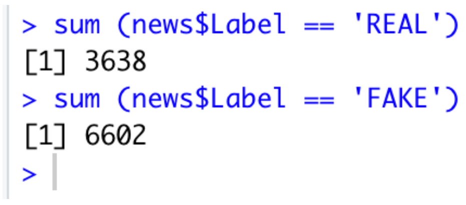
Robust Data Pipeline

In this project, there are five main steps, including data exploratory, data cleaning, dataset splitting, model training and model validation, used for building the holistic data pipeline. The process of data exploratoration can be a guide for data cleaning and getting model building well prepared. By splitting the dataset into training dataset and testing dataset, we can train the model on training dataset and test the model performance with testing dataset. The better performed model can be selected by comparing the evaluation metrics which indicates higher correlation rate.

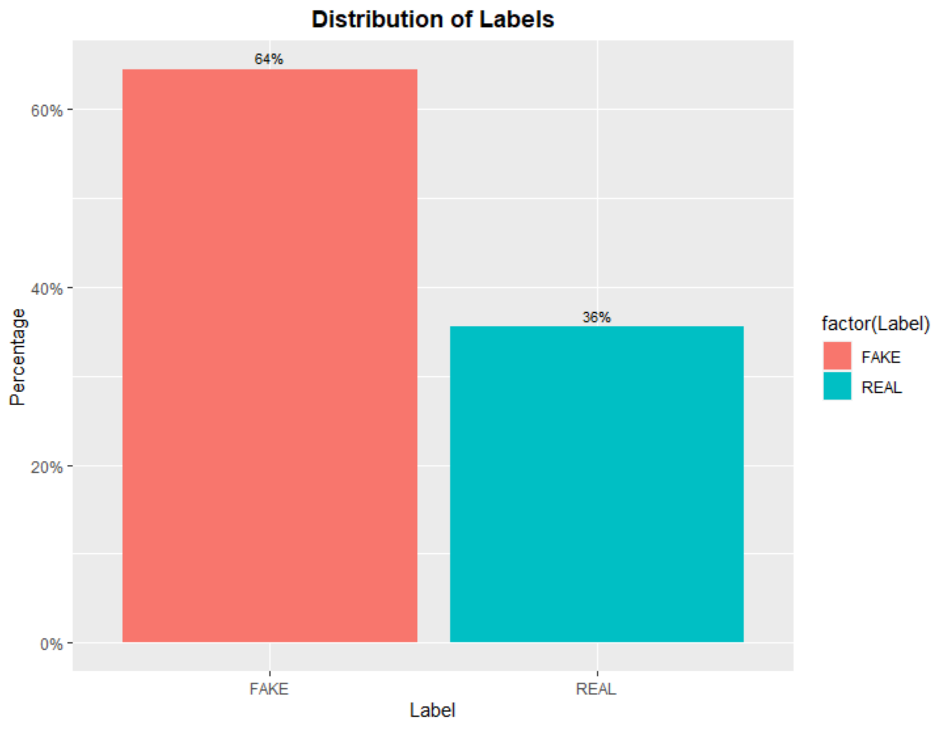
Exploratory Data Analysis (EDA)

In this part, cloud and histogram are used for briefly summaring data information. Firstly, I look at the proportion of the value of label. It can be observed that, of all data, the fake news accounts for 64% (6602 items), while real news accounts for 36% (3638 items).



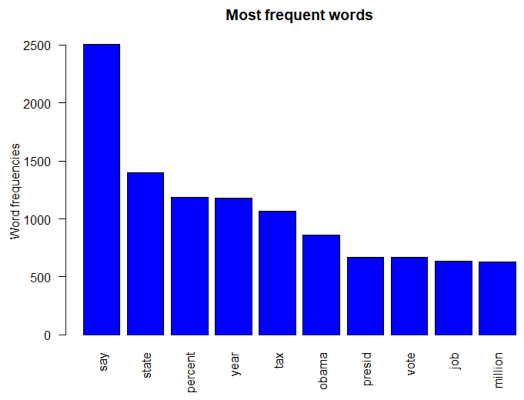
```
> sum (news$Label == 'REAL')
[1] 3638
> sum (news$Label == 'FAKE')
[1] 6602
>
```

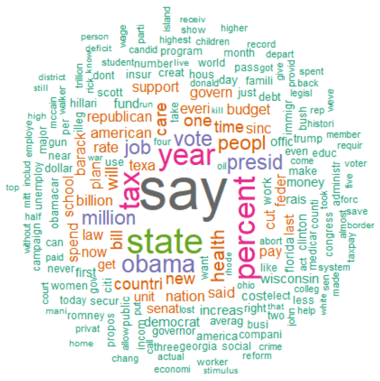
It indicates that the dataset is imbalanced, thus, we should consider resample method when building the model.

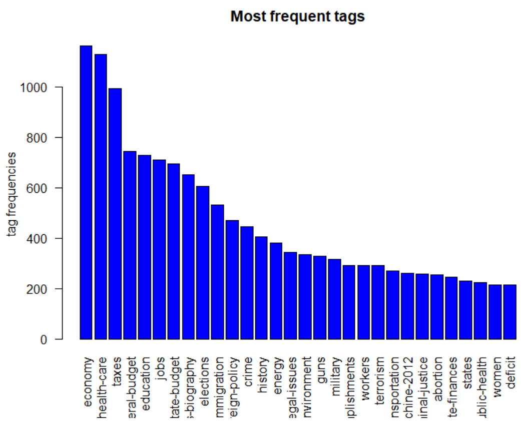


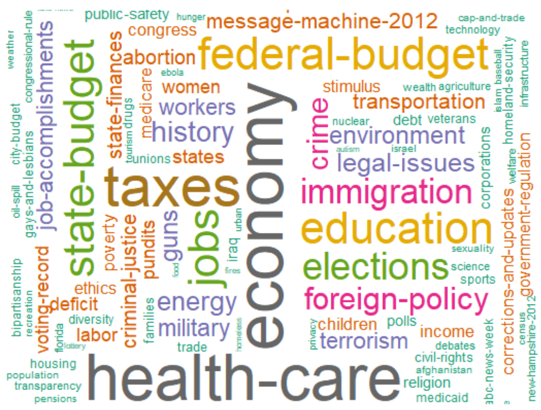
When modeling an imbalanced classification data set, the machine learning algorithm may be unstable, its prediction results may even be biased, and the prediction accuracy becomes misleading. In unbalanced data, algorithm cannot obtain enough information from a class with a small sample size to make accurate predictions. Therefore, machine learning algorithms are often required to be applied to balanced data sets.

Secondly, I create word cloud and calculate the term frequency. It shows the words and tags that most frequently appeared.







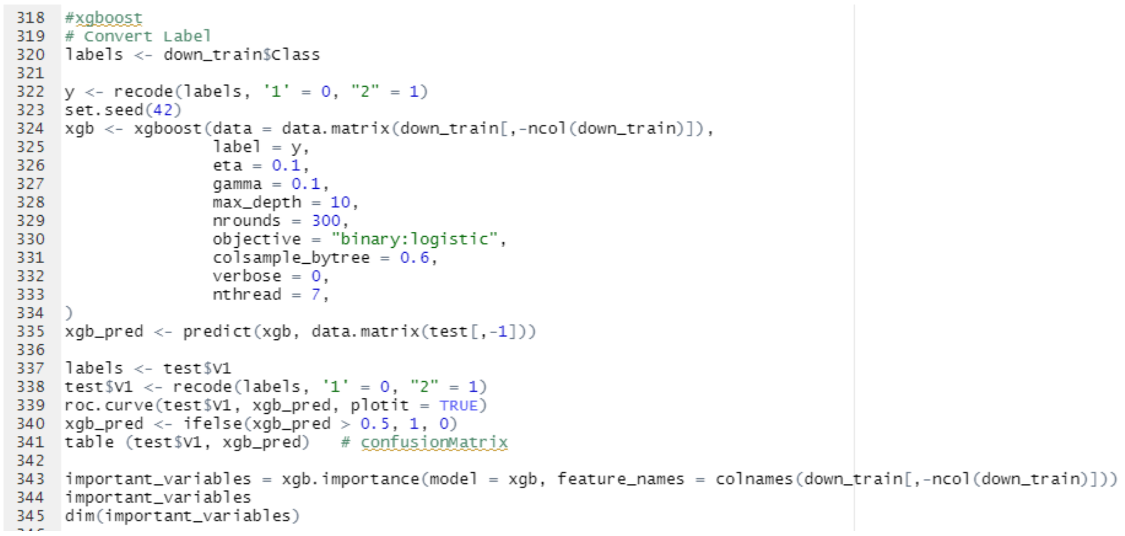


Related R Libraries

For data manipulation and data visualization, we need to use the R libraries “ggplot2” (for visualization), “dplyr”, and “tidyverse” (for data manipulation). For text mining, we need to use the R libraries “tm” (for texting mining), “SnowballC” (for text stemming), and “wordcloud” (for generating word cloud). For model building, we need to use the R libraries “pROC” (for plotting ROC curve on testing dataset), “xgboost” (for building model on training dataset), “caret” (for sampling) and “caTools” (for train and test split).

Predictive Analysis

To predict the label of news, I transform the unstructured text data into word-frequency matrix. After transforming text and text tag, I can use classification method to do predictions. The method could be random forest, xgboost, logistic regression, svm. Considering the volume of the data (more than 8,000 variables) and the number of dataset, here we use xgboost for modeling. After training the model, I predict a result on the test data and compare it with real label on test. The evaluation metrics would be AUC score.



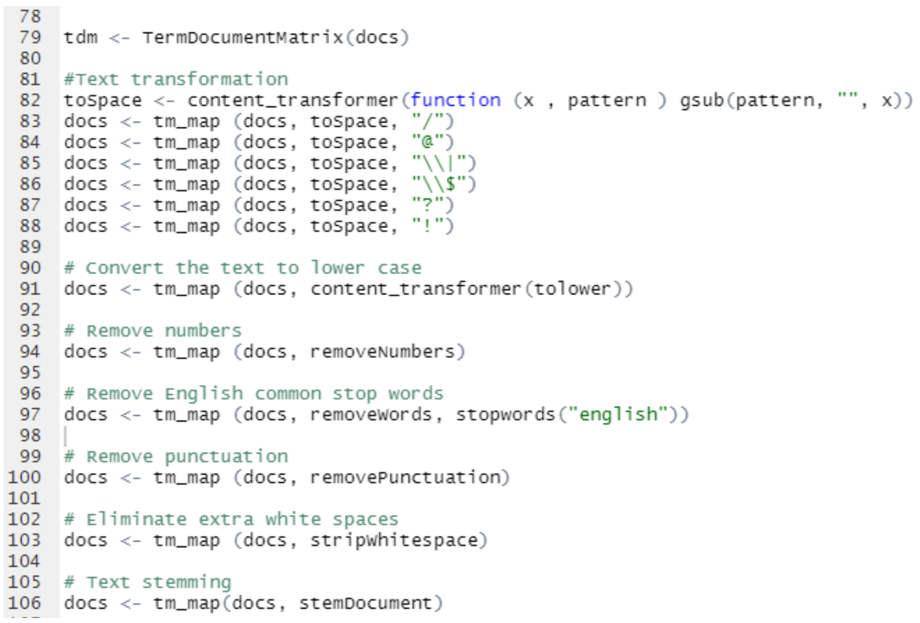
```r
#xgboost
# Convert Label
labels <- down_train$Class

y <- recode(labels, '1' = 0, "2" = 1)
set.seed(42)
xgb <- xgboost(data = data.matrix(down_train[,-ncol(down_train)]),
               label = y,
               eta = 0.1,
               gamma = 0.1,
               max_depth = 10,
               nrounds = 300,
               objective = "binary:logistic",
               colsample_bytree = 0.6,
               verbose = 0,
               nthread = 7,
)
xgb_pred <- predict(xgb, data.matrix(test[,-1]))

labels <- test$V1
test$V1 <- recode(labels, '1' = 0, "2" = 1)
roc.curve(test$V1, xgb_pred, plotit = TRUE)
xgb_pred <- ifelse(xgb_pred > 0.5, 1, 0)
table (test$V1, xgb_pred)   # confusionMatrix

important_variables = xgb.importance(model = xgb, feature_names = colnames(down_train[,-ncol(down_train)]))
important_variables
dim(important_variables)
```

Model Building

A clean dataset is prepared, which contains a word matrix and labels. All features should be numeric and the response variable should be factor with two levels (FAKE or REAL). Based on the trained model, we can predict the authenticity of the news.

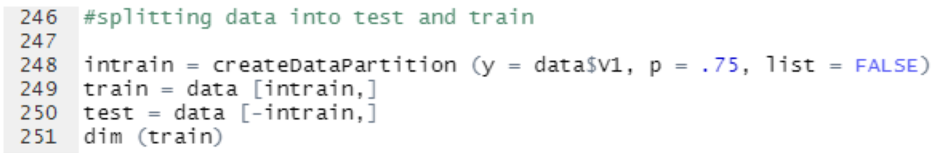
Process

For data cleaning, I treat the text data and the text tag data differently. For text data, I keep the useful information and remove the redundant punctuation as well as stop words, then transform the text data into word-frequency matrix. For tags, I split the tags into different columns and assign the appeared tag for 1 to indicates the tag vector.

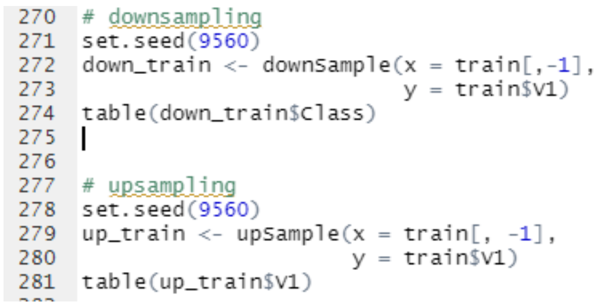


```r

tdm <- TermDocumentMatrix(docs)

#Text transformation
toSpace <- content_transformer(function (x , pattern ) gsub(pattern, "", x))
docs <- tm_map (docs, toSpace, "/")
docs <- tm_map (docs, toSpace, "@")
docs <- tm_map (docs, toSpace, "\\|")
docs <- tm_map (docs, toSpace, "\\$")
docs <- tm_map (docs, toSpace, "?")
docs <- tm_map (docs, toSpace, "!")

# Convert the text to lower case
docs <- tm_map (docs, content_transformer(tolower))

# Remove numbers
docs <- tm_map (docs, removeNumbers)

# Remove English common stop words
docs <- tm_map (docs, removeWords, stopwords("english"))

# Remove punctuation
docs <- tm_map (docs, removePunctuation)

# Eliminate extra white spaces
docs <- tm_map (docs, stripWhitespace)

# Text stemming
docs <- tm_map(docs, stemDocument)
```

The next step is to combine the text, tag and label, then split it into training and texting data. Since the data is imbalanced, down sampling, over sampling and non-resampling are apllied and corresponding modelling results are compared separately.



```r
#splitting data into test and train

intrain = createDataPartition (y = data$V1, p = .75, list = FALSE)
train = data [intrain,]
test = data [-intrain,]
dim (train)
```



```r
# downsampling
set.seed(9560)
down_train <- downSample(x = train[,-1],
                         y = train$V1)
table(down_train$Class)


# upsampling
set.seed(9560)
up_train <- upSample(x = train[, -1],
                     y = train$V1)
table(up_train$V1)
```

To select the most fitted model among these , I plot the ROC curve and calculate the AUC score. Also, we can compare them based on the confusion matrix.

Conclusion

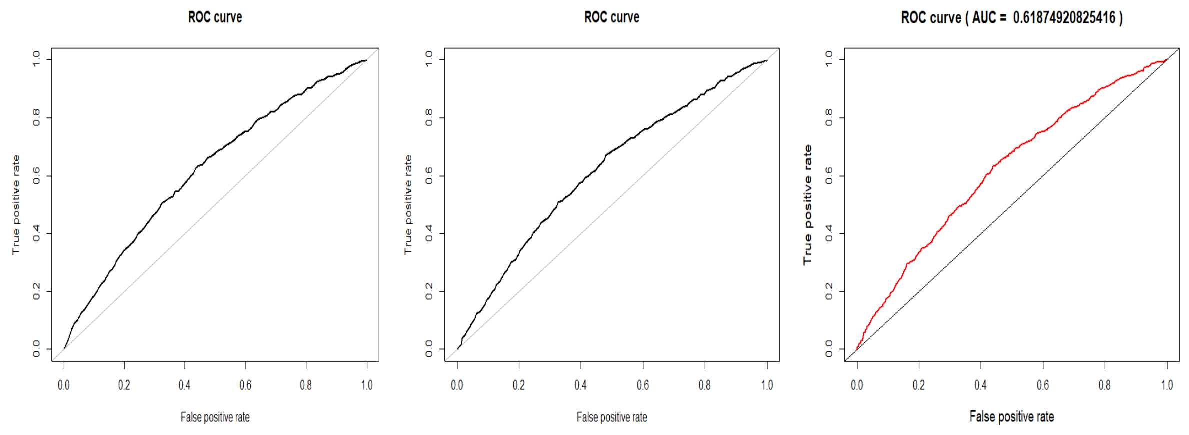


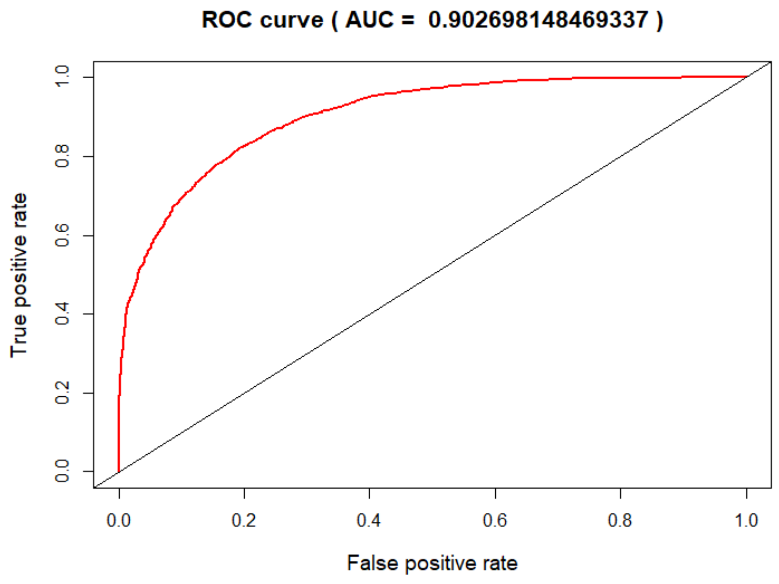
Figure ROC-curve of down sample (left), up sample (middle) and without resampling (right)

Based on the result, it can be found that the auc score of up sample is 62.1%, and that of model without sample is 61.9%. Down sample model has worst performance, with the auc score equals 61.6%. Then we look at the confusion matrix listed as below. Since we want to identify the FAKE news, we should maximize the ratio of recalling FAKE news (predict as 0 while in true it is 0). There are 1650 FAKE news and 909 REAL news in the testing data. It indicates that, with resampling, the recall rate is smaller but the precision is higher. It may because resampling can adjust the balance rate of data and give a more accurate split rule. However, since it lowers the proportion of FAKE news, it changes the distribution of data, making it harder to recall all FAKE news.

Confusion Matrix		
Down sample	Up sample	Without resampling
xgb_pred 0 1 0 944 706 1 364 545	xgb_pred.up 0 1 0 1114 536 1 450 459	xgb_pred.train 0 1 0 1450 200 1 716 193

0 indicates FAKE. 1 indicates REAL.
If we use the third model, though we protect us from many FAKE news, we may lose a lot of REAL news at the same time. Therefore, model choosing should depend on different purposes. The third model should be applied, if we are supposed to drop all FAKE news, as it can help us identify more FAKE news and drop them automatically. In other case, if we are supposed to determine if the news is real or fake, we should use model 2, which shows the highest auc score. This model can help us determine whether a new is FAKE or REAL based on its text and its text tags. It also solve the issue of new and unlabelled data.

The figure below is the ROC curve on the training dataset with auc score of 90.27%. the auc score is high in the training dataset, but perform unsatisfactorily on the test dataset. It means the model may have overfitting issue. Therefore, new training method may help to avoid this issue, such as cross validation.



Reflection and Recommendation

In real-world scenario, fake news detection model can decrease fake news spereading effiecncy. Also, it could significantly reduce the workload and improve the information authenticity on the platform. Media companies equipped with machine learning algorithm could automatically drop off the fake news rate thus lead to the companies with a reputation for trust and reliability.

Cloud based computation platforms, such as AWS, Azure and GCP, are strong toolds for building model with large scale dataset due to its extrodianryily highrt computing power and almost ‘infinite’ storage compare to PCs. Deploying the project on clouds could significantly decrease running time and save PC memories.