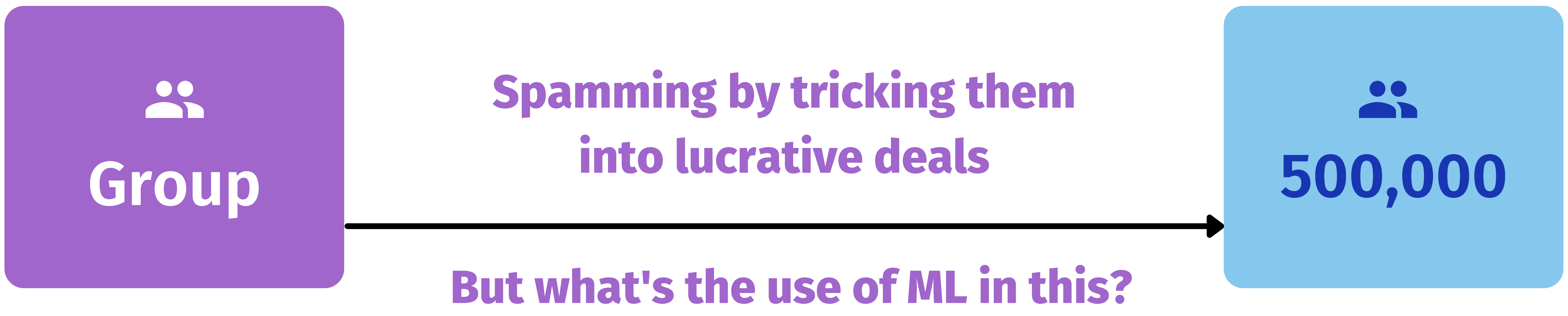
Comparative study on best ML algorithm for SPAM message classification.

Yash Raj Ojha
ENG19CS0367

Understanding the topic

We first need to understand what is Spam and what it is widely used for.
Spam messages are messages sent to a large group of recipients without their prior consent, typically advertising for goods and services or business opportunities.
In the recent period, the percentage of scam messages amongst has increased sharply.



Diving into the data

With the use of ML and DL, we can build a spam message classifier/filter which will be a step towards building a tool for scam message identification and early scam detection.

From various datasets available I could conclude that for easier identification, the messages are either classified as 'spam' or 'ham', so what are these?

Ham

"Ham" is e-mail that is not Spam. In other words, "non-spam", or "good mail". It should be considered a shorter, snappier synonym for "non-spam".

Spam

It is an irrelevant or unsolicited message, sent to a large number of users, for advertising, phishing, spreading malware.

Analysing the dataset

Using different visualizations to understand and modify the dataset

Image 1

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN
5	spam	FreeMsg Hey there darling it's been 3 week's n...	NaN	NaN	NaN
6	ham	Even my brother is not like to speak with me. ...	NaN	NaN	NaN
7	ham	As per your request 'Melle Melle (Oru Minnamin...	NaN	NaN	NaN
8	spam	WINNER!! As a valued network customer you have...	NaN	NaN	NaN
9	spam	Had your mobile 11 months or more? U R entitle...	NaN	NaN	NaN

Image 2

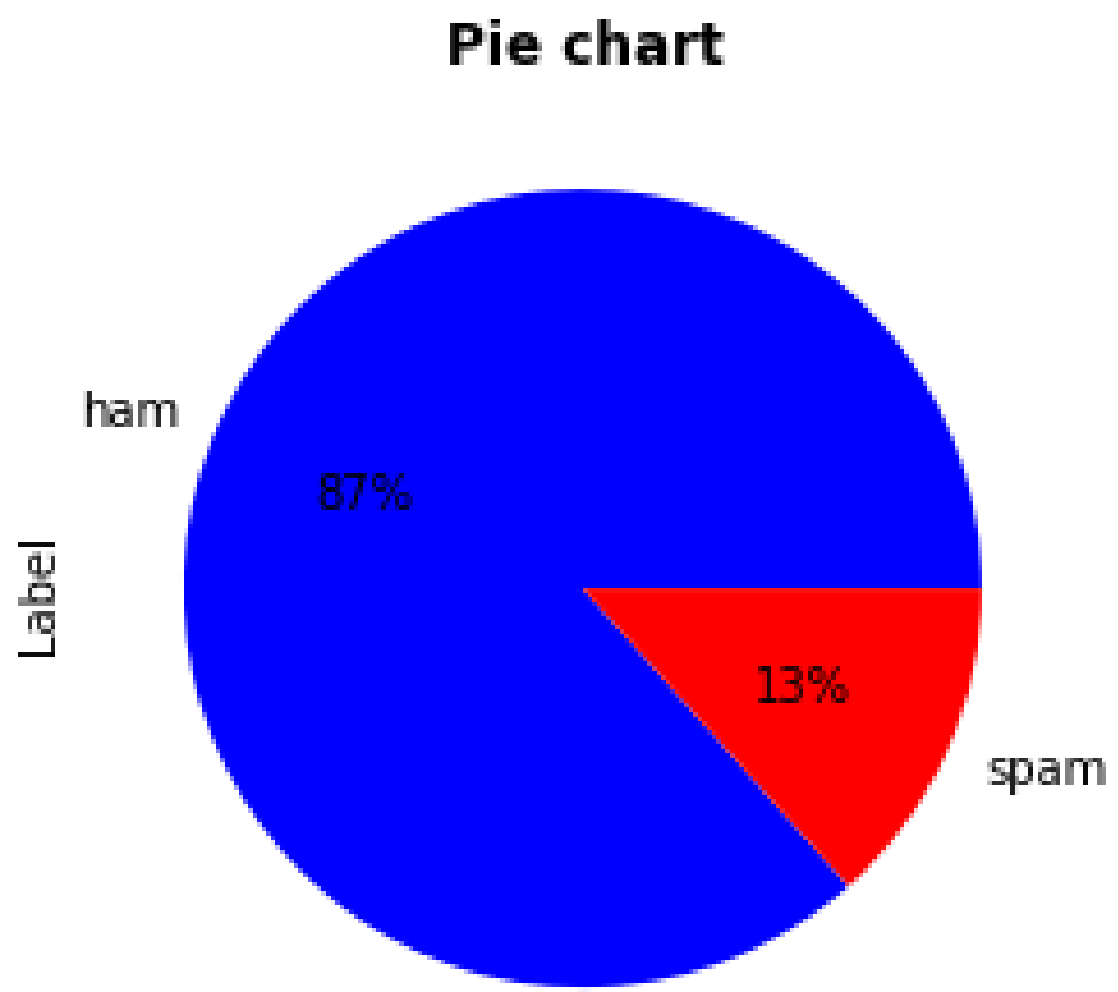
	Label	Message	length
0	ham	Go until jurong point, crazy.. Available only ...	111
1	ham	Ok lar... Joking wif u oni...	29
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	155
3	ham	U dun say so early hor... U c already then say...	49
4	ham	Nah I don't think he goes to usf, he lives aro...	61
5	spam	FreeMsg Hey there darling it's been 3 week's n...	148
6	ham	Even my brother is not like to speak with me. ...	77
7	ham	As per your request 'Melle Melle (Oru Minnamin...	160
8	spam	WINNER!! As a valued network customer you have...	158
9	spam	Had your mobile 11 months or more? U R entitle...	154

Converting CSV file to a dataframe using pandas and displaying the values.

Removing unnecessary columns and introducing a new column length of messages.

Analysing the dataset

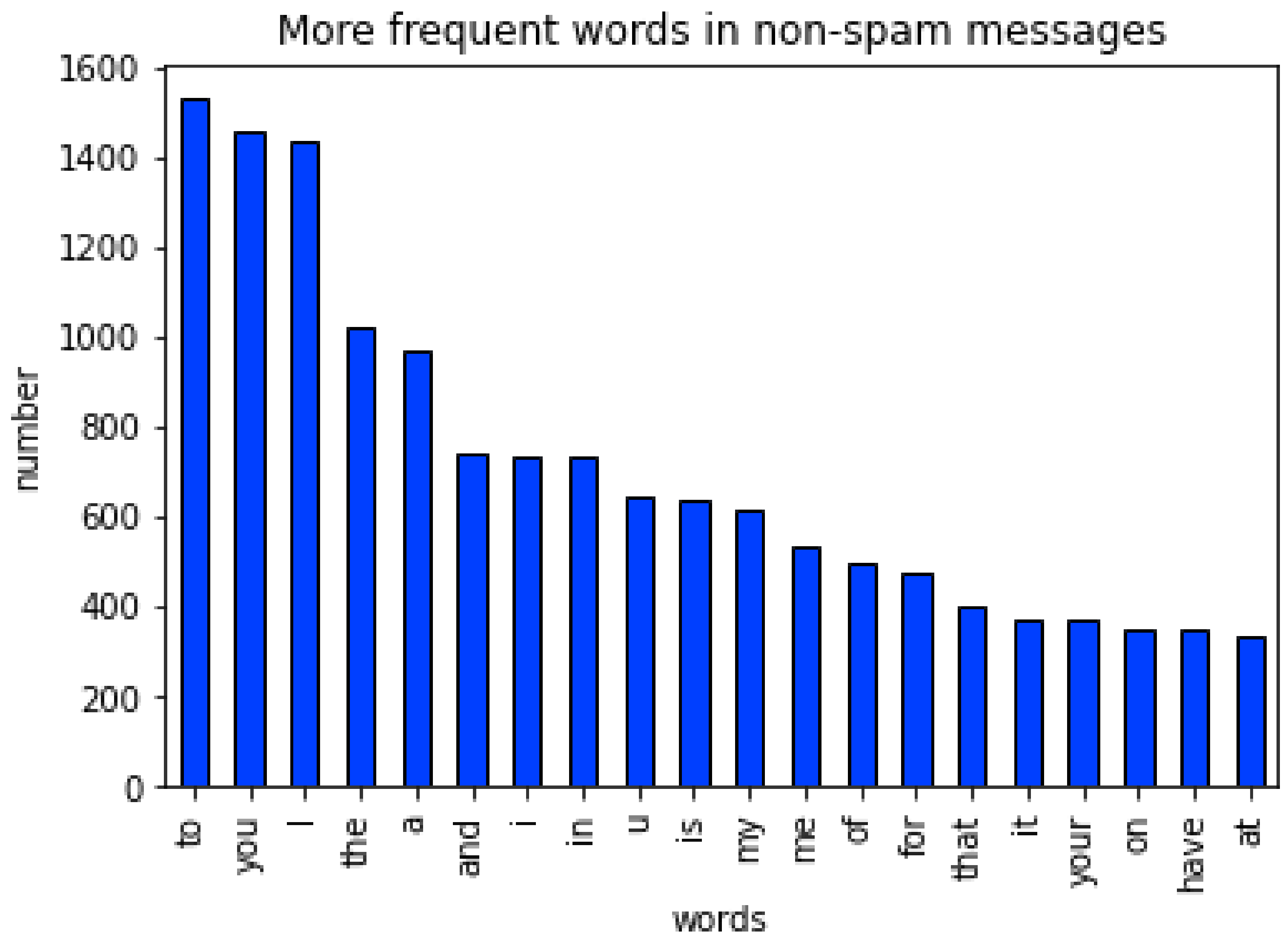
Using different visualizations to understand and modify the dataset

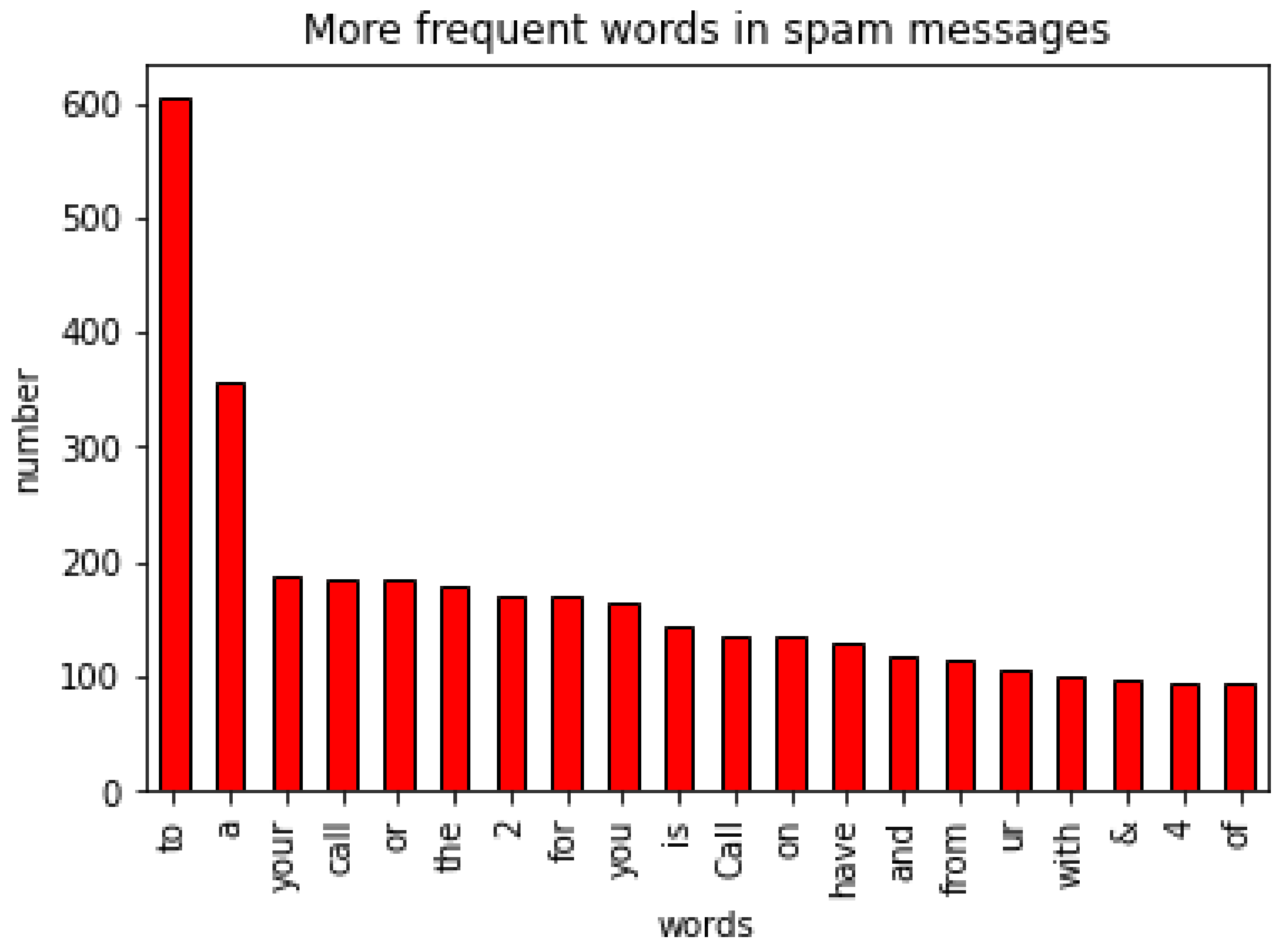


87% of the messages in the dataset is HAM
13% of the messages in the dataset is SPAM

Analysing the dataset

Using different visualizations to understand and modify the dataset





While looking carefully we notice that the most frequent words used in SPAM and HAM are same, these are mostly 'STOP WORDS' which we will look into later.

Cleaning the dataset for predictive analysis

As mentioned earlier we need to ignore the 'STOP WORDS' for the algorithm to work with better efficiency.

Steps:

1. import stopwords from NLTK,
2. use maketrans() and translate() method to replace certain characters
3. converting every word to lower cases and joining in the sentence after removing the stop words in the English language.

After removing stopwords

```
0       Go jurong point crazy Available bugis n great ...
1                                  Ok lar Joking wif u oni
2       Free entry 2 wkly comp win FA Cup final tkts 2...
3                      U dun say early hor U c already say
4                 Nah dont think goes usf lives around though
                              ...
5567    2nd time tried 2 contact u U å£750 Pound prize...
5568                          Ì b going esplanade fr home
5569                          Pity mood Soany suggestions
5570    guy bitching acted like id interested buying s...
5571                                       Rofl true name
Name: Message, Length: 5572, dtype: object
```

Vectorizing

We cannot work with the text directly when using machine learning algorithms. Instead, we need to convert the text to numbers, this process is called Vectorization.

Scikit learn helps us in this process of 'Vectorizing' by providing these utilities:

- **tokenizing strings and giving an integer id for each possible token,**
- **counting the occurrences of tokens in each document.**
- **normalizing and weighting with diminishing importance tokens that occur in the majority of samples/documents.**

Two common types of Vectorizer

Count Vectorizer

TF-IDF Vectorizer

Difference between Count and TF-IDF Vectorizer

Count Vectorization involves counting the number of occurrences each word appears in a document.
Count vectorizer will fit and learn the word vocabulary and try to create a document term matrix in which the individual cells denote the frequency of that word in a particular document.

Document-1: He is a smart boy. She is also smart.
Document-2: Chirag is a smart person.

Unique Words: ['He', 'She', 'smart', 'boy', 'Chirag', 'person']

	He	She	smart	boy	Chirag	person
D1	1	1	2	1	0	0
D2	0	0	1	0	1	1

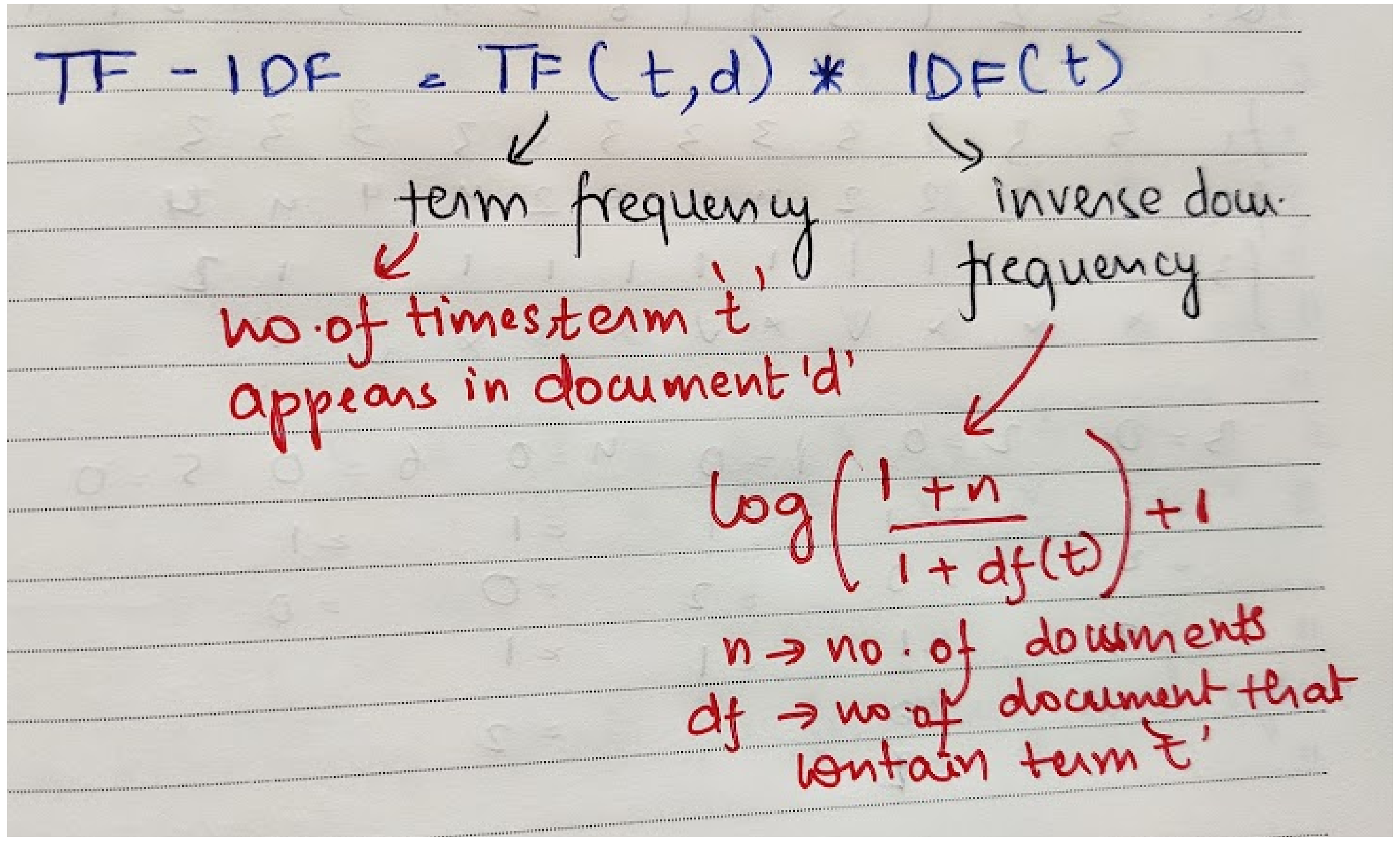
Vector for 'smart' is [2,1]

Why will I use the TF-IDF Vectorizer?

If we were to feed the direct count data directly to a classifier, very frequent terms like, "a", "the", "is" would overshadow the frequencies of rarer yet more important terms.

In order to re-weight the count features into floating-point values suitable for usage by a classifier, it is very common to use the tf–idf transform.

Tf means term-frequency while idf means inverse document-frequency.



The matrix contains a weight value that signifies how important a word is for an individual text message or document.

How will a TF-IDF Matrix look like?

Document-1: Text processing is necessary.
Document-2: Text processing is necessary and important.
Document-3: Text processing is easy.

Unique Words: ['and','easy','important','is','necessary','processing','text']

```
[[0 0 0 1 1 1 1]
 [1 0 1 1 1 1 1]
 [0 1 0 1 0 1 1]]
```

Count Vectorizer array

```
[[0.         0.         0.         0.46333427 0.59662724 0.46333427
  0.46333427]
 [0.52523431 0.         0.52523431 0.31021184 0.39945423 0.31021184
  0.31021184]
 [0.         0.69903033 0.         0.41285857 0.         0.41285857
  0.41285857]]
```

TF-IDF Array

This will be fitted into different classifier models.

Let's try out different classifier ML algorithms

We first split the dataset into features train and test, labels train and test, which will be used for classifier.fit and classifier.predict methods.

```
[ ]  from sklearn.linear_model import LogisticRegression
     from sklearn.svm import SVC
     from sklearn.naive_bayes import MultinomialNB
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import accuracy_score
```

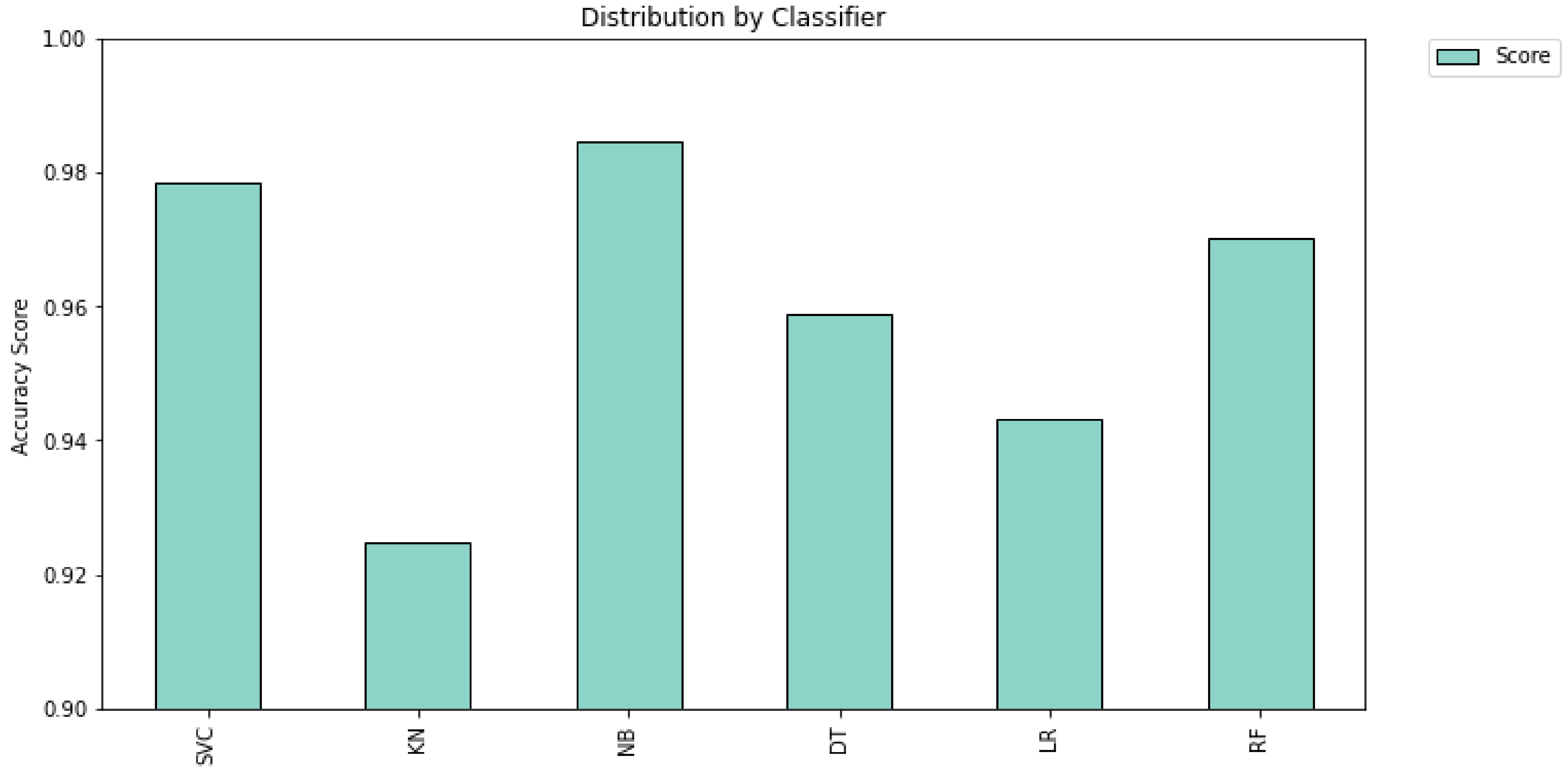
```
def train_classifier(clf, feature_train, labels_train):
    clf.fit(feature_train, labels_train)

def predict_labels(clf, features):
    return (clf.predict(features))
```

After importing the classifier algorithms and creating their object, I convert the objects into the dictionary for easier accessibility and then use a for loop to call these fit and predict user defined functions.

Accuracy and Graphs (Part 1)

I create a list, having the accuracy value of different classifiers and then covert this whole thing into a data frame for easier plotting.

	Score
SVC	0.978469
KN	0.924641
NB	0.984450
DT	0.958732
LR	0.943182
RF	0.970096



Ensemble Algorithms like Decision Trees and Random Forest have performed relatively poor in comparision to Support Vector Machine and Naive Bayes.

A study of twists and turns

Now let me introduce you to Stemming, it is Text Normalization, these techniques in the field of Natural Language Processing are used to prepare text, words, and documents for further processing.

Playing, Plays, Played are having a common root "play".
Similarly Cars, Car's, car have a common root "car".

In our case Stemming helps us to achieve the root forms.

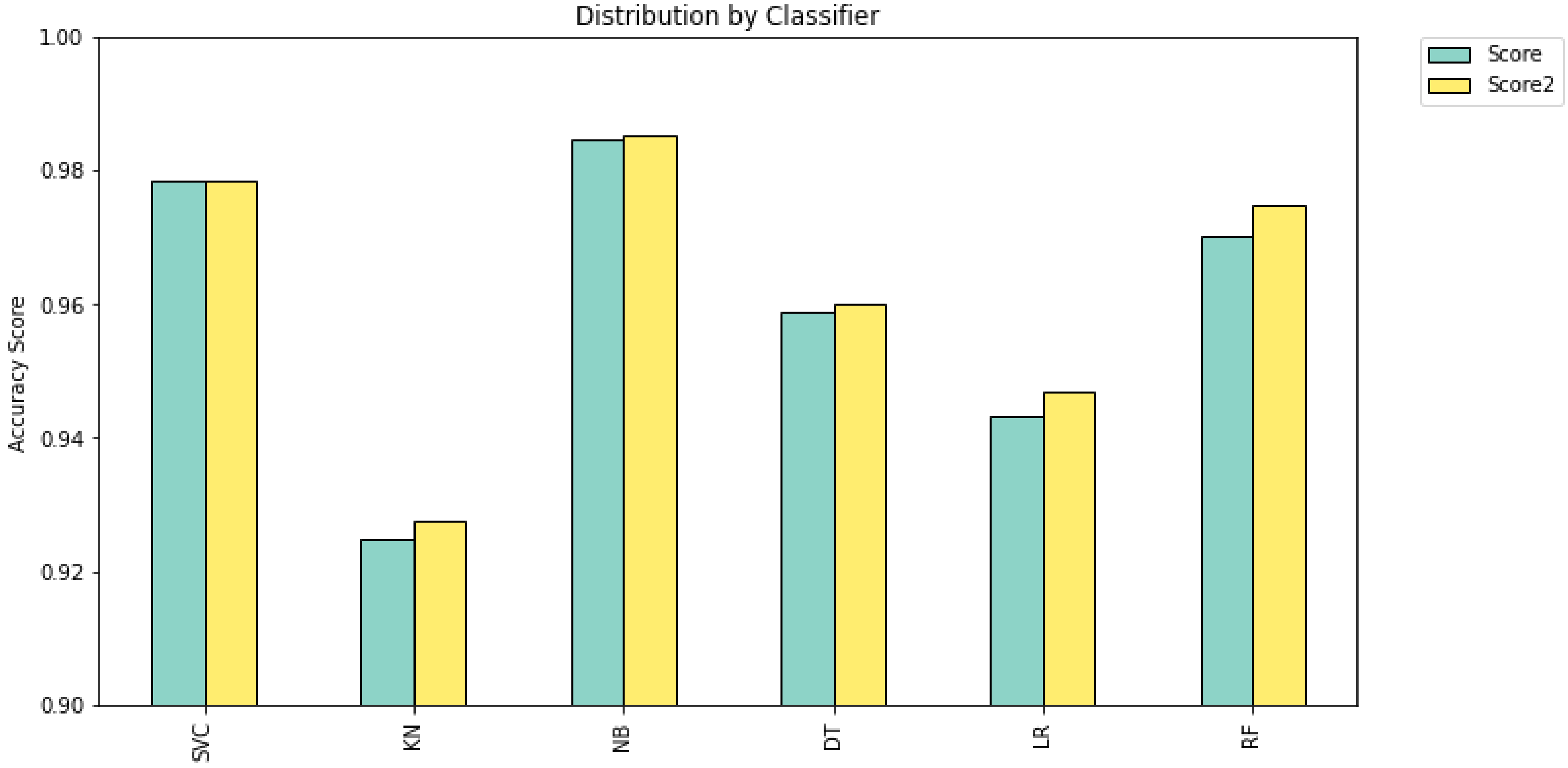
```
0       Go jurong point crazy Available bugis n great ...
1                           Ok lar Joking wif u oni
2       Free entry 2 wkly comp win FA Cup final tkts 2...
3                     U dun say early hor U c already say
4                  Nah dont think goes usf lives around though
                                ...
5567    2nd time tried 2 contact u U å£750 Pound prize...
5568                     Ì b going esplanade fr home
5569                     Pity mood Soany suggestions
5570    guy bitching acted like id interested buying s...
5571                                    Rofl true name
```

```
0       go jurong point crazi avail bugi n great world...
1                          ok lar joke wif u oni
2       free entri 2 wkli comp win fa cup final tkts 2...
3                     u dun say earli hor u c alreadi say
4                  nah dont think goe usf live around though
                                ...
5567    2nd time tri 2 contact u u å£750 pound prize 2...
5568                          ì b go esplanad fr home
5569                          piti mood soani suggest
5570    guy bitch act like id interest buy someth els ...
5571                                    rofl true name
```

Accuracy and Graphs (Part 2)

I vectorized the stemmed data, split it to train and test, ran a for loop so every classifier algorithm could run fit and predict methods on the modified 'stemmed' data. I then obtain the accuracy scores inside a data frame for visualization.

	Score	Score2
SVC	0.978469	0.978469
KN	0.924641	0.927632
NB	0.984450	0.985048
DT	0.958732	0.959928
LR	0.943182	0.946770
RF	0.970096	0.974880



The scores are slightly better than previous check.

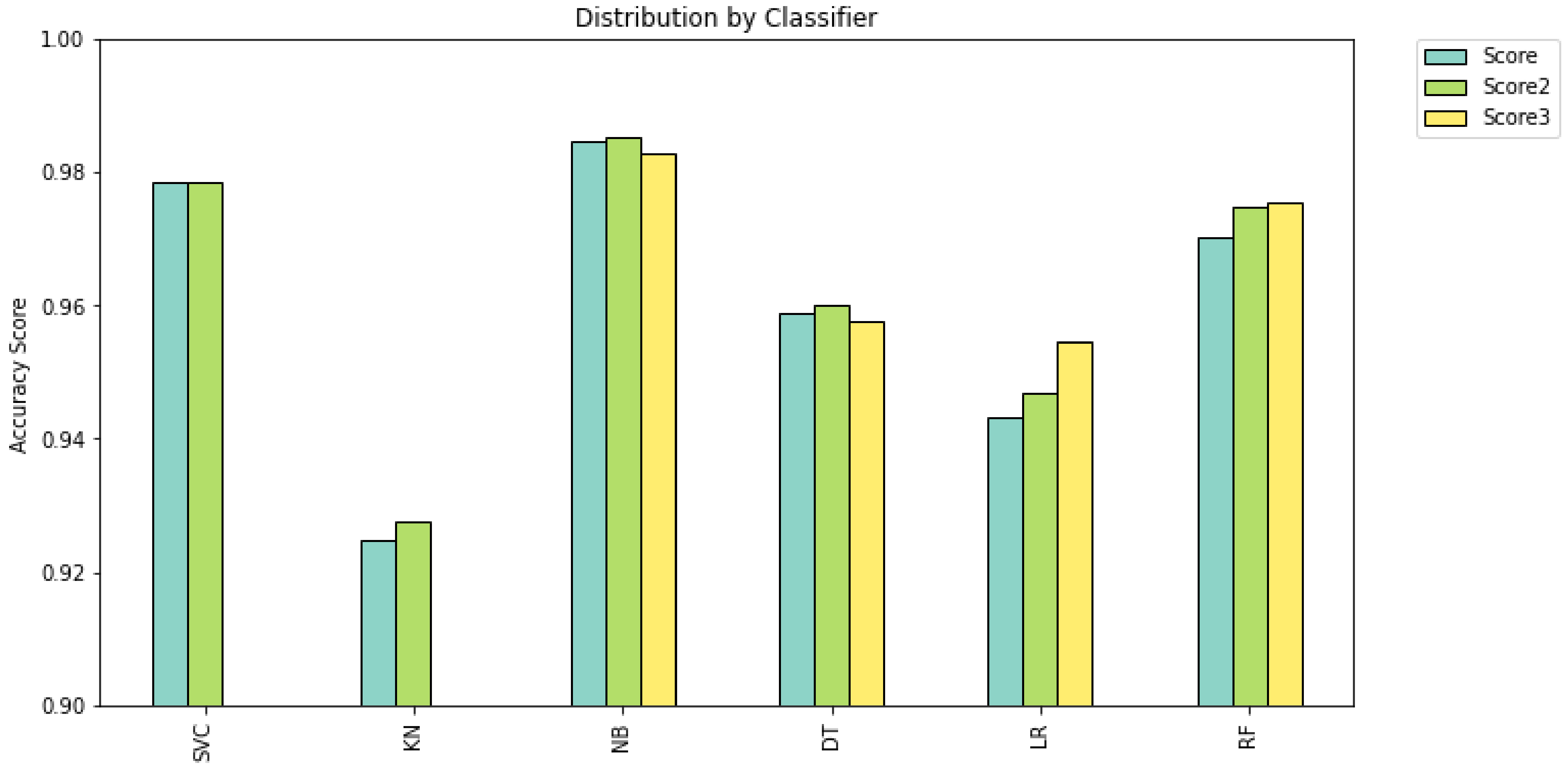
The lengthier the message, the more the chances of it being a SPAM.

After considering a normal message with stop words removed and shortening the length of the word to just its root (Stemming), we will now check the accuracy of these classfiers considering a key factor that is, length of the messages.

This time there's no need for vectorizing the dataset again, I'll just add 'length' column of our data frame to the already vectorized features and labels variables and run the fit and predict method of various classifiers again.

Accuracy and Graphs (Part 3)

	Score	Score2	Score3
SVC	0.978469	0.978469	0.861244
KN	0.924641	0.927632	0.881579
NB	0.984450	0.985048	0.982656
DT	0.958732	0.959928	0.957536
LR	0.943182	0.946770	0.954545
RF	0.970096	0.974880	0.975478



In this study, Multinomial Naive Bayes comes out as the undisputed best algorithm for SPAM classification, having the best accuracy regardless of the conditions, but why?

Understanding Naive Bayes Algorithm

Naive Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.

We will be using multinomial classification as it suits best for the discrete values like word counts. It is very useful in text classification. Let's understand why.

Let's take the example of normal and spam message:

Let's take probabilities of certain words from a friendly message and a spam. (example)

HAM

P('Dear'|NM) : 0.47
P('Friend'|NM) : 0.29
P('Lunch'|NM) : 0.18
P('Money'|NM) : 0

SPAM

P('Dear'|SPAM) : 0.29
P('Friend'|SPAM) : 0.14
P('Lunch'|SPAM) : 0
P('Money'|SPAM) : 0.57

Probabilities or Likelihoods?

Understanding Naive Bayes Algorithm

HAM

P('Dear'|NM) : 0.47
P('Friend'|NM) : 0.29
P('Lunch'|NM) : 0.18
P('Money'|NM) : 0.06

SPAM

P('Dear'|SPAM) : 0.29
P('Friend'|SPAM) : 0.14
P('Lunch'|SPAM) : 0
P('Money'|SPAM) : 0.57

Let's assume 8 out of 12 messages are normal messages so
P(Normal Messages): 0.67
P(Spam Messages): 0.33

We get a message "Dear Friend"

We take an initial guess that it's a normal message, the probability of a normal message coming from our initial guess is called Prior Probability.

✓

Score of Dear Friend being Normal = P(Normal Message) x P(Dear|NM) x P(Friend|NM) = 0.09
Score of Dear Friend being SPAM = P(SPAM Message) x P(Dear|SPAM) x P(Friend|SPAM) = 0.01

Because we got a greater score for the message being normal than for the score for the message being SPAM, we can conclude that the message we have received is a Normal Message.

Understanding Naive Bayes Algorithm

HAM

P('Dear'|NM) : 0.47
P('Friend'|NM) : 0.29
P('Lunch'|NM) : 0.18
P('Money'|NM) : 0.06

SPAM

P('Dear'|SPAM) : 0.29
P('Friend'|SPAM) : 0.14
P('Lunch'|SPAM) : 0
P('Money'|SPAM) : 0.57

Let's assume 8 out of 12 messages are normal messages so
P(Normal Messages): 0.67
P(Spam Messages): 0.33

We get a message "Lunch Money Money Money"

From the first look we can say that this is SPAM, because of the higher probability of 'money' in SPAM.

But But But ! As per the last example we considered the scores of the message to tell if they were SPAM or not.

Score of the message being Normal = 0.000002 ✓
Score for the message being a SPAM = P(SPAM) x P(Lunch|SPAM) x 4x P(Money|SPAM)
= 0.33 x 0 x 4 x 0.57 = 0
By this logic we will conclude wrongly that this message is NORMAL.

Understanding Naive Bayes Algorithm

To avoid this 0 factor coming into play for the probability, we increase the count of words by 1 as a buffer.
This increasing count by x value is termed as ALPHA.

SPAM

P('Dear'|SPAM) : abc
P('Friend'|SPAM) : abc
P('Lunch'|SPAM) : 0.09
P('Money'|SPAM) : abc

Every probability will be updated accordingly. But more importantly, we won't have a 0 to deal with.

Score of the message being Normal = 0.00001
Score for the message being a SPAM = 0.00122 ✓

Because of the alpha value we got a greater score for the message being spam than for the score for the message being normal, we can conclude that the message we have received is a SPAM Message.

Why is Naive Bayes so naive?

The thing about Naive Bayes is it treats all the words equally, for example, NB will treat "Dear Friend" and "Friend Dear" equally consequently returning the same score.

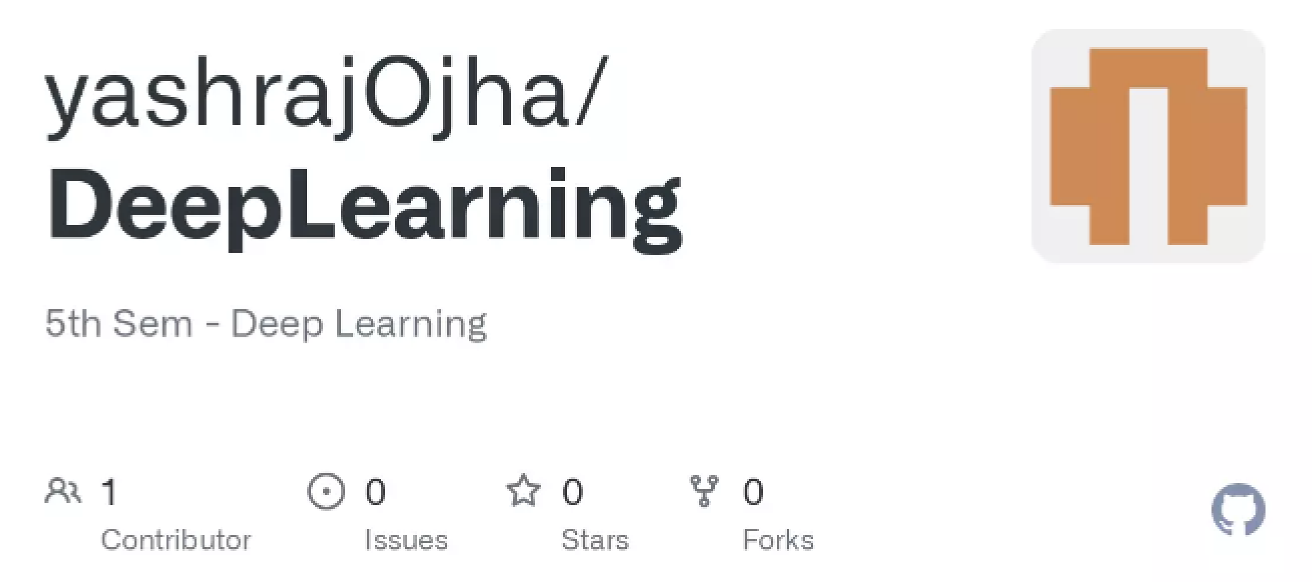
NB assumes that the occurrence of a certain feature is independent of the occurrence of other features.

Why does it ignore the order of phrases and words? Because keeping track of orders is totally unnecessary and understanding each and every language would be impossible.

And so Multinomial Naive Bayes does comparatively well against other classifier algorithms for text classification.

Thank You

Link to the repository:



DeepLearning/DL Seminar Code.ipynb at main · yashrajOjha/DeepLearning

5th Sem - Deep Learning. Contribute to yashrajOjha/DeepLearning development by creating an account on GitHub.

GitHub