# YOU TUBE SPAM COLLECTION

# BY

# KOTA SANJEEVA

## Domain Background



The aim of this project is to explore the results of applying machine learning techniques to Message spam or ham detection.

## Problem Statement



In this project, we have analyzed

different methods to identify spam or ham messages. We will use the different approach, based on word count and term-frequency inverse document-frequency (tf-idf) transform to classify the messages using the You Tube Spam Collection v.1 dataset originates from the UCI Machine Learning Repository.

## Datasets and Inputs



The dataset used for this project is you tube Spam Collection v.1 dataset originates from the UCI Machine Learning Repository. This dataset has been collected from free or free for research sources at the Internet. The Data set consists of five csv files. Each .csv file has five attributes, namely, comment Id, author, Date, Content, Class. Each .csv file have five attributes.

Data set consists of five csv files. Each csv file have Five attributes i.e.., comment Id, author, date, Content, Tag.

Dataset YouTube ID # Spam # Ham Total

Psy 9bZkp7q19f0 --- 175 --- 175 --- 350

Katy Perry CevxZvSJLk8 --- 175 --- 175 --- 350

LMFAO KQ6zr6kCPj8 --- 236 --- 202 --- 438

Eminem uelHwf8o7\_U --- 245 --- 203 --- 448

Shakira pRpeEdMmmQ0 --174 --- 196 --- 370

The collection is composed of just one text file, where each line has the correct class followed by the raw message. There are total 1956 entries in the dataset and has five columns after merging five .csv files.

“Class” and “Content” where each row represent different message and Class contain two unique categories ham and spam. Dataset does not require any kind of cleaning, wrangling and there is no null value in any column

## Preprocessing



**Create corpus**

* 1. Now, we load the text data in to a Corpus.
  2. Think of a corpus as a collection of documents or text snippets.
  3. In our example, each line in the csv file represents a document that we have categorized.
  4. So, we create a Corpus called docs consisting of the Text column in our data frame.
  5. We specify that the source is a vector (i.e. list of elements).

**Clean corpus**

1. The next step is to clean our corpus.
2. Basically, we strip out all the noise from the text and leave only the important parts.
3. Most of the below code is pretty self explanatory.
4. First, we convert all text to lower case.
5. Then we remove all numbers, punctuation and extra white space.
6. Then, we remove stop words.
7. These are common words like “then”, “I”, “it”, “there” etc. And finally, we stem words.
8. This means removing common endings from words.
9. E.g. instead of having the two words “bicycle” and “bicycles”, we end up with one word “bicycl” because we only kept the stem of the word.

**Create DTM**

1. Now, we create the document term matrix.
2. This is a matrix with each document on the x axis and each term on the y axis.
3. Then, we count how many times each term is present in each document.
4. So, basically a numeric representation of the frequency of each word in each document.
5. We will use this matrix to perform our model on.

After we have created the DTM, we convert it to a data frame.

Transform dtm to matrix to data frame - df is easier to work with

Next step is to create a new column with the known category of each text.

We use cbind to bind a new column i.e.., mat.df.

Afterwards, we change the name of the last column of i.e.., mat.df to “category”.

Column bind category (known classification)

Change name of new column to "category"

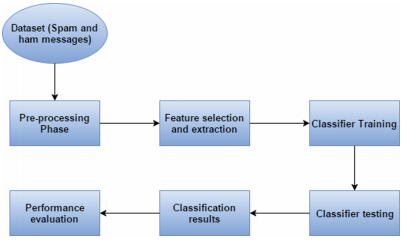
## Solution Statement



We are given labelled training data, so this makes it a supervised machine learning problem. For every message, it can be predicted whether it is ham or spam. To begin with I would like to experiment with techniques which we are going to us are based on word count and term-frequency inverse document-frequency (tf-idf) transform. After which I would like to test the approach using many different algorithms like Naive Bayes, Decision Tree, SVM, K-Nearest Neighbours, Random Forest and Neural Networks.

**System architecture**





### **Flavor of text mining**



1. Separate the words (or phrases) in a large body of text
2. Clean up the data by eliminating punctuation, numbers, homogenizing on case, removing non-content words like “The”
3. Create incidence matrix for words in a document (called a Document-Term Matrix), or some other way of studying word frequency and document similarity. The process is called tokenization.
4. Use frequency of words in documents to classify documents into types or associate documents by similarity

**System Design, Implementation and Results**



1. Our aim is to construct a new classification model which can filter YouTube spam efficiently.
2. This section presents implementation details, dataset, the brief summary of machine learning algorithms and performance evaluation measures to judge the performance of our proposed approach.
3. The detection of spam YouTube is a binary classifi- cation problem where various features are used to train the classifier.

## Best Models



KNN: KNN (K Nearest Neighbor) is one of the most eﬀective algorithms. Based on False Positive, False Negative and Accuracy. For text classiﬁcation, the metric can be used such as the overlap metric (or Hamming distance).

Remember that the KNN model takes three sets of data: Train,test and classifier.

All three sets must have the same number of rows.

Let’s create those three sets of data.

We take a random sample with a size of 50% of the full data and call this train.

Test will hold all the rest of the data.

Train and test contain just the row numbers, so we can use the row numbers for indexing our data, when we create the KNN model.

The last step is to create the classifier.

We isolate all the known categories and put them into cl.

Split data by rownumber into two equal portions

A) Isolate classifier

In order to use our data in the KNN model, we need it without the categories.

The categories are what we are trying to predict.

We create a data frame modeldata with all columns from mat.df except the category.

B) Create model data and remove "category"

Now we are finally ready to create the model! So, from our modeldata, we take the rows that we decided to use for training and test. And, we also feed the known categories of the training data into the model. We call the model knn.pred.

C) Create model: training set, test set, training set classifier

And now comes the cool part: The confusion matrix.

This is a matrix that tells what documents the model predicted correctly, and what documents it did not predict correctly.

Neural Networks: Regarding the parameters, in all simulations, we have employed the following stopping criteria: maximum number of iterations be greater than a threshold θ, the mean square error (MSE) of the training set be smaller than a threshold γ or when the MSE of the validation set increases (checked every 10 iterations).

The utility of artificial neural network Models lies in the fact that they can be used to infer a function from observations. This is particularly useful in applications where the complexity of the data or task makes the design of such a function by hand impractical.

a)Motivation for a New Approach:

* The original Neural Network (MLP) uses linear output nodes when the class is numeric; here we assume the class is nominal. Therefore the code can be used only for classification and does not support regression.
* Last attribute must correspond to the class (this could be easily changed though)
* There's no GUI, so the topology cannot be modified during training time

There is only 1 hidden layer.

* Missing attributes are eliminated using a Filter. This is the reason for having new feature, not present in original implementation.

Algorithm :

Input:

N // Starting neural network

X={x1….xh} // Input tuple from training set

D= {d1,….dm} // Output tuple desired

Output: N // Improved neural network

Back propagation algorithm:

// Illustrate back propagation Propagation (N,X);

m

E=1/2 ∑ (di – yi) 2 ;

i=1

Gradient (N, E) ;

The MSE (Mean Squared Error) is used to calculate the error. Each tuple in the training set is input to this algorithm. The last step of the algorithm uses gradient descent as the technique to modify the weights in the graph. The basic idea of gradient descent is to find the set of weights that minimizes the MSE.

Naïve Bayes: The presence of a particular feature in a class is unrelated to the presence of any other feature. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Finally, Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

### Naive Bayes Classifier

* This type of “classifier” assigns a probability that a new sample is in one class or another (spam or ham).
* From the words in the message and the words not in the message, compute the probability of spam or ham.
* It is based on Bayes rule, frequency analysis of occurrences of words and an independence assumption (the naive part).

### Compute probabilities from training data

Given a word W,

*P*(*spam*/*W*)=*P*(*W*/*spam*)×*P*(*spam*)*P*(*W*)

For words *W*1, *W*2 and ¬*W*3, e.g.,

*P*(*spam*|*W*1∩*W*2∩¬*W*3)=*P*(*W*1∩*W*2∩¬*W*3|*spam*)×*P*(*spam*)*P*(*W*1∩*W*2∩¬*W*3)

### Independence assumption

Assuming the *Wi* are independent, this becomes

*P*(*spam*|*W*1∩*W*2∩¬*W*3)=*P*(*W*1|*spam*)×*P*(*W*2|*spam*)×*P*(¬*W*3|*spam*)×*P*(*spam*)*P*(*W*1)×*P*(*W*2)×*P*(¬*W*3)

The validity of the assumption is less important than how the classifier performs.

### Training the probabilities

* Probabilities on right hand side of the equation are calculated in training data.
* Estimated probability on the left hand side assigned to new messages and used to classify.
* Some test data is set aside to check the accuracy of the classification.

Decision Tree : - Decision table is a machine learning algorithm that represents a set of rules and in this result is good only for some continuous features.

## Evaluation Matrix



|  |  |
| --- | --- |
| Model name | Accuracy |
| Decision Tree | 82.41% |
| Naïve Bayes | 60.65% |
| SVM | 48.67% |
| KNN | 98.69% |
| Neural Networks | 89% |

In order to evaluate the effectiveness of our proposed approach, we will consider eight possible outcomes i.e. true positive rate, false positive rate, true negative rate, false negative rate, f1 score, accuracy, precision, and recall. These are the standard metrics to judge any spam detection system. These evaluation metrics are described in brief as follows:-

• True Positive Rate (TP) - It denotes the percentage of spam messages that were accurately classified by the machine learning algorithm. If we denote spam messages as S and spam messages that were accurately categorized as P, then

TP = P / S

• True Negative Rate (TN) - It denotes the percentage of ham messages that were accurately categorized as ham messages by the machine learning algorithm. If we denote ham message as H and ham messages that were accurately categorized as ham by Q, then

TN = Q / H

• False Positive Rate (FP) - It denotes the percentage of ham messages that were wrongly categorized as spam by the machine learning algorithm. If we denote ham messages as H and ham messages that were wrongly classified as spam by R, then

FP = R / H

• False Negative Rate (FN) - It denotes the percentage of spam messages that were incorrectly classified as ham message by the machine learning algorithm. If we denote spam messages as S and number of SMS spam messages that were incorrectly classified as ham by T, then

FN = T / S

• Precision - It denotes the percentage of messages that were spam and actually classified as spam by the classification algorithm. It shows the exact correctness. It is given as:-

Precision = TP / (TP + FP)

• Recall - It denotes the percentage of messages that were spam and classified as spam. It shows the completeness. It is given as:-

Recall =TP / (TP + FN )

• F-measure - It is defined as the harmonic mean of Precision and Recall. It is given as:-

F-measure =2\* Precision \*Recall \ Precision \* Recall

**Project Design**



The theoretical workflow of the project would look like:

1. Download and pre-process the You Tube Spam Collection v.1 dataset.
2. Test and find best approach (word count or tf-idf vectorizer) to classify the messages.
3. Selection of approach and splitting the dataset into training and testing data.
4. Initialize various classifier and train it using training data.
5. Evaluate the classifiers and finding best the model for a dataset using testing data.

**Experiments / Observations**



1. We trained the classifier on 100% of the training data and tried to classify the Corpus given to us. It consists of 1956 messages.
2. Some of those messages were repeated in that corpus many times. The classifier labeled 1005 messages as spam and 951 messages as ham.
3. There were false positives (ham labeled as spam) and false negatives(spam labeled as ham). But occurence of false positives was greater than false negatives.
4. False positives > False Negatives was because the classifier associated the label 'spam' with features – length of message, numeric terms and word features such as 'free', 'call', 'urgent' etc.
5. Presence of false positives and false negatives was because of message content having other language text (hindi, chinese) typed in English and mixed with regular English content. Some of these words are unseen in training data to some extent, but they are handled by laplace smoothing in the training phase to some effect.
6. This scenario can get improved accuracy if we introduce some features which use ratio of capital words to small words, ratio of spam words to normal words, consider more different currency symbols, number of URLs in the message.

**Conclusion and FutureWork**



* The YouTube Spam problem is increasing nowadays with the increase in the use of text messaging.
* YouTube Spam filtering is the big challenge these days.
* In this paper, we propose a technique for YouTube Spam filtering based on 10 feature using five machine learning algorithms namely Naïve Bayes, KNN, SVM, Decision Tree, Neural Networks and Random Forest.
* The dataset that we have used in our work consists of 1956 messages Out of all classification algorithms, KNN Classification Algorithm gives best results with 98.1% accuracy.
* In our futu.re work, we will try to add more features as best spam features help in detecting spam messages more accurately.
* We will also try to collect more and more datasets from the real world.