Sardar Patel Institute of Technology,Mumbai
Department of Electronics and Telecommunication Engineering
B.E. Sem-VII (2021-2022)
Data Analytics

Experiment:

Name: Sneha Ghuge **UID:** 2019110015 **BE ETRX** **DA LAB 6**

Aim: Separating spam from ham

Problem Statement :

Problem 1.1 – Loading the Dataset

Begin by loading the dataset emails.csv into a data frame called emails

● How many emails are in the dataset?
● How many of the emails are spam?
● Which word appears at the beginning of every email in the dataset?
Respond as a lower-case word with punctuation removed.
● Could a spam classifier potentially benefit from including the frequency of the word that appears in every email?
    ○ Yes -- the number of times the word appears might help us differentiate spam from ham.
    ○ No -- the word appears in every email so this variable would not help us differentiate spam from ham.
● The nchar() function counts the number of characters in a piece of text.

How many characters are in the longest email in the dataset (where longest is measured in terms of the maximum number of characters)?

Problem 2.1 -Preparing the Corpus
Follow the standard steps to build and pre-process the corpus:
1) Build a new corpus variable called corpus.
2) Using tm_map, convert the text to lowercase.
3) Using tm_map, remove all punctuation from the corpus.
4) Using tm_map, remove all English stop words from the corpus.
5) Using tm_map, stem the words in the corpus.
6) Build a document term matrix from the corpus, called dtm.
If the code length(stopwords("english")) does not return 174 for you, then please run the line of code in stopwords.tex (given in lab folder) file, which will store the standard stop words in a variable called sw. When removing stop words, use tm_map(corpus, removeWords, sw) instead of tm_map(corpus, removeWords, stopwords("english")).

How many terms are in dtm?
To obtain a more reasonable number of terms, limit dtm to contain terms appearing in at least 5% of documents, and store this result as spdtm (don't overwrite dtm, because we will use it in a later step of this homework). How many terms are in spdtm?

Problem 3.1 –
Building machine learning models First, create a variable called "emailsSparse" from "spdtm" using the command "emailsSparse = as.data.frame(as.matrix(spdtm))" and ensure it has legal variable names with "names(emailsSparse) = make.names(names(emailsSparse))". Then, copy the dependent variable from the original data frame called "emails" to "emailsSparse" using the command "emailsSparse$spam = as.factor(emails$spam)".

Next, set the random seed to 123 and use the sample.split function to split emailsSparse 70/30 into a training set called "train" and a testing set called "test". Make sure to perform this step on emailsSparse instead of emails.

Using the training set, train the following two machine learning models. The models should predict the dependent variable "spam", using all other available variables as independent variables. Please be patient, as these models may take a few minutes to train.

1) A CART model called spamCART, using the default parameters to train the model (don't worry about adding minbucket or cp or specifying losses for false positives and false negatives). Remember to add the argument method="class" since this is a binary classification problem.

2) A random forest model called spamRF, using the default parameters to train the model (don't worry about specifying mtry or ntree or nodesize or the losses for false positives and false negatives). Directly before training the random forest model, set the random seed to 123 (even though we've already done this earlier in the problem, it's important to set the seed right before training the model so we all obtain the same results. Keep in mind though that on certain operating systems, your results might still be slightly different).

Similar to logistic regression, CART and random forest can be told to give you predicted probabilities for classification problems. CART does this by returning the proportion of observations in the bucket of interest that fall into the relevant category, and random forest does this by returning the proportion of the trees that predict the relevant category. The following commands can be used to obtain predicted probabilities for the two fitted models on the training set:

```
predTrainCART = predict(spamCART)[,2]

predTrainRF = predict(spamRF, type="prob")[,2]
```

What is the training set accuracy of spamCART, using a threshold of 0.5 for predictions?

What is the training set accuracy of spamRF, using a threshold of 0.5 for predictions? (Remember that your answer might not match ours exactly, due to random behavior in the random forest algorithm on different operating systems.)

How many of the word stems "enron", "hou", "vinc", and "kaminski" appear in the CART tree?
What is the training set AUC of spamCART?
What is the training set AUC of spamRF?

Problem 4.1 – Evaluating on the Test Set

Obtain predicted probabilities for the testing set for each of the models, again ensuring that probabilities instead of classes are obtained. This can be achieved with the following code:

predTestCART = predict(spamCART, newdata=test)[,2]

predTestRF = predict(spamRF, newdata=test, type="prob")[,2]

What is the testing set accuracy of spamCART, using a threshold of 0.5 for predictions?

What is the testing set AUC of spamCART?

What is the testing set accuracy of spamRF, using a threshold of 0.5 for predictions?

What is the testing set AUC of spamRF?

Which model had the best testing set performance, in terms of accuracy and AUC?

-CART or Random Forest Compare three models and show result

CODE & OUTPUT:

IMPORTING THE REQUIRED LIBRARIES AND DATASET

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import missingno
```

```
email = pd.read_csv('emails.csv')
email
```

	text	spam
0	Subject: naturally irresistible your corporate...	1
1	Subject: the stock trading gunslinger fanny i...	1
2	Subject: unbelievable new homes made easy im ...	1
3	Subject: 4 color printing special request add...	1
4	Subject: do not have money , get software cds ...	1
...	...	...
5723	Subject: re : research and development charges...	0
5724	Subject: re : receipts from visit jim , than...	0

```
email.head()
```

	text	spam
0	Subject: naturally irresistible your corporate...	1
1	Subject: the stock trading gunslinger fanny i...	1
2	Subject: unbelievable new homes made easy im ...	1
3	Subject: 4 color printing special request add...	1
4	Subject: do not have money , get software cds ...	1

#1) How many emails are in the dataset?

```
print(f'Length of Dataset before dropping duplicate rows {len(email)}')
```

```
Length of Dataset before dropping duplicate rows 5728
```

#2) How many spam emails are in the dataset?

```
print(f'No of Spam Messages {len(email[email.spam == 1])}')
```

```
No of Spam Messages 1368
```

REMOVING REPEATED VALUES

```
email = email.drop_duplicates()
email.head()
```

	text	spam
0	Subject: naturally irresistible your corporate...	1
1	Subject: the stock trading gunslinger fanny i...	1
2	Subject: unbelievable new homes made easy im ...	1
3	Subject: 4 color printing special request add...	1
4	Subject: do not have money , get software cds ...	1

```
print(f'Length of Dataset after dropping duplicate rows {len(email)}')
```

```
Length of Dataset after dropping duplicate rows 5695
```

```
print(f'No of Spam Messages after dropping any duplicates {len(email[email.spam == 1])}')
```

```
No of Spam Messages after dropping any duplicates 1368
```

* Removing the punctuation marks from data

A List of all punctuation marks present

```
import string
punctuations = list(string.punctuation)
print(punctuations)
```

```
['!', '"', '#', '$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', ':', ';', '<', '=', '>', '?', '@', '[', '\\', ']',
'^', '_', '`', '{', '|', '}', '~']
```

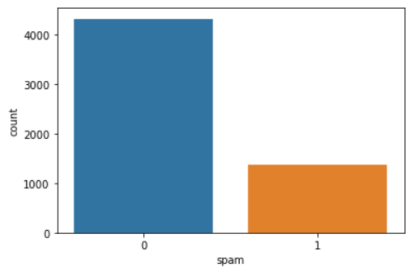
NLTK RegexTokenzier to select alphanumeric words only
Tokenization is used in natural language processing to split paragraphs and sentences into smaller units that can be more easily assigned meaning

5) The nchar() function counts the number of characters in a piece of text. Ho many characters are in the longest email in the dataset (where longest is measured in terms of the maximum number of characters)?

```
sns.countplot(email['spam'])
```

```
C:\Users\91744\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keywo
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explici
yword will result in an error or misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:xlabel='spam', ylabel='count'>
```



Now we will add a new column that will store the length of every corresponding text

```
email['text_length'] = email['text'].apply(len)
email.head()
```

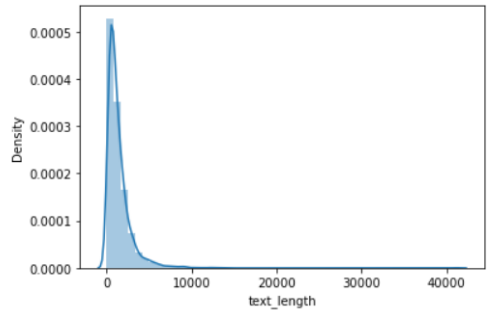
```
C:\Users\91744\AppData\Local\Temp\ipykernel_7984\1259697070.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
rsus-a-copy
  email['text_length'] = email['text'].apply(len)
```

	text	spam	text_length
0	Subject naturally irresistible your corporate ...	1	1403
1	Subject the stock trading gunslinger fanny is ...	1	593
2	Subject unbelievable new homes made easy im wa...	1	424
3	Subject 4 color printing special request addit...	1	444
4	Subject do not have money get software cds fro...	1	207

```
sns.distplot(email[email['spam'] == 0]['text_length'])
```

```
C:\Users\91744\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a depr
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level functio
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

```
<AxesSubplot:xlabel='text_length', ylabel='Density'>
```



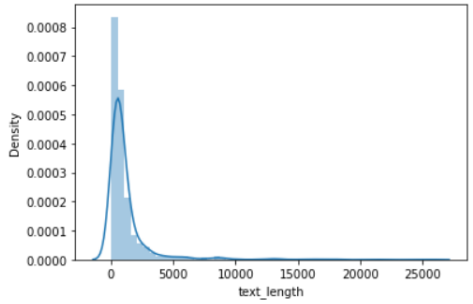

```
email[email['spam'] == 0]['text_length'].median()
```

```
963.0
```

```
sns.distplot(email[email['spam'] == 1]['text_length'])
```

```
C:\Users\91744\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

```
<AxesSubplot:xlabel='text_length', ylabel='Density'>
```



```
email[email['spam'] == 1]['text_length'].median()
```

```
624.0
```

After the above calculations we can see that the median amount of words in spam is much lesser than ham emails

```
maxlen = email["text_length"].max()
maxindex = email["text_length"].idxmax()
print(maxlen,maxindex)
email["spam"][2650]
```

```
41301 2650
```

```
0
```

```
email["text"][2650]
```

```
'Subject from the enron india newsdesk april 27 th newsclips fyi news articles from indian press forwarded by sandeep ko
nron _ development on 04 27 2001 08 24 am nikita varma 04 27 2001 07 51 am to nikita varma enron _ development enron _ d
pment cc bcc sandeep kohli enron _ development subject from the enron india newsdesk april 27 th newsclips friday apr 27
http www economictimes com today cmo 3 htm dpc board empowers md to cancel mseb contract friday apr 27 2001 http www eco
times com today 27 compl 1 htm mseb pays rs 134 cr under protest to dpc friday april 27 001 http www businessstandard co
ay economy 4 asp menu 3 enron india md authorised to terminate ppa friday april 27 2001 http www financialexpress com fe
0427 topl html foreign lenders slam brakes on disbursements to dpc sanjay jog raghu mohan global banks comfortable with
pull out friday april 27 2001 http www indian express com ie 20010427 nat 23 html enron dabhol chief gets powers to end
```

Build a new corpus variable called corpus

```
corpus = pd.DataFrame()
corpus['text'] = email['text']
corpus['spam'] = email['spam']
corpus.head()
```

	text	spam
0	Subject naturally irresistible your corporate ...	1
1	Subject the stock trading gunslinger fanny is ...	1
2	Subject unbelievable new homes made easy im wa...	1
3	Subject 4 color printing special request addit...	1
4	Subject do not have money get software cds fro...	1

Using tm_map, convert the text to lowercase. (lambda function in python)

```
corpus['text'] = corpus['text'].apply(lambda x: x.lower())
```

Remove all punctuation from the corpus.

```
corpus['text'] = corpus['text'].map(remove_punctuations)
```

Remove all English stop words from the corpus.

```
import nltk
nltk.download("stopwords")
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\91744\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
True
```

```
from nltk.corpus import stopwords
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'y
ourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself',
'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those',
'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'a
n', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'b
```

```
# creating the stopwords dictionary object
from collections import Counter
stop_words = stopwords.words('english')
stopwords_dict = Counter(stop_words)
```

```
sentence = 'Start Studying for your exams'
modified_sentence = ' '.join([word for word in sentence.split() if word not in stopwords_dict])
modified_sentence
```

```
'Start Studying exams'
```

```
def remove_stopwords_from_sentence(sentence):
    modified_sentence = ' '.join([word for word in sentence.split() if word not in stopwords_dict])
    return modified_sentence

# remove stopwords from the sentence
corpus['text'] = corpus['text'].apply(remove_stopwords_from_sentence)
```

```
# finally
corpus.head()
```

	text	spam
0	subject naturally irresistible corporate ident...	1
1	subject stock trading gunslinger fanny merrill...	1
2	subject unbelievable new homes made easy im wa...	1

Stem the words in the corpus

Creating a stemmer object

```
from nltk.stem import PorterStemmer
ps = PorterStemmer()
```

Stem the words

```
def stem_words_in_sentence(sentence):
    modified_sentence = ' '.join([ps.stem(word) for word in sentence.split()])
    return modified_sentence

corpus['text'] = corpus['text'].apply(stem_words_in_sentence)
corpus.head()
```

	text	spam
0	subject natur irresist corpor ident lt realli ...	1
1	subject stock trade gunsl fanni merril muzo co...	1
2	subject unbeliev new home made easi im want sh...	1
3	subject 4 color print special request addit in...	1
4	subject money get softwar cd softwar compat gr...	1

Build a DTM (Document Term Matrix) from the corpus

```
from sklearn.feature_extraction.text import CountVectorizer
vec = CountVectorizer()
dtm  = vec.fit_transform(corpus['text'])
```

For analytic simplicity, we will focus on the first five text fields and the top 25 words in this demonstration.

```
# Select the first five rows from the data set
td = pd.DataFrame(dtm.todense()).iloc[:5]
td.columns = vec.get_feature_names()
term_document_matrix = td.T
term_document_matrix.columns = ['Doc '+str(i) for i in range(1, 6)]
term_document_matrix['total_count'] = term_document_matrix.sum(axis=1)

# Top 25 words
term_document_matrix = term_document_matrix.sort_values(by ='total_count',ascending=False)[:25]

# Print the first 10 rows
print(term_document_matrix.drop(columns=['total_count']).head(10))
```

```
         Doc 1  Doc 2  Doc 3  Doc 4  Doc 5
subject      1      1      1      1      1
click        0      0      0      4      0
logo         4      0      0      0      0
market       4      0      0      0      0
compani      3      0      0      0      0
tri          0      3      0      0      0
inform       1      0      0      2      0
format       1      0      0      2      0
form         0      0      1      2      0
order        1      0      0      2      0
```

```
: dtm.shape
```

```
: (5695, 29222)
```

How many terms are in dtm? To obtain a more reasonable number of terms, limit dtm to contain terms appearing in at least 5% of documents, and store this result as spdtm (don't overwrite dtm, because we will use it in a later step of this homework).

```
print(f'No of terms in the document term matrix is {dtm.shape[1]}')
```

No of terms in the document term matrix is 29222

How many terms are in spdtm?

```
five_percent_vectorizer = CountVectorizer(min_df = 0.05)
five_percent_document_term_matrix = five_percent_vectorizer.fit_transform(corpus['text'])


print(f'No of terms in the the five percent document term matrix is
{five_percent_document_term_matrix.shape[1]}')
```

No of terms in the the five percent document term matrix is 368

Building Machine Learning Models

```
# setting the seed variable as specified in the question
SEED = 123
np.random.seed(SEED)
```

```
# creating the train and test data from document term matrix by splitting the train and test in the ratio 70:30
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(dtm, corpus['spam'], test_size=0.3, random_state=SEED)
```

Implementing the CART Model

```
from sklearn import tree
clf = tree.DecisionTreeClassifier()
clf.fit(X_train, y_train)
```

```
DecisionTreeClassifier()
```

What is the training set accuracy of spamCART, using a threshold of 0.5 for predictions?

```
# finding the training data accuracy of the model
from sklearn.metrics import accuracy_score
print(accuracy_score(clf.predict(X_test), y_test))
```

```
0.9566998244587478
```

```
# plotting the confusion matrix
from sklearn.metrics import confusion_matrix
print(confusion_matrix(clf.predict(X_test), y_test))
```

```
[[1267   32]
 [  42  368]]
```

```
# printing a more detailed analysis of the CART model
from sklearn.metrics import classification_report
print(classification_report(clf.predict(X_test), y_test))
```

What is the training set AUC of spamCART?

```
from sklearn.metrics import roc_auc_score
print(roc_auc_score(clf.predict(X_test), y_test))
```

```
0.9364633207532999
```

Implementing the RANDOM FOREST Model

```
from sklearn.ensemble import RandomForestClassifier
random_forest_classifier = RandomForestClassifier()
random_forest_classifier.fit(X_train, y_train)
```

```
RandomForestClassifier()
```

What is the training set accuracy of spamRF, using a threshold of 0.5 for predictions?

```
# finding the training data accuracy of the model
from sklearn.metrics import accuracy_score
print(accuracy_score(random_forest_classifier.predict(X_test), y_test))
```

```
0.9806904622586308
```

```
# plotting the confusion matrix
from sklearn.metrics import confusion_matrix
print(confusion_matrix(random_forest_classifier.predict(X_test), y_test))
```

```
[[1304   28]
 [   5  372]]
```

```
# printing a more detailed analysis of the rf model
from sklearn.metrics import classification_report
print(classification_report(random_forest_classifier.predict(X_test), y_test))
```

What is the training set AUC of spamRF?

```
from sklearn.metrics import roc_auc_score
print(roc_auc_score(random_forest_classifier.predict(X_test), y_test))
```

```
0.9828581897547416
```

Conclusion:

1. Successfully implemented spam and ham email classification using CART and Random Forest
2. Studying in detail the confusion matrix we can infer that the precision and recall score are very good which means that very less number of negative samples have been incorrectly classified. Recall is independent of the number of negative sample classifications. Further, if the model classifies all positive samples as positive, then Recall will be 1. The recall measures the model's ability to detect positive samples. The higher value of recall for our CART model suggests that more positive samples are detected.
3. Studying in detail the confusion matrix we can infer that the precision and recall score are very good compared to the CART Model, we have a precision value of 1 or 100% which shows that all positive samples are classified as positive, and there is not any Negative sample that is incorrectly classified.