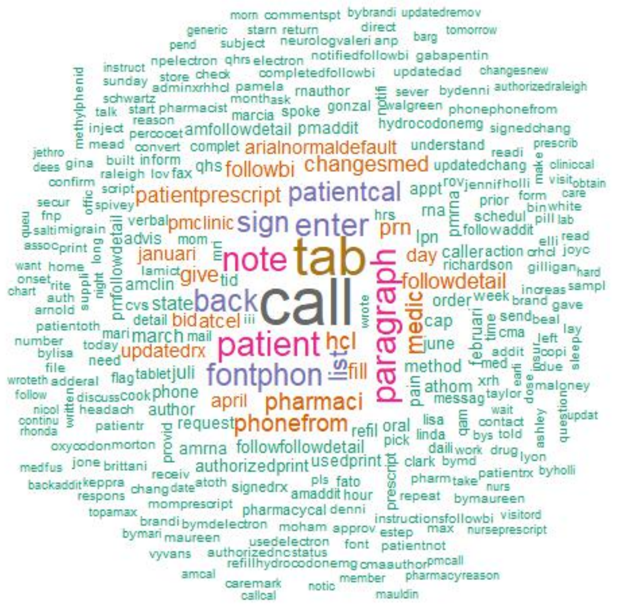
LAB CLOUD



MISCELLANEOUS CLOUD

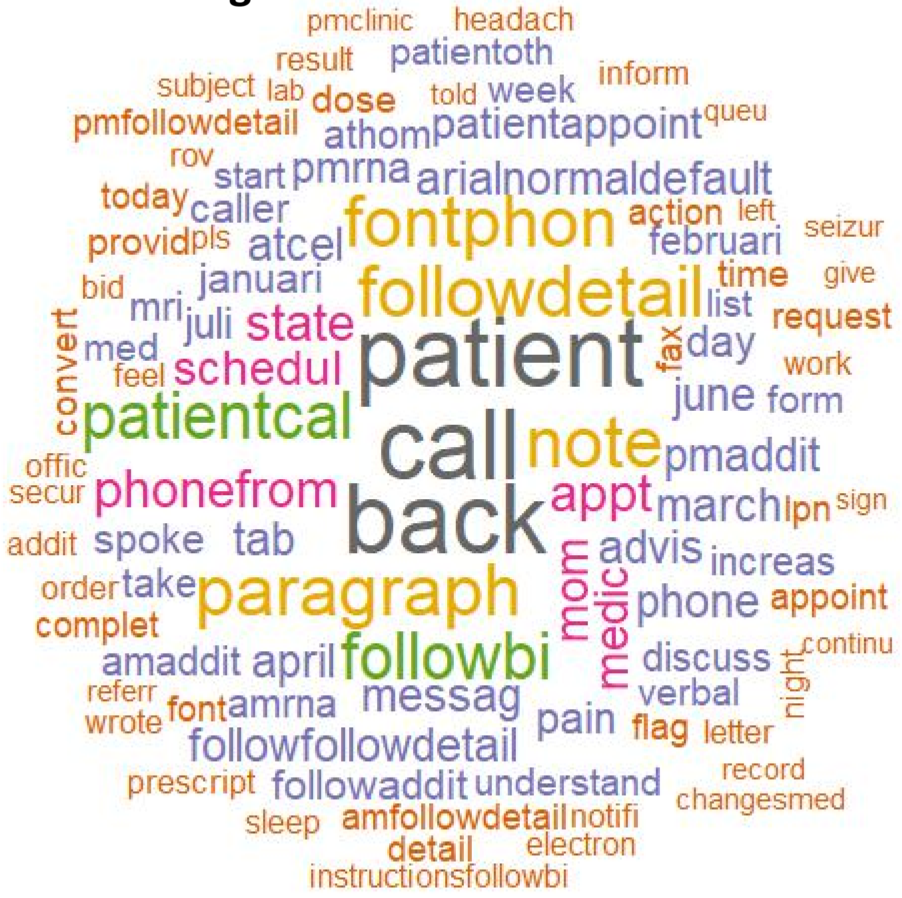


PRESECRIPTION CLOUD

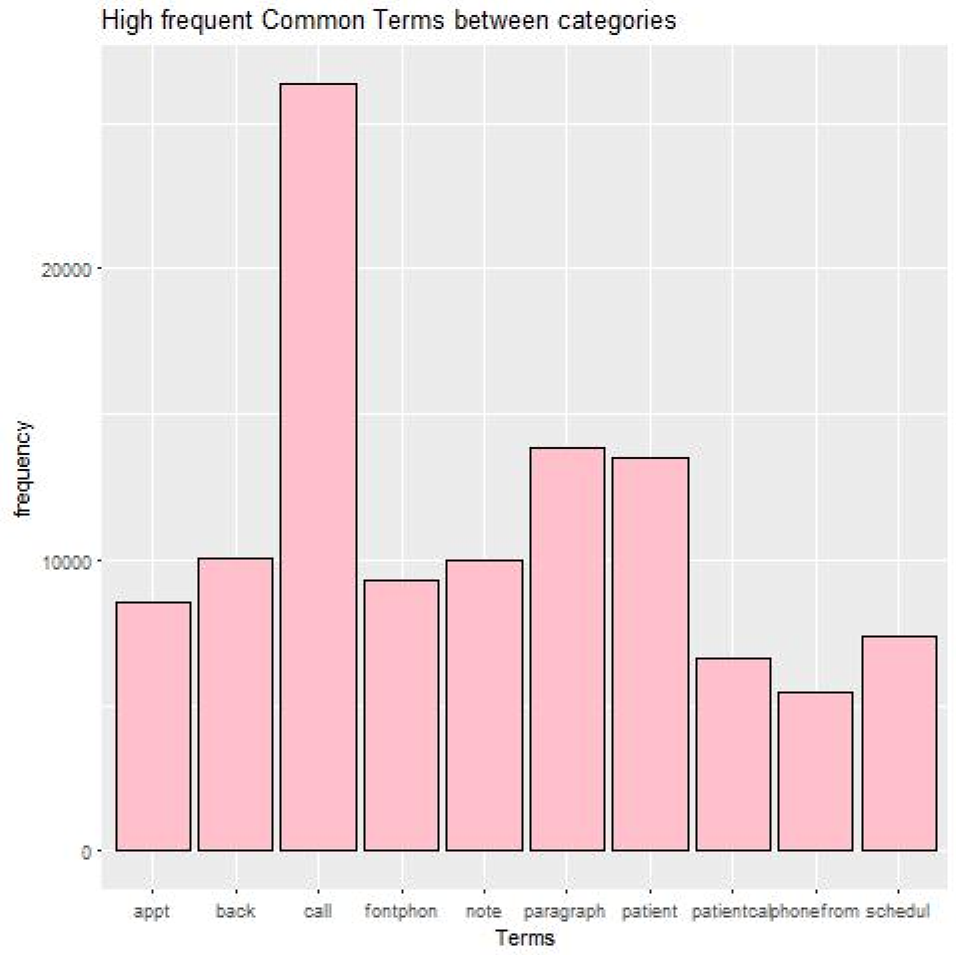


I have explored the individual datasets of each categories to have a better understanding of the generated terms for SUMMARY_data, DATA_data and combined_data. And also extracted the most common terms between the categories which I used in feature engineering.

World cloud-Common Tems between all the categories



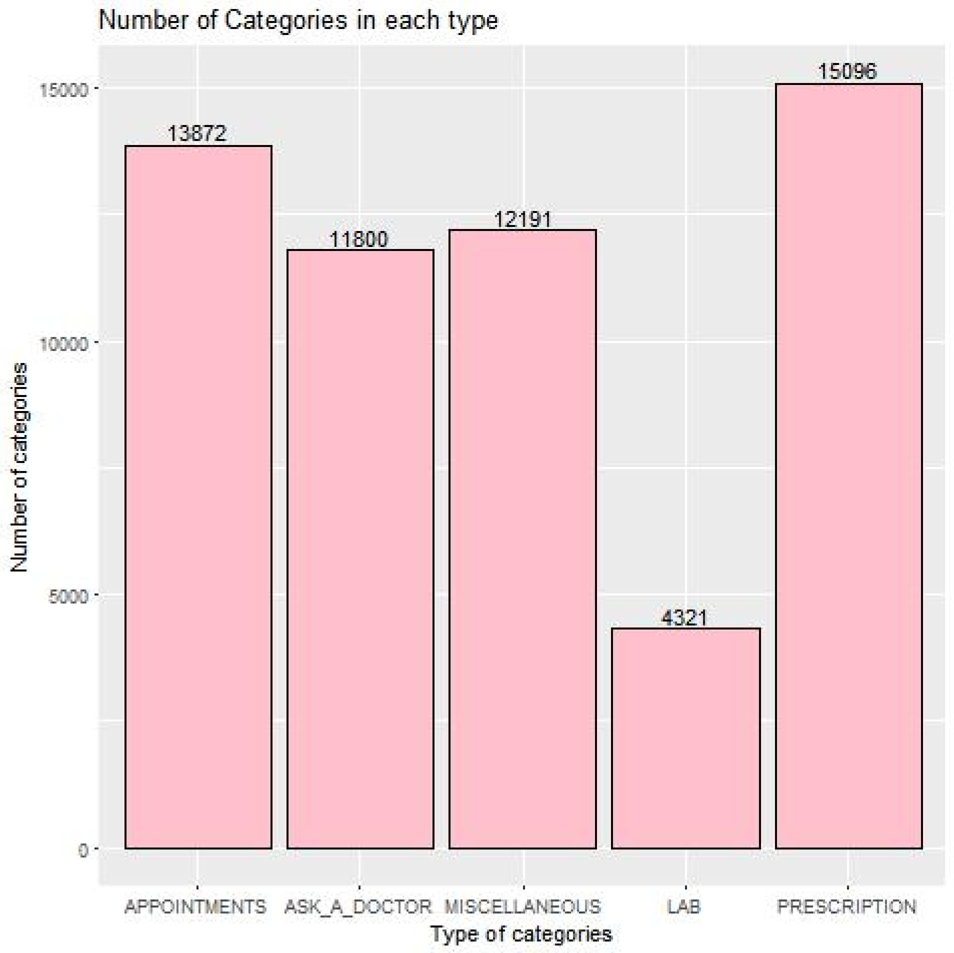
Corresponding bar plot

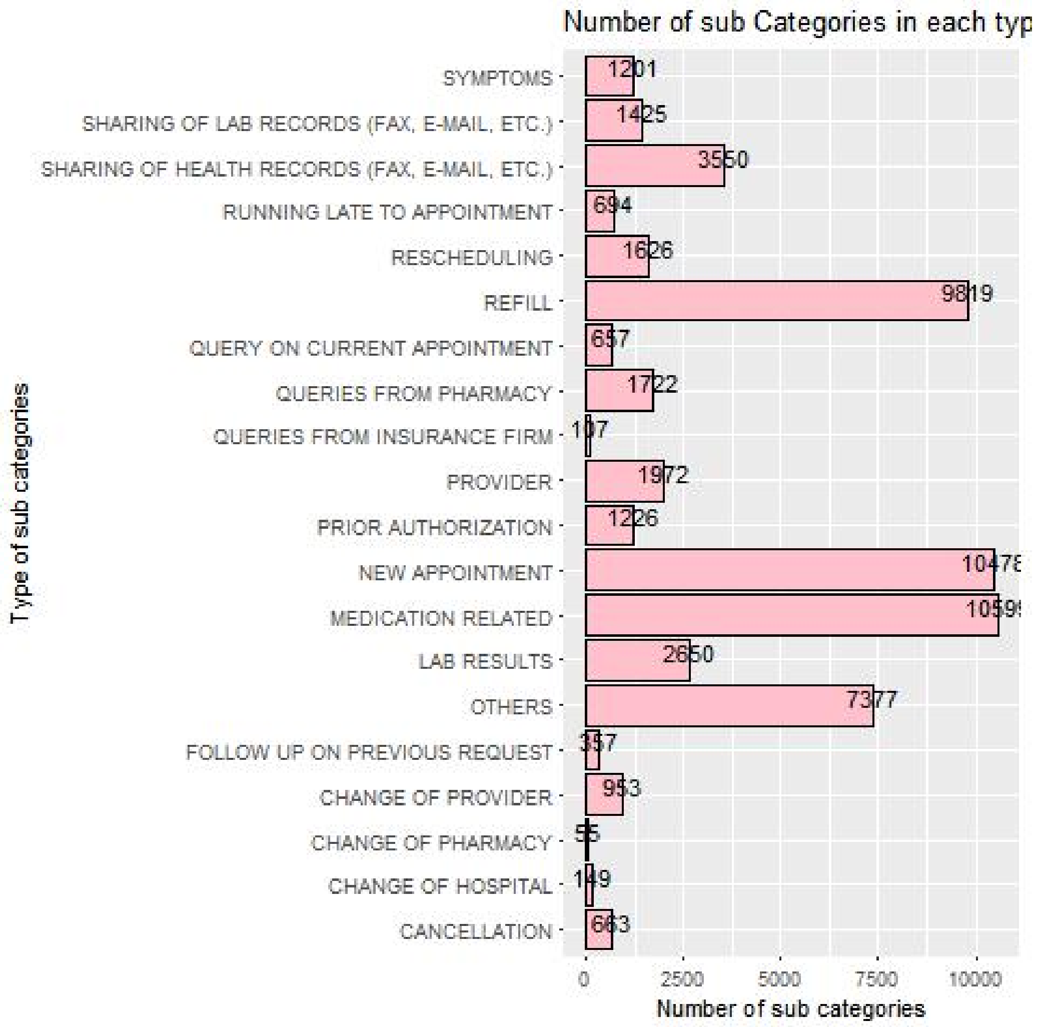


```
#Finding the common terms present in any two categories from SUMMARY column
common_SUMMARY = app_freq_SUMMARY[app_freq_SUMMARY$Terms %in% ask_freq_SUMMARY$Terms,] %>%
  rbind(app_freq_SUMMARY[app_freq_SUMMARY$Terms %in%  mis_freq_SUMMARY$Terms,]) %>%
  rbind(app_freq_SUMMARY[app_freq_SUMMARY$Terms %in% lab_freq_SUMMARY$Terms,]) %>%
  rbind(app_freq_SUMMARY[app_freq_SUMMARY$Terms %in% pre_freq_SUMMARY$Terms,]) %>%
  rbind(ask_freq_SUMMARY[ask_freq_SUMMARY$Terms %in% mis_freq_SUMMARY$Terms,]) %>%
  rbind(ask_freq_SUMMARY[ask_freq_SUMMARY$Terms %in% lab_freq_SUMMARY$Terms,]) %>%
  rbind(ask_freq_SUMMARY[ask_freq_SUMMARY$Terms %in% pre_freq_SUMMARY$Terms,]) %>%
  rbind(mis_freq_SUMMARY[mis_freq_SUMMARY$Terms %in% lab_freq_SUMMARY$Terms,]) %>%
  rbind(mis_freq_SUMMARY[mis_freq_SUMMARY$Terms %in% pre_freq_SUMMARY$Terms,]) %>%
  rbind(lab_freq_SUMMARY[lab_freq_SUMMARY$Terms %in% pre_freq_SUMMARY$Terms,])
#Finding the common terms present in any two categories from DATA column
common_DATA = app_freq_DATA[app_freq_DATA$Terms %in% ask_freq_DATA$Terms,] %>%
  rbind(app_freq_DATA[app_freq_DATA$Terms %in% mis_freq_DATA$Terms,]) %>%
  rbind(app_freq_DATA[app_freq_DATA$Terms %in% lab_freq_DATA$Terms,]) %>%
  rbind(app_freq_DATA[app_freq_DATA$Terms %in% pre_freq_DATA$Terms,]) %>%
  rbind(ask_freq_DATA[ask_freq_DATA$Terms %in% mis_freq_DATA$Terms,]) %>%
  rbind(ask_freq_DATA[ask_freq_DATA$Terms %in% lab_freq_DATA$Terms,]) %>%
  rbind(ask_freq_DATA[ask_freq_DATA$Terms %in% pre_freq_DATA$Terms,]) %>%
  rbind(mis_freq_DATA[mis_freq_DATA$Terms %in% lab_freq_DATA$Terms,]) %>%
  rbind(mis_freq_DATA[mis_freq_DATA$Terms %in% pre_freq_DATA$Terms,]) %>%
  rbind(lab_freq_DATA[lab_freq_DATA$Terms %in% pre_freq_DATA$Terms,])

common = rbind(common_DATA,common_SUMMARY)
```

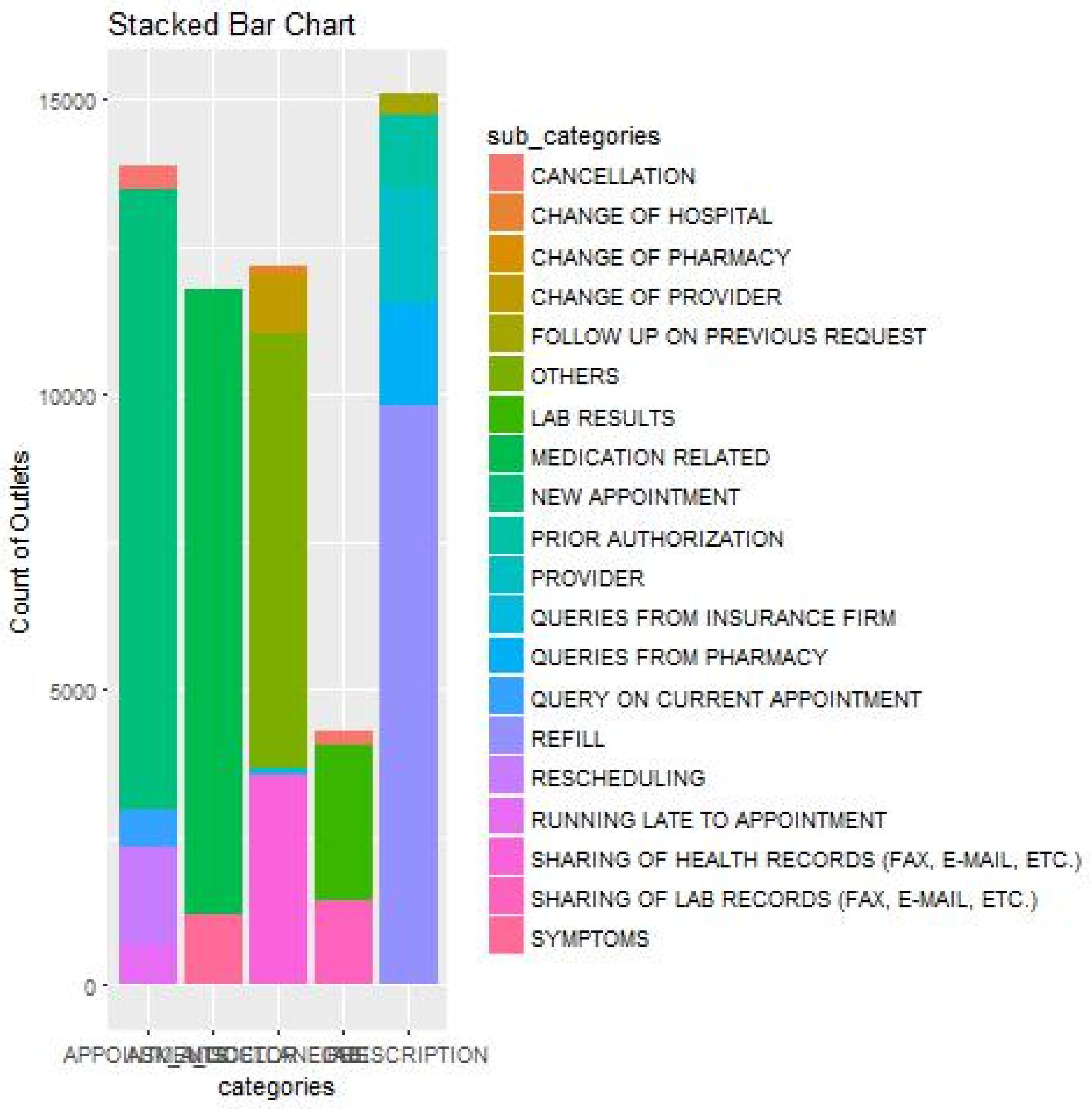
- Words like ‘call’, ‘patient’, ‘note’, ‘paragraph’, ‘back’ are most common frequent words in all kinds.
- There are many common terms between categories which may not be useful to the model. We have to figure out whether to remove most common terms or all common terms.

Additional Visualizations:

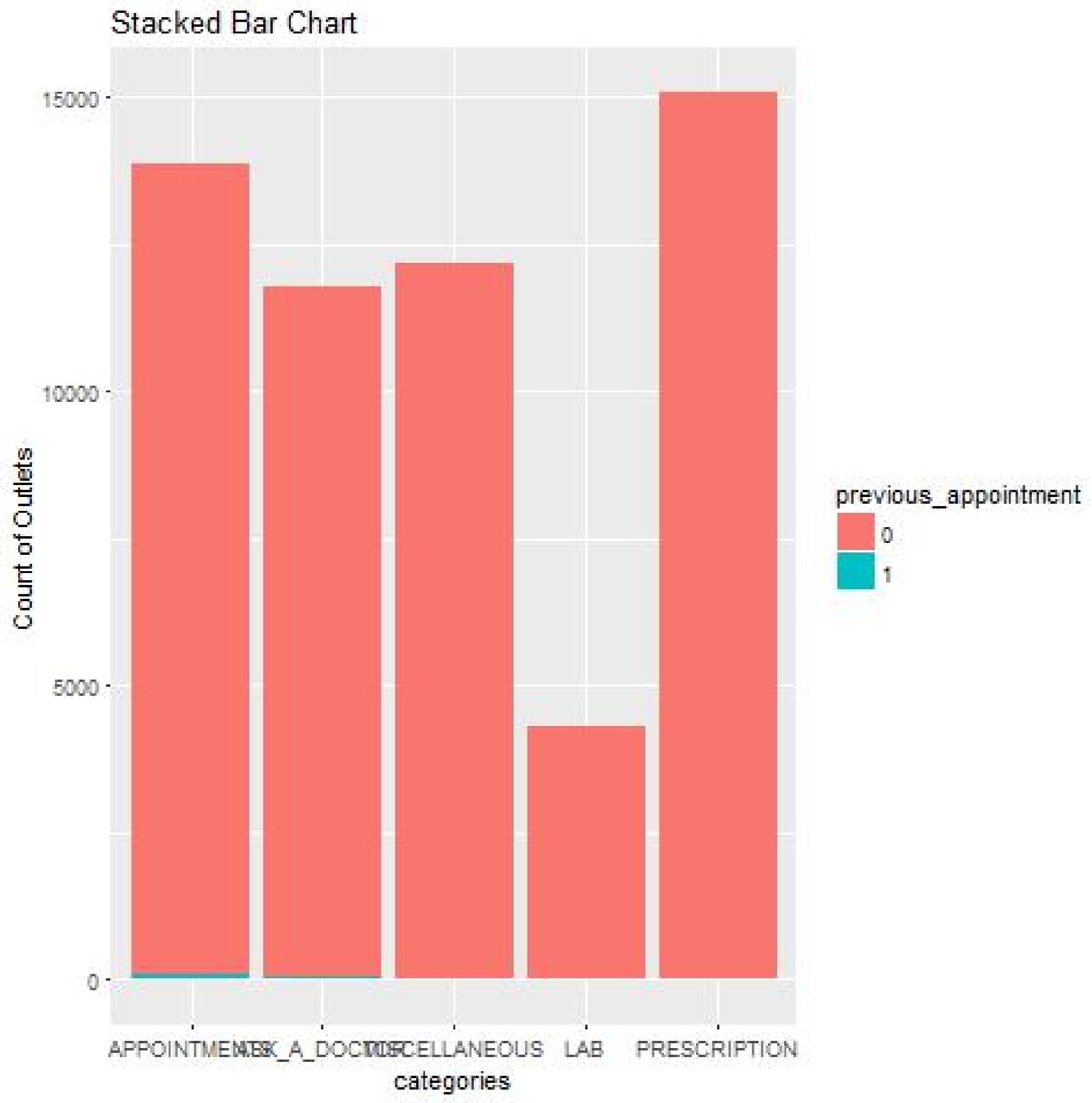


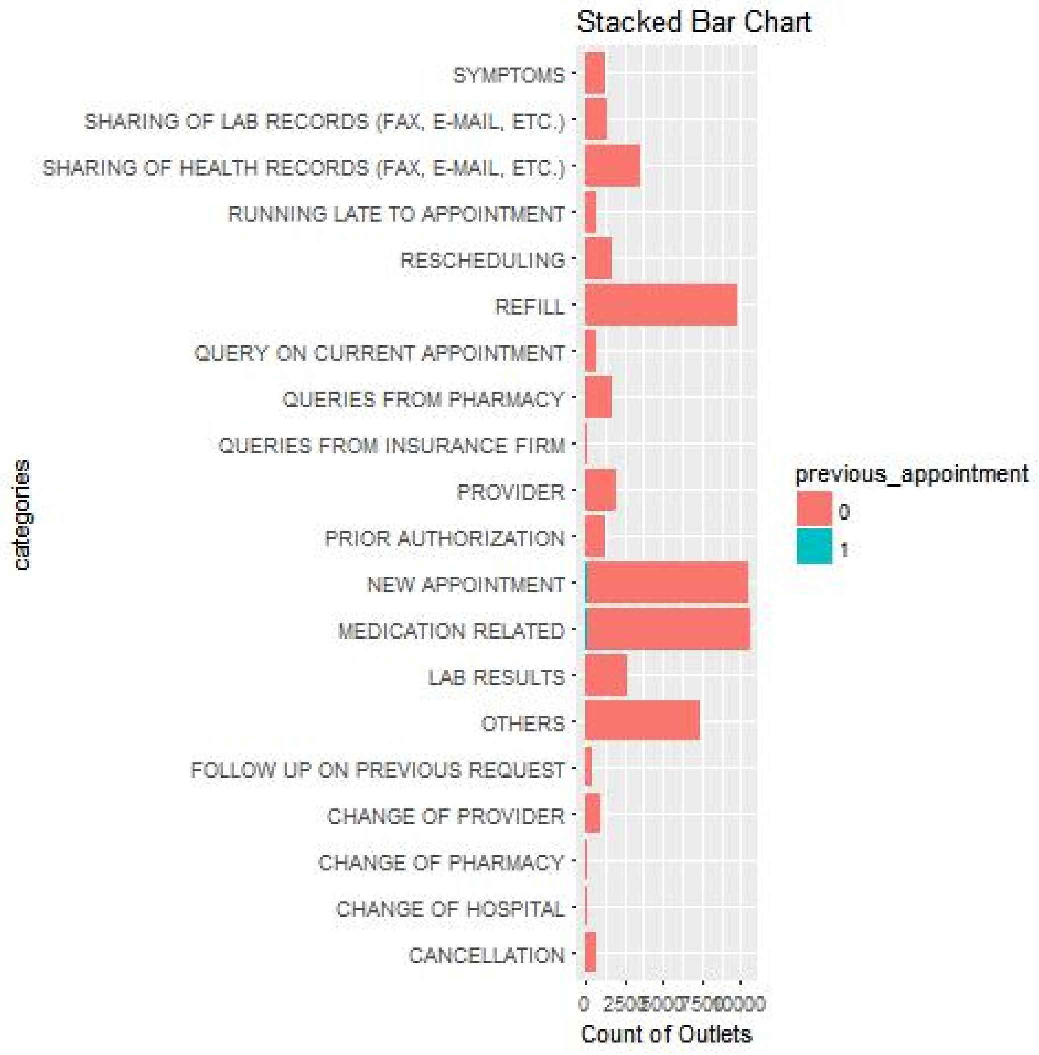


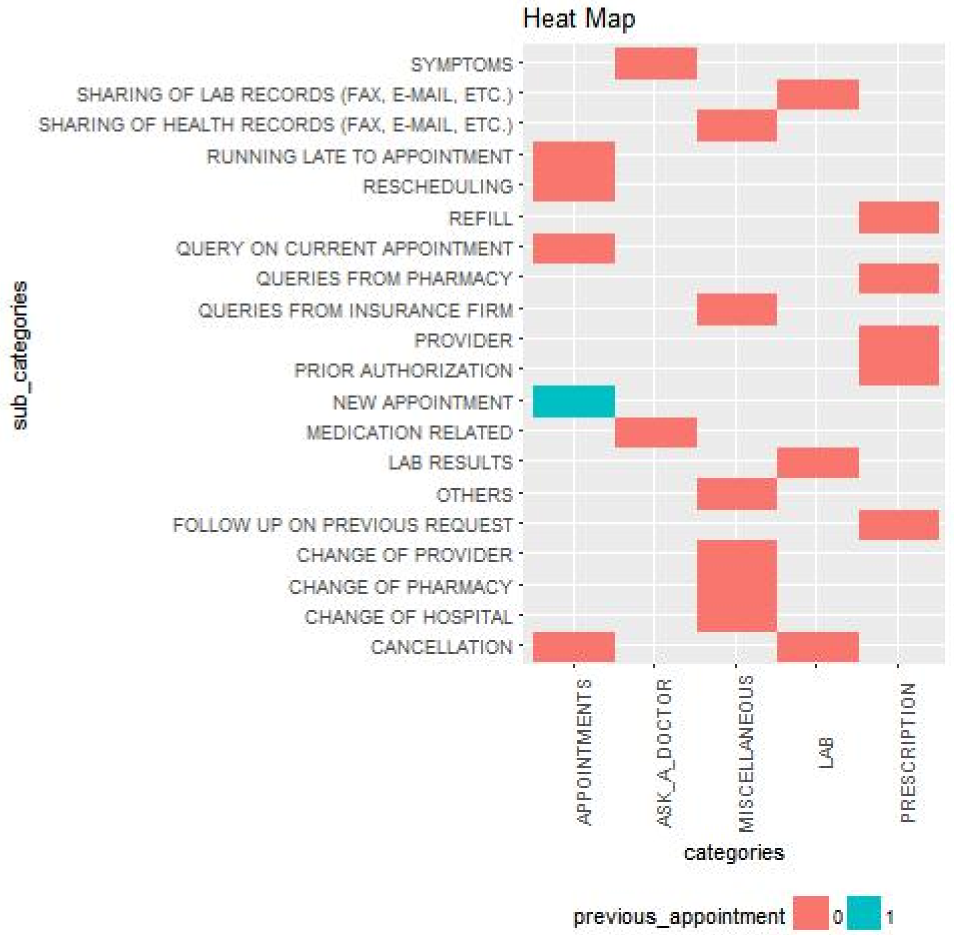
Stacked bar chart between categories &subcategories



Stacked Bar plot between categories & previous appointment




Stacked Bar Chart
SYMPTOMS
SHARING OF LAB RECORDS (FAX, E-MAIL, ETC.)
SHARING OF HEALTH RECORDS (FAX, E-MAIL, ETC.)
RUNNING LATE TO APPOINTMENT
RESCHEDULING
REFILL
QUERY ON CURRENT APPOINTMENT
QUERIES FROM PHARMACY
QUERIES FROM INSURANCE FIRM
PROVIDER
PRIOR AUTHORIZATION
NEW APPOINTMENT
MEDICATION RELATED
LAB RESULTS
OTHERS
FOLLOW UP ON PREVIOUS REQUEST
CHANGE OF PROVIDER
CHANGE OF PHARMACY
CHANGE OF HOSPITAL
CANCELLATION
categories
0 2500 5000 7500 10000
Count of Outlets
previous_appointment
0
1


Heat Map
SYMPTOMS
SHARING OF LAB RECORDS (FAX, E-MAIL, ETC.)
SHARING OF HEALTH RECORDS (FAX, E-MAIL, ETC.)
RUNNING LATE TO APPOINTMENT
RESCHEDULING
REFILL
QUERY ON CURRENT APPOINTMENT
QUERIES FROM PHARMACY
QUERIES FROM INSURANCE FIRM
PROVIDER
PRIOR AUTHORIZATION
NEW APPOINTMENT
MEDICATION RELATED
LAB RESULTS
OTHERS
FOLLOW UP ON PREVIOUS REQUEST
CHANGE OF PROVIDER
CHANGE OF PHARMACY
CHANGE OF HOSPITAL
CANCELLATION
sub_categories
APPOINTMENTS
ASK_A_DOCTOR
MISCELLANEOUS
LAB
PRESCRIPTION
categories
previous_appointment
0
1

Feature engineering

- Checking correlation between the predictors is a must in Analysis. We have used tm’s findAssocs and pearson’s correlation matrix to detect correlation and association. A correlation matrix is built representing positive and negative correlations between terms. We have taken 85% as correlation limit and filtered out highly correlated terms from our structured data. Words like “marcia , richardson”, “brown , lori”, “linda , clark”, “tisha , walker”, “cook , denni” are highly correlated pairs. A term is taken from each correlation pair and made a vector called corr.terms.

```
## [1] "Number of correlated terms which are to be filtered are:"
## [1] 40
```

- We have to remove these corelated terms from the variable of our combined_data set.Thus forming a data set without highly (>85%) corelated terms.When I designed a model with the dataset without corelated words, The accuracy is improved by 2-3%.
- It was clear that removing these correlated terms, thus reducing our dimensionality, increases our machine learning model performance.
- Further I thought, it is advantageous to remove common words between all the categories column could yield the model performance.
- When we visualized the wordclouds of each category, there are many common wordscbetween the categories. These common words may effect the model accuracy. Let’s see how these common words impact our model.
- We can see from the above visualization of common words between the categories and corresponding bar plot.

```
## [1] "Top common terms between the categories are:"
## Terms frequency
## 1: call 11399
## 2: patient 4499
## 3: paragraph 4319
## 4: back 4041
## 5: note 3967
## 6: fontphon 3605
```

Correlated terms when combined with coomon words and together removed from the data.

```
## [1] "Dimensions of data after removing corelated terms and top most common terms are:"
## [1] 57280 858
```

- Now that we have removed the most common terms and correlated terms (%85), we have our final features to predict the target variables. Our final no of variable are 858. We are going to form two master data sets each for predicting categories and sub-categories.

- Each master datasets is having dimensions as 57280 X 861, including the previous appointment variable and a target variable (categories or sub categories).
- First master dataset is going to have dimensions 57280* 861, including the previous appointment and sub_categories for classifying categories.
- Second master dataset is going to have 57280*861, including the previous appointment and categories for classifying sub_categories.

Sampling the master dataset and Forming train & test sets from the sample

- The function sampling(), from the master data set samples it and returns train and test sets required from the sample. We are using stratified sampling to preserve this ratio throughout sampling and splitting. caTools package is used to implement stratifiedsampling.

```
library(caTools)
categories = data_req$categories
sub_categories = data_req$sub_categories
previous_appointment = data_req$previous_appointment
master_data_cat= as.data.frame(cbind(df,previous_appointment,sub_categories,c
ategories))
master_data_sub = as.data.frame(cbind(df,previous_appointment,categories,sub_categories))
sampling <- function(master_data, set_seed = 123, samp.ratio= 0.075, train.ra
tio= 0.75){
 set.seed(seed = set_seed)
 samp_split = sample.split(master_data[,ncol(master_data)], samp.ratio)
 sample = subset(master_data, samp_split == T)
# training and testing
 smpl = sample.split(sample[,ncol(sample)], train.ratio)
 x_train = subset(sample, smpl == T )
 x_test = subset(sample, smpl == F )
 y_train = x_train[,ncol(x_train)]
 y_test = x_test[,ncol(x_test)]
 x_train[,ncol(x_train)] = NULL
 x_test[,ncol(x_test)] = NULL
 train_test = list(x_train,x_test,y_train,y_test)
 return(train_test)
}
```

```
## [1] "Dimensions of each train and test sets for classifying categories excluding target variable are:"
## [1] 3222 860
## [1] 1073 860
## [1] "Dimensions of each train and test sets for classifying categories excluding target variable are:"
## [1] 3223 860
## [1] 1072 860
```

- We can take different samples with different seeds to train the model.We are taking 10 samples of master data set and train our models using set.seed function.

Model building and tuning

- We have used h2o package to build random forest, gbm, deeplearning and xgboost models as it is really quick in training a model using h2o.
- Random forest, gbm, deep learning and xgboost models are flexible and can enhance the accuracy/performance of the weak algorithm to a better extent, at the expense of heavier computational resources required.
- Firstly I independently used **h2o gbm**, **h2o random forest**, **h2o deep learning** and **h2o xgboost models** and train and cross validated them.All of them gave **99%+ accuracy** for classifying category and **65%+ accuracy** in classifying subcategories. I came up with a strategy of stacking modelling.

How stacking works:

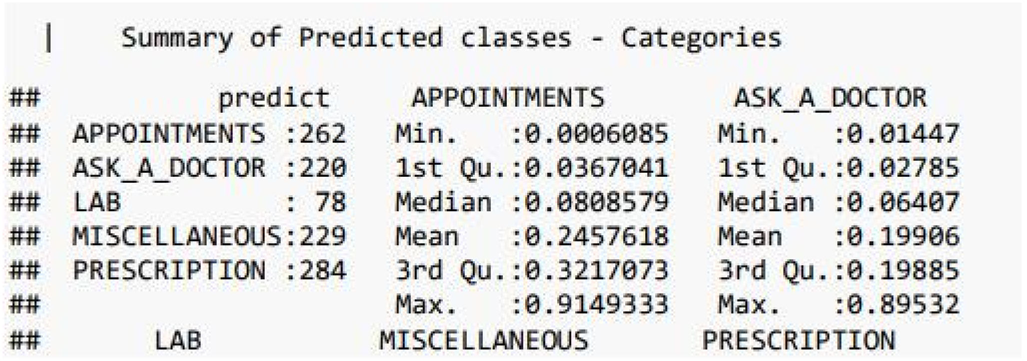
- Set up the ensemble.
- Specify a list of L base algorithms (with a specific set of model parameters).
- Specify a metalearning algorithm.
- Train the ensemble.
- Train each of the L base algorithms on the training set.
- Perform k-fold cross-validation on each of these learners and collect the cross-validated predicted values from each of the L algorithms.
- The N cross-validated predicted values from each of the L algorithms can be combined to form a new N x L matrix. This matrix, along wtih the original response vector, is called the “level-one” data. (N = number of rows in the training set.)
- Train the metalearning algorithm on the level-one data. The “ensemble model” consists of the L base learning models and the metalearning model, which can then be used to generate predictions on a test set.
- Predict on new data.
- To generate ensemble predictions, first generate predictions from the base learners.
- Feed those predictions into the metalearner to generate the ensemble prediction.

Predicting categories

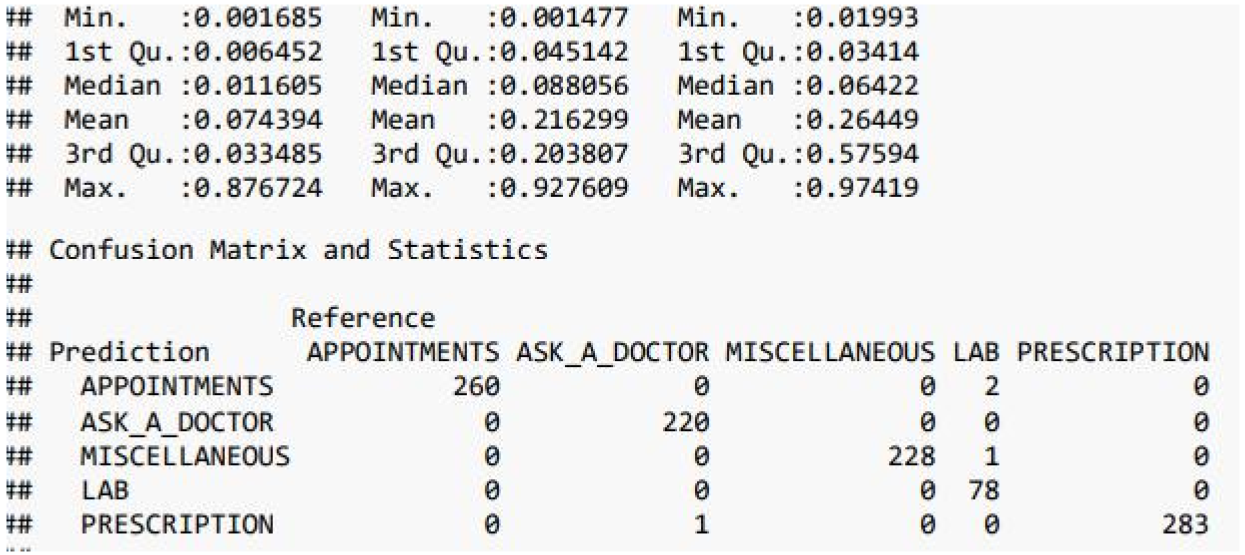
```
# Number of CV folds (to generate level-one data for stacking)
nfolds <- 5

# Train & Cross-validate a GBM
my_gbm_cat <- h2o.gbm(x = x_cat,y = y_cat,training_frame = h_train_cat,distribution = "bernoulli",max_depth = 3,
              min_rows = 2,learn_rate = 0.2,nfolds = nfolds,fold_assignment = "Modulo",
              keep_cross_validation_predictions = TRUE,seed = 1)

# Train & Cross-validate a RF
my_rf_cat <- h2o.randomForest(x = x_cat,y = y_cat,training_frame = h_train_cat,nfolds = nfolds,fold_assignment = "Modulo",
                    ntrees=1000,keep_cross_validation_predictions = TRUE,seed = 1)

# Train & Cross-validate a DNN
my_dl_cat <- h2o.deeplearning(x = x_cat,y = y_cat,training_frame = h_train_cat,l1 = 0.001,l2 = 0.001,
                    hidden = c(200, 200, 200),nfolds = nfolds,fold_assignment = "Modulo",
                    keep_cross_validation_predictions = TRUE,seed = 1)

# Train & Cross-validate a (shallow) XGB-GBM
my_xgb1_cat <- h2o.xgboost(x = x_cat,y = y_cat,training_frame = h_train_cat,distribution = "bernoulli",
                  ntrees = 50,max_depth = 3,min_rows = 2,learn_rate = 0.2,nfolds = nfolds,
                  fold_assignment = "Modulo",keep_cross_validation_predictions = TRUE,seed = 1)
```

```
# Train & Cross-validate another (deeper) XGB-GBM
my_xgb2_cat <- h2o.xgboost(x = x_cat, y = y_cat, training_frame = h_train_cat, distribution = "bernoulli",
               ntrees = 50,max_depth = 8,min_rows = 1,learn_rate = 0.1,sample_rate = 0.7,
               col_sample_rate = 0.9,nfolds = nfolds,fold_assignment = "Modulo",
               keep_cross_validation_predictions = TRUE,seed = 1)


# Train a stacked ensemble using the H2O and XGBoost models from above
base_models_cat <- list(my_gbm_cat@model_id, my_rf_cat@model_id, my_dl_cat@model_id,
             my_xgb1_cat@model_id, my_xgb2_cat@model_id)

ensemble_cat <- h2o.stackedEnsemble(x = x_cat, y = y_cat, training_frame = h_train_cat,
                     base_models = base_models_cat_)

pred_cat <- as.data.frame(h2o.predict(ensemble_cat, h_test_cat))
summary(pred_cat)
library(caret)
confusionMatrix(pred_cat$predict,y_test_cat)
h2o.varimp(ensemble_cat)
```

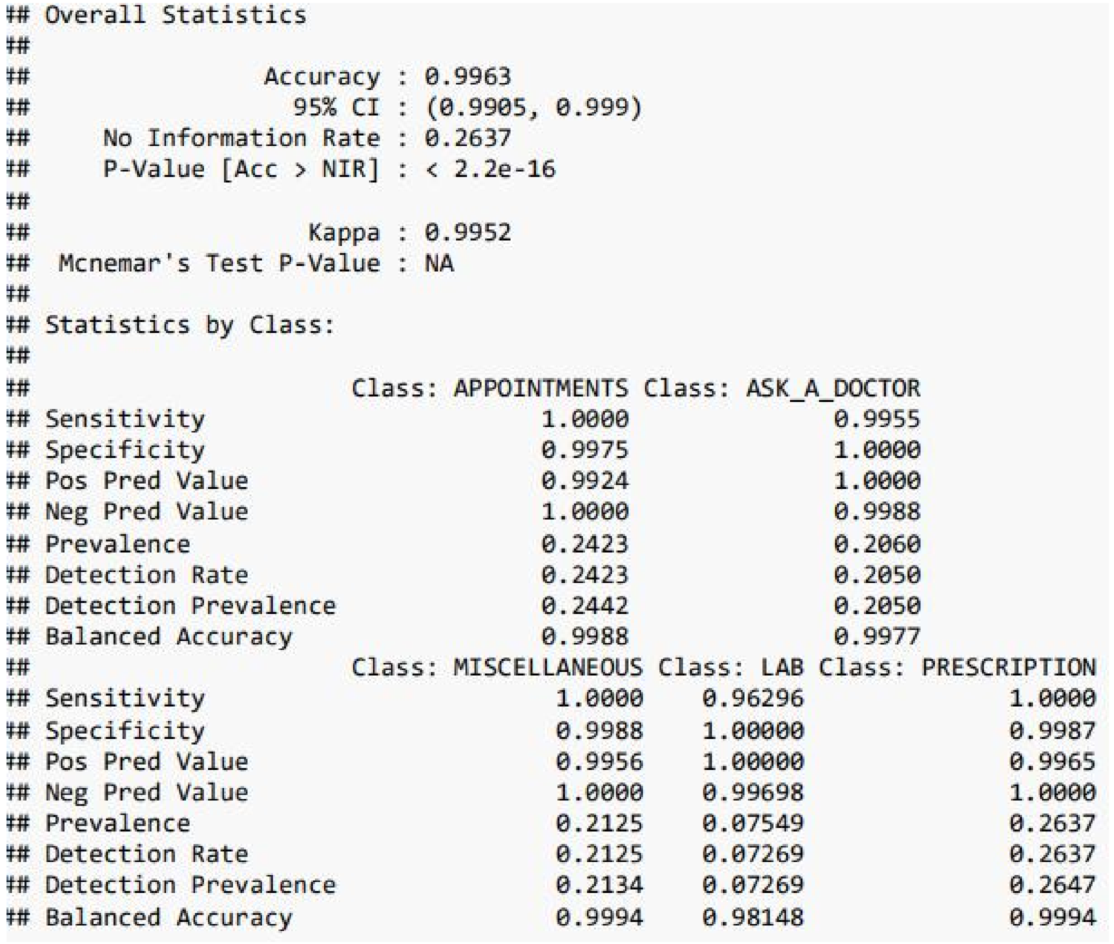
Confusion matrix for stacking-ensemble for predicting categories



```
  |    Summary of Predicted classes - Categories

##          predict      APPOINTMENTS        ASK_A_DOCTOR
##  APPOINTMENTS :262   Min.   :0.0006085   Min.   :0.01447
##  ASK_A_DOCTOR :220   1st Qu.:0.0367041   1st Qu.:0.02785
##  LAB          : 78   Median :0.0808579   Median :0.06407
##  MISCELLANEOUS:229   Mean   :0.2457618   Mean   :0.19906
##  PRESCRIPTION :284   3rd Qu.:0.3217073   3rd Qu.:0.19885
##                      Max.   :0.9149333   Max.   :0.89532
##       LAB           MISCELLANEOUS       PRESCRIPTION
```



```
##  Min.   :0.001685   Min.   :0.001477   Min.   :0.01993
##  1st Qu.:0.006452   1st Qu.:0.045142   1st Qu.:0.03414
##  Median :0.011605   Median :0.088056   Median :0.06422
##  Mean   :0.074394   Mean   :0.216299   Mean   :0.26449
##  3rd Qu.:0.033485   3rd Qu.:0.203807   3rd Qu.:0.57594
##  Max.   :0.876724   Max.   :0.927609   Max.   :0.97419

## Confusion Matrix and Statistics
##
##                Reference
## Prediction      APPOINTMENTS ASK_A_DOCTOR MISCELLANEOUS LAB PRESCRIPTION
##   APPOINTMENTS           260            0             0   2            0
##   ASK_A_DOCTOR             0          220             0   0            0
##   MISCELLANEOUS            0            0           228   1            0
##   LAB                      0            0             0  78            0
##   PRESCRIPTION             0            1             0   0          283
```

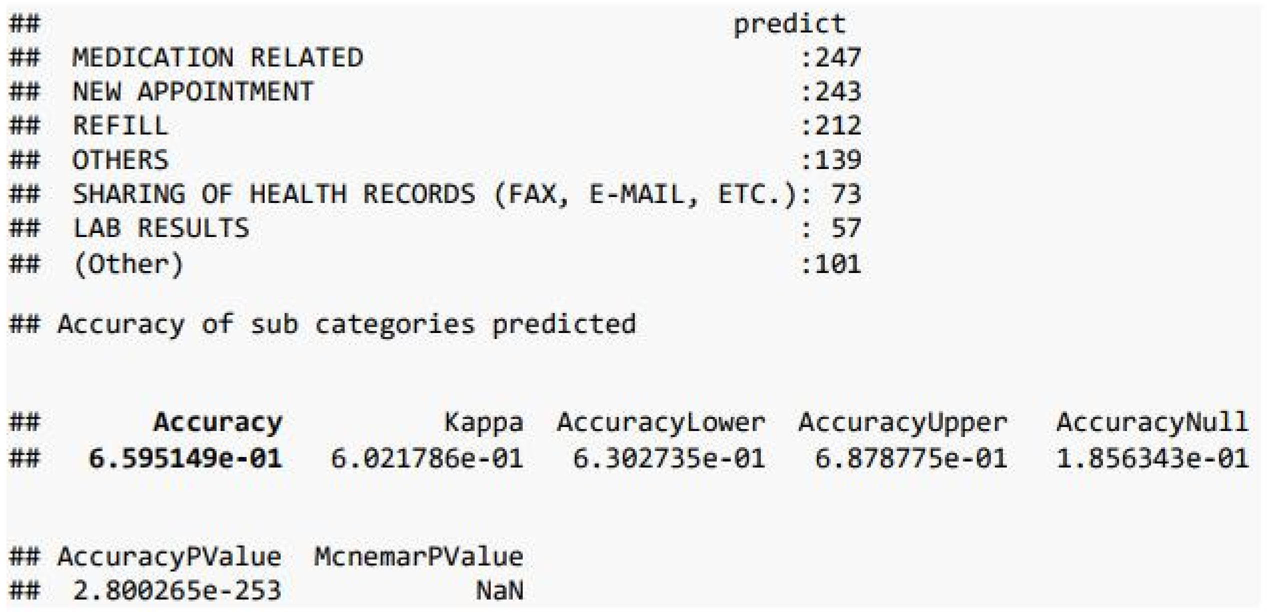


```
## Overall Statistics
##
##                Accuracy : 0.9963
##                  95% CI : (0.9905, 0.999)
##     No Information Rate : 0.2637
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9952
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: APPOINTMENTS Class: ASK_A_DOCTOR
## Sensitivity                       1.0000              0.9955
## Specificity                       0.9975              1.0000
## Pos Pred Value                    0.9924              1.0000
## Neg Pred Value                    1.0000              0.9988
## Prevalence                        0.2423              0.2060
## Detection Rate                    0.2423              0.2050
## Detection Prevalence              0.2442              0.2050
## Balanced Accuracy                 0.9988              0.9977
##                      Class: MISCELLANEOUS Class: LAB Class: PRESCRIPTION
## Sensitivity                        1.0000    0.96296              1.0000
## Specificity                        0.9988    1.00000              0.9987
## Pos Pred Value                     0.9956    1.00000              0.9965
## Neg Pred Value                     1.0000    0.99698              1.0000
## Prevalence                         0.2125    0.07549              0.2637
## Detection Rate                     0.2125    0.07269              0.2637
## Detection Prevalence               0.2134    0.07269              0.2647
## Balanced Accuracy                  0.9994    0.98148              0.9994
```

```
# Number of CV folds (to generate level-one data for stacking)
nfolds <- 5

# Train & Cross-validate a GBM
my_gbm_sub <- h2o.gbm(x = x_sub,y = y_sub,training_frame = h_train_sub,distribution = "bernoulli",max_depth = 3,
              min_rows = 2,learn_rate = 0.2,nfolds = nfolds,fold_assignment = "Modulo",
              keep_cross_validation_predictions = TRUE,seed = 1)

# Train & Cross-validate a RF
my_rf_sub <- h2o.randomForest(x = x_sub,y = y_sub,training_frame = h_train_sub,nfolds = nfolds,fold_assignment =
"Modulo",
                    ntrees=1000,keep_cross_validation_predictions = TRUE,seed = 1)

# Train & Cross-validate a DNN
my_dl_sub <- h2o.deeplearning(x = x_sub,y = y_sub,training_frame = h_train_sub,l1 = 0.001,l2 = 0.001,
                    hidden = c(200, 200, 200),nfolds = nfolds,fold_assignment = "Modulo",
                    keep_cross_validation_predictions = TRUE,seed = 1)

# Train & Cross-validate a (shallow) XGB-GBM
my_xgb1_sub <- h2o.xgboost(x = x_sub,y = y_sub,training_frame = h_train_sub,distribution = "bernoulli",
                  ntrees = 50,max_depth = 3,min_rows = 2,learn_rate = 0.2,nfolds = nfolds,
                  fold_assignment = "Modulo",keep_cross_validation_predictions = TRUE,seed = 1)
```

```
# Train a stacked ensemble using the H2O and XGBoost models from above
base_models_sub <- list(my_gbm_sub@model_id, my_rf_sub@model_id, my_dl_sub@model_id,
                my_xgb1_sub@model_id, my_xgb_sub2@model_id)

ensemble_sub <- h2o.stackedEnsemble(x = x_sub, y = y_sub, training_frame = h_train_sub,
                        base_models = base_models_sub)

###predicting the stacked model
pred_sub <- as.data.frame(h2o.predict(ensemble_sub, h_test_sub))
summary(pred_sub)
library(caret)
confusionMatrix(pred_sub$predict,y_test_sub)
h2o.varimp(ensemble_sub)
```

Confusion matrix



```
##                                             predict
##  MEDICATION RELATED                             :247
##  NEW APPOINTMENT                                :243
##  REFILL                                         :212
##  OTHERS                                         :139
##  SHARING OF HEALTH RECORDS (FAX, E-MAIL, ETC.): 73
##  LAB RESULTS                                    : 57
##  (Other)                                        :101

## Accuracy of sub categories predicted


##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   6.595149e-01   6.021786e-01   6.302735e-01   6.878775e-01   1.856343e-01


## AccuracyPValue  McnemarPValue
##   2.800265e-253           NaN
```

In h2o gbm, I used following metrics and where I train and cross- validated the gbm model.

- **keep cross validation predictions:** Specify whether to keep the predictions of the cross-validation models.
- **fold assignment:** Cross-validation fold assignment scheme.
- **ntrees:** number of trees.
- **nfolds:** Number of folds for cross-validation.
- **min rows:** The minimum number of rows to assign to the terminal nodes.

In h2o rf, I used following metrics and where I train and cross- validated the rf model.

- **ntrees:** number of trees.
- **nfolds:** Number of folds for cross-validation.
- **fold assignment:** Cross-validation fold assignment scheme.
- **keep cross validation predictions:** Specify whether to keep the predictions of the cross-validation models.

In h2o deeplearning, I used following metrics and where I train and cross- validated the rf model.

- **hidden:** Specifies the number and size of each hidden layer in the model.
- **keep cross validation predictions:** Specify whether to keep the predictions of the cross-validation models.
- **nfolds:** Number of folds for cross-validation.
- **Add l1& l2 regularization.**
- **fold assignment:** Cross-validation fold assignment scheme.
- Added hidden layer

In h2o xgboost, I used following metrics and where I train and cross-validated the xgboost model.

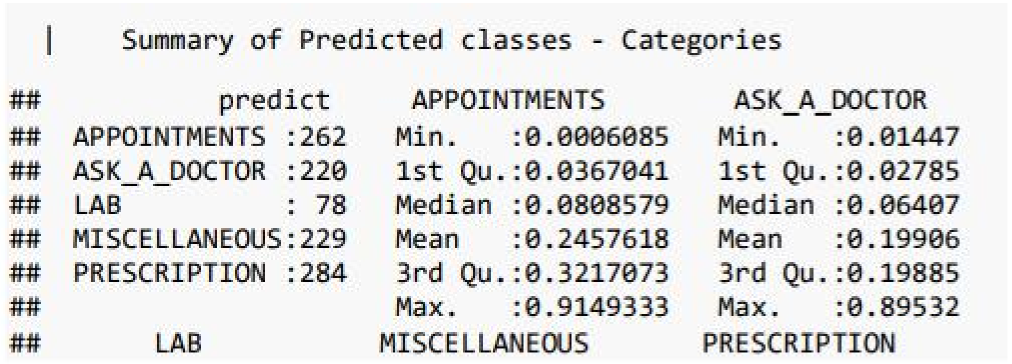
ntrees: number of trees.
nfolds: Number of folds for cross-validation.
fold assignment: Cross-validation fold assignment scheme.
keep cross validation predictions: Specify whether to keep the predictions of the cross-validation models.
learn_rate: Makes the model more robust by shrinking the weights on each step,

SVM model:

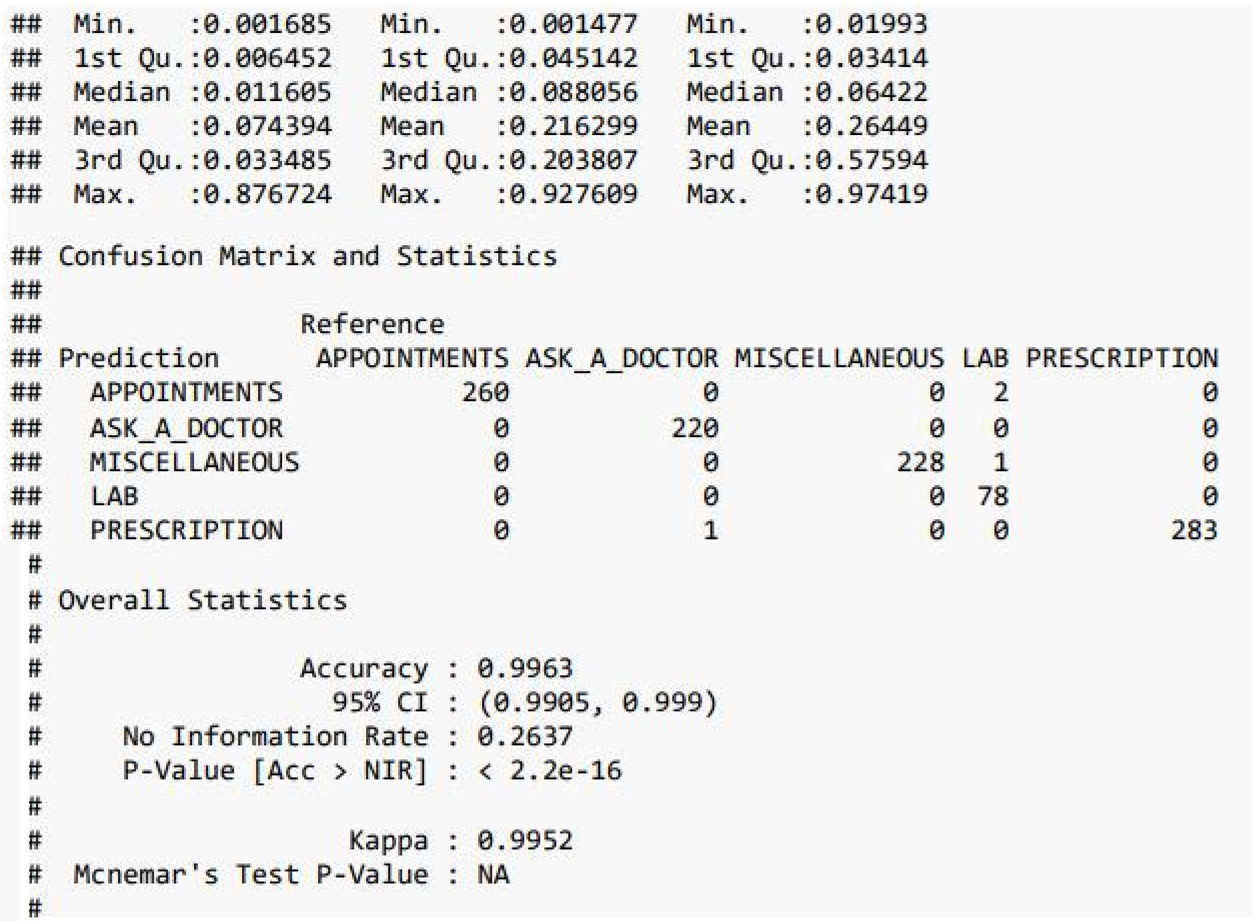
We are training with SVM not in H2o perspective as it uses a subset of training points in the decision function (called support vectors), so it is also memory efficient and also works really well with clear margin of separation.

SVM trained on 10 samples drawn with different seed gave a highest accuracy of 90% for classifying categories and 67% for classifying sub categories.
SVM performs well in case of non-linear separable data using kernel trick.
It works well in high dimensional space (i.e. large number of predictors)
It works well image classification.
It does not suffer multicollinearity problem

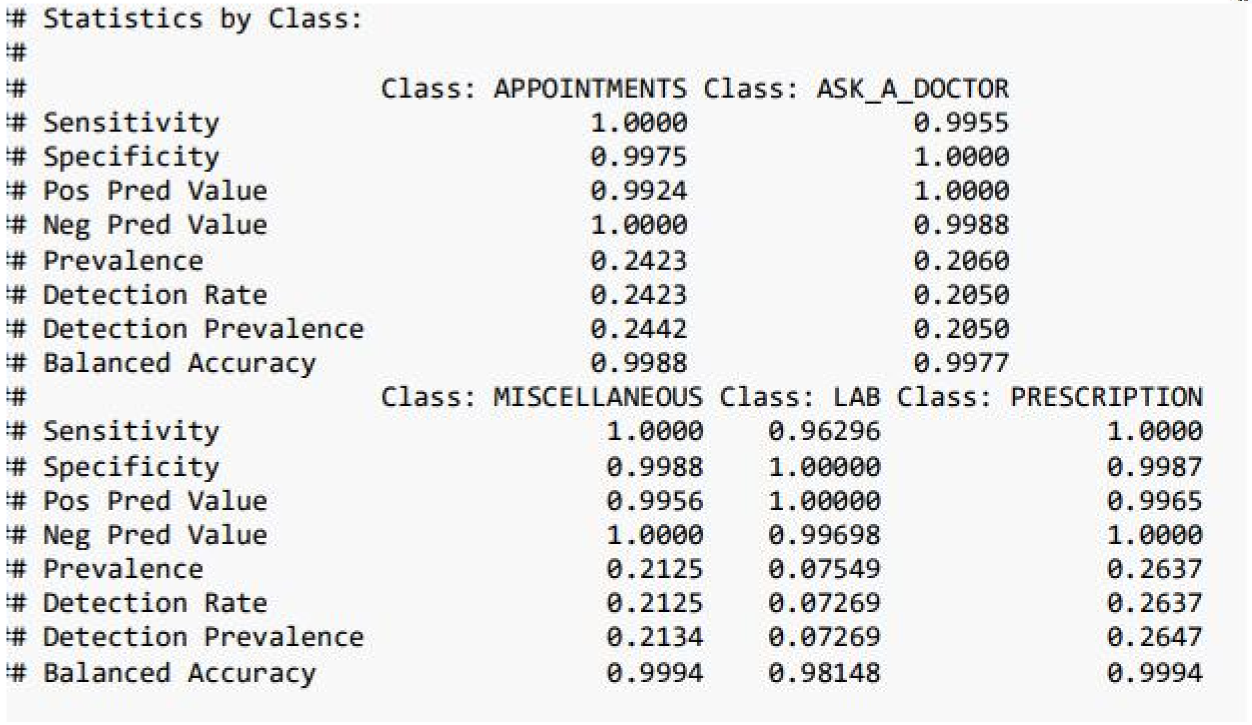
Confusion matrix for classifying categories for SVM model



```
 |    Summary of Predicted classes - Categories

##          predict      APPOINTMENTS       ASK_A_DOCTOR
##  APPOINTMENTS :262   Min.   :0.0006085   Min.   :0.01447
##  ASK_A_DOCTOR :220   1st Qu.:0.0367041   1st Qu.:0.02785
##  LAB          : 78   Median :0.0808579   Median :0.06407
##  MISCELLANEOUS:229   Mean   :0.2457618   Mean   :0.19906
##  PRESCRIPTION :284   3rd Qu.:0.3217073   3rd Qu.:0.19885
##                      Max.   :0.9149333   Max.   :0.89532
##       LAB             MISCELLANEOUS       PRESCRIPTION
```

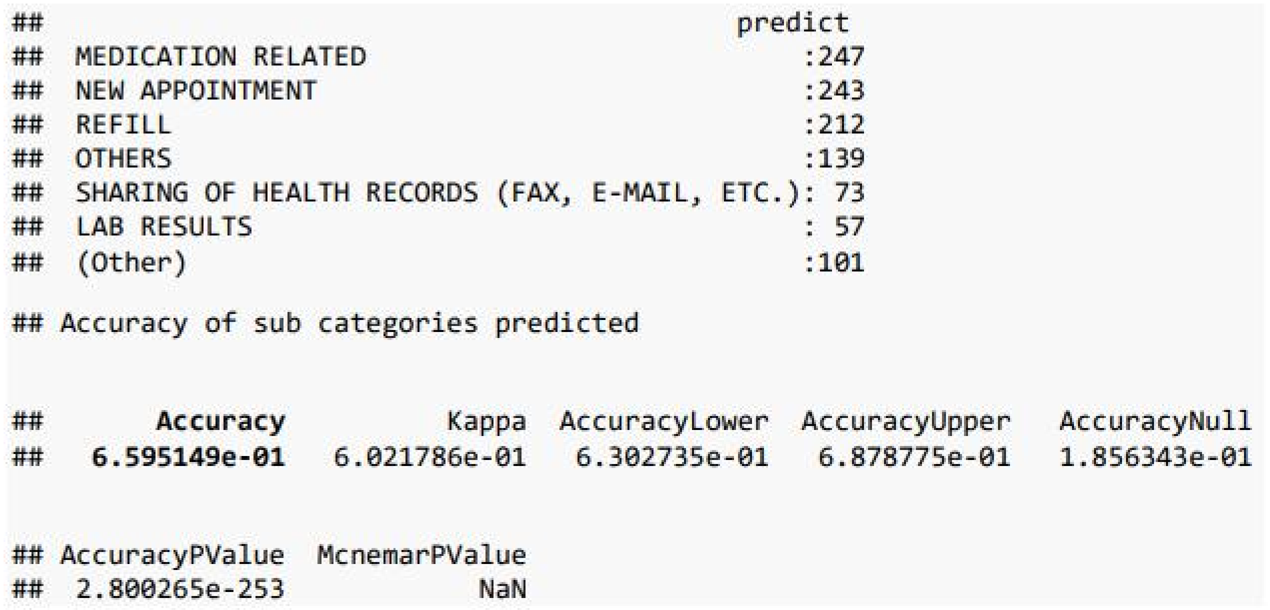


```
##  Min.   :0.001685   Min.   :0.001477   Min.   :0.01993
##  1st Qu.:0.006452   1st Qu.:0.045142   1st Qu.:0.03414
##  Median :0.011605   Median :0.088056   Median :0.06422
##  Mean   :0.074394   Mean   :0.216299   Mean   :0.26449
##  3rd Qu.:0.033485   3rd Qu.:0.203807   3rd Qu.:0.57594
##  Max.   :0.876724   Max.   :0.927609   Max.   :0.97419

## Confusion Matrix and Statistics
##
##                Reference
## Prediction      APPOINTMENTS ASK_A_DOCTOR MISCELLANEOUS LAB PRESCRIPTION
##   APPOINTMENTS           260            0             0   2            0
##   ASK_A_DOCTOR             0          220             0   0            0
##   MISCELLANEOUS            0            0           228   1            0
##   LAB                      0            0             0  78            0
##   PRESCRIPTION             0            1             0   0          283
#
# Overall Statistics
#
#                Accuracy : 0.9963
#                  95% CI : (0.9905, 0.999)
#     No Information Rate : 0.2637
#     P-Value [Acc > NIR] : < 2.2e-16
#
#                   Kappa : 0.9952
#  Mcnemar's Test P-Value : NA
#
```



```
# Statistics by Class:
#
#                      Class: APPOINTMENTS Class: ASK_A_DOCTOR
# Sensitivity                       1.0000              0.9955
# Specificity                       0.9975              1.0000
# Pos Pred Value                    0.9924              1.0000
# Neg Pred Value                    1.0000              0.9988
# Prevalence                        0.2423              0.2060
# Detection Rate                    0.2423              0.2050
# Detection Prevalence              0.2442              0.2050
# Balanced Accuracy                 0.9988              0.9977
#                      Class: MISCELLANEOUS Class: LAB Class: PRESCRIPTION
# Sensitivity                        1.0000    0.96296              1.0000
# Specificity                        0.9988    1.00000              0.9987
# Pos Pred Value                     0.9956    1.00000              0.9965
# Neg Pred Value                     1.0000    0.99698              1.0000
# Prevalence                         0.2125    0.07549              0.2637
# Detection Rate                     0.2125    0.07269              0.2637
# Detection Prevalence               0.2134    0.07269              0.2647
# Balanced Accuracy                  0.9994    0.98148              0.9994
```

Confusion matrix for classifying sub_categories for SVM model



```
##                                           predict
##  MEDICATION RELATED                           :247
##  NEW APPOINTMENT                              :243
##  REFILL                                       :212
##  OTHERS                                       :139
##  SHARING OF HEALTH RECORDS (FAX, E-MAIL, ETC.): 73
##  LAB RESULTS                                  : 57
##  (Other)                                      :101

## Accuracy of sub categories predicted


##      Accuracy         Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   6.595149e-01  6.021786e-01   6.302735e-01   6.878775e-01   1.856343e-01


## AccuracyPValue  McnemarPValue
##  2.800265e-253            NaN
```

Comparision between different models used:

The system time taken by svm was quite less than other models which were used with h2o api. But model performance wise, here h2o stacked model of gbm,rf,deep learning and xgboost models gave an acurracy of 99%+ where svm model gave 90% for classifying categories and for classifying both the stacked model and svm gave near to near performance.

I would like to freeze with my h2o stacked model as it performs really well on classifying categories but on a cost of system time

Comparision between different models used:

The system time taken by svm was quite less than other models which were used with h2o api. But model performance wise, here h2o stacked model of gbm,rf,deep learning and xgboost models gave an acurracy of 99%+ where svm model gave 90% for classifying categories and for classifying sub_categories both the stacked model and svm gave near to near performance of 65% accuracy.

I would like to freeze with my h2o stacked model as it performs really well on classifying categories but on a cost of computational time and power

ERROR METRICS

As there might be situations when a patient calling for emergency situations might be misclassified as the one with least priority value and a situation where the patient is calling for general advices is put on the top of the priority order for immediate attention from the Doctor. So, we consider both False Positive and False Negative and take them as whole as
misclassification error and minimize it. We are aiming to freeze the model which is giving least misclassification error with the minimal computational power and time. So, the model with highest accuracy is our model of Deployment.
So, my h2o stacked model is freezed with accuarcy of 99%+

Recommendations

- If we analyze the Document Term Matrices generated out of the SUMMARY and DATA column along with sub categories, there are many common terms between those 20 subcategories. This makes the problem more complex because the common terms misguide the models.
- If we try to remove the common terms between the sub categories, we are left with very few terms as we have only 57280 rows, which won't contribute much to the classification.
- So, I would suggest designing a model separately for classifying sub-categories with much more data. More data always yields the better results though it is complicated to clean it.
- I saw when I removed all the common terms from the categories, I was just left with 350 terms and where my model performance with that 350 terms was quite less. So when I removed top 15 common terms along with the correlated terms. My model performance reached 99% accuracy.
- This selection of top 15 common was completely random. So further reasearch of exacting removing how many common terms can be done as a feature engineering.
- When I made DTM from my corpus of both SUMMARY column and DATA column, the I wasn't able to convert them to matrix directly because of large columns. So I need to sparse term and reached a stage of 500 and 400 terms correspondly.
- Separate Visualizations of Summary, Data and combined data gave us a lots of insight of which word is often used for which category.
- Finally in the machine learning model, use of h2o library was must as simple random forest or even use of caret library was laking loads of computational time. I would recommend to always use h2o package for large data set as h2o package invites lots of machine learning models.