Final project CS 418

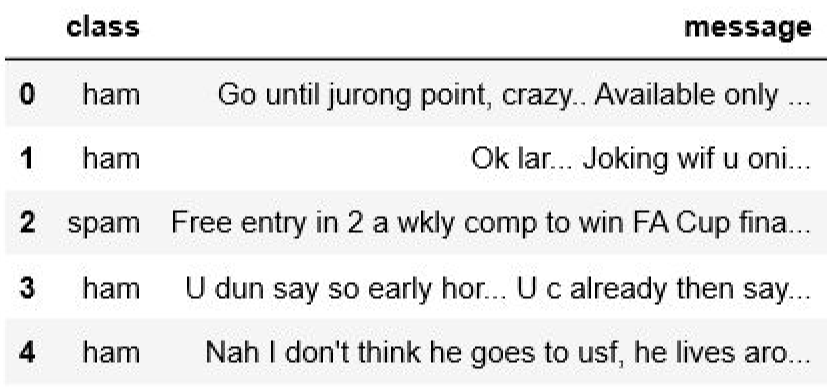
SMS spam classification

Utsav Bhadresh Shah (ushah21@uic.edu)

Tuhin Kundu (tkundu2@uic.edu)

Sai Aakarsh Reddy Koppula (skoppu3@uic.edu)

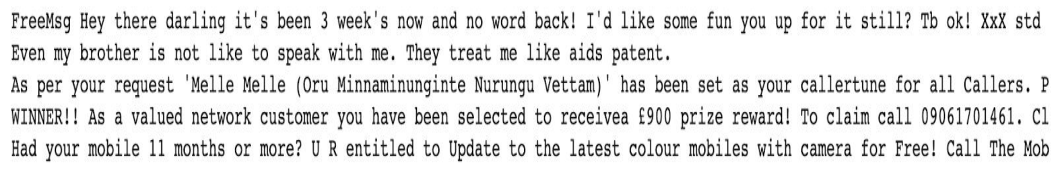
- **Problem Selection**:
    - The goal of this project is to classify SMS text messages using machine learning models and automatedly classify them as ham or spam signifying whether a particular text message is spam or not. Apart from applying the classification to our dataset, we also perform text analysis from a statistical point of view. We achieve our best results using SVM classifier with linear kernel with TfIdf vectorized with 98.44% accuracy.
    - Dataset: https://archive.ics.uci.edu/ml/datasets/sms+spam+collection
- **Data Collection:**
    - The dataset contains 5572 text messages which are appropriately labelled ham and spam. The dataset contains words which are in a noisy form and colloquial in nature with respect to daily textual conversations between people. Hence the dataset comprises data from the informal domain space and a significant amount of data pre-processing needs to be done for our modelling part.
    - The dataset is imbalanced in nature with 4825 instances of ham class and 747 instances of spam class.



	class	message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

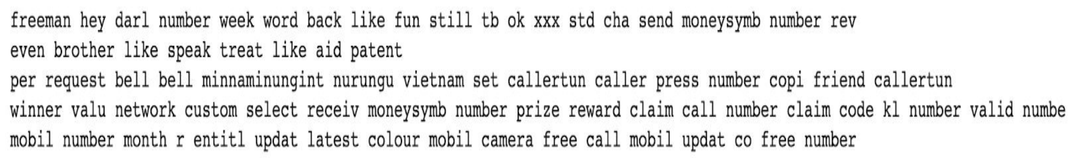
- **Data Preparation & Text Analysis:**

- The data belongs to an informal domain of text messages and therefore requires a fair bit of pre-processing.
- Converted the messages to lowercase to capture the unique tokens and not differentiate between alphabet cases.
- The messages contained phone numbers, email addresses, http links, money symbols, and other numbers which were substituted by a fixed string. Eg. xyz@abc.com was replaced with “emailaddr”. We masked the phone numbers in the messages to maintain privacy. Implemented regular expressions to perform this cleaning step.
- Implemented regular expressions to remove punctuations and replace them with whitespace.
- The text messages also contained normal textual conversations between people. That means a lot of short-forms and typos were present in the conversation. To correct them, a spell corrector was implemented which basically decides the best word based on edit distance. Pyspellchecker library was used for the same.
- Implemented stemming on words to bring them to their root form.
- Majority of textual data has very high occurrences of stop words like and, the, is , of, etc. These words do not describe or differentiate the text in any way, they are just used to make the sentences grammatically correct. The important words are the tokens other than the stop words. So, we removed the stopwords from the messages as they do not act as the differentiating factor.
- Input messages:



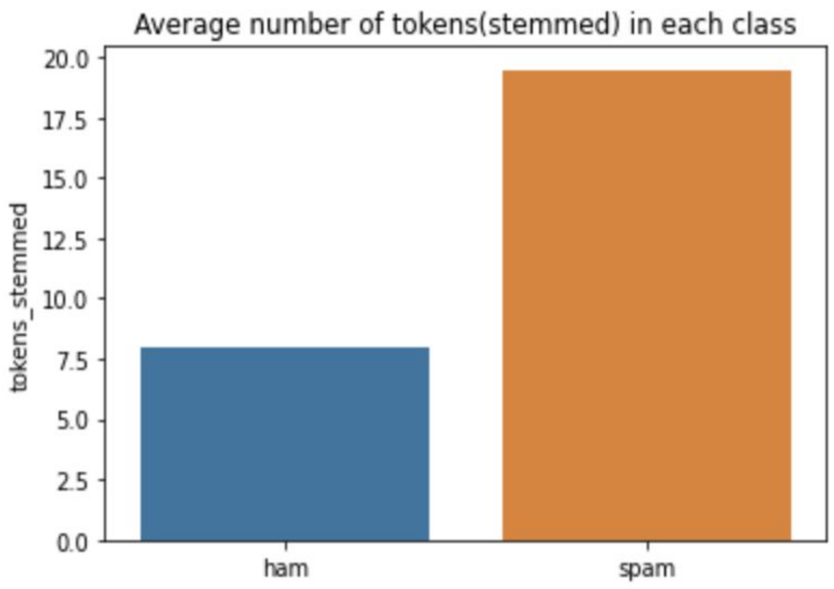
```
FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std
Even my brother is not like to speak with me. They treat me like aids patent.
As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. P
WINNER!! As a valued network customer you have been selected to receivea £900 prize reward! To claim call 09061701461. Cl
Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mob
```

- Processed messages:

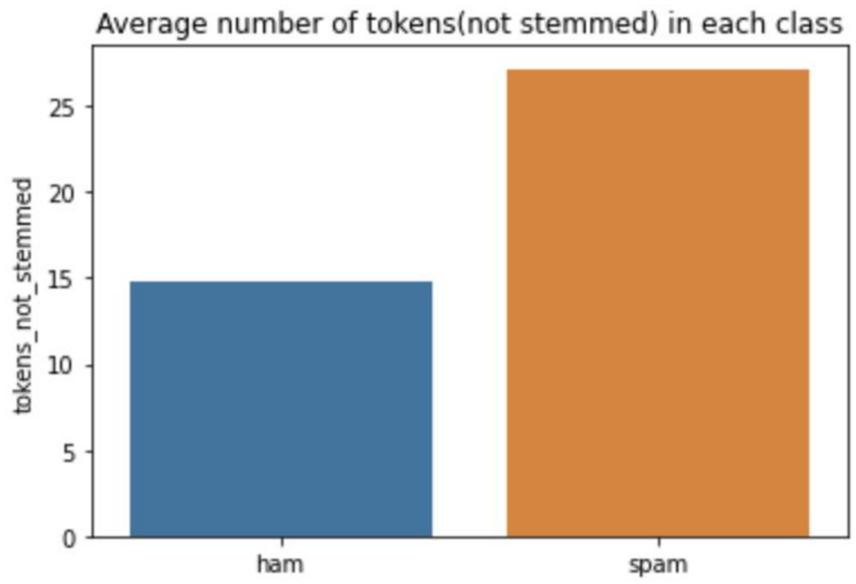


```
freeman hey darl number week word back like fun still tb ok xxx std cha send moneysymb number rev
even brother like speak treat like aid patent
per request bell bell minnaminungint nurungu vietnam set callertun caller press number copi friend callertun
winner valu network custom select receiv moneysymb number prize reward claim call number claim code kl number valid numbe
mobil number month r entitl updat latest colour mobil camera free call mobil updat co free number
```

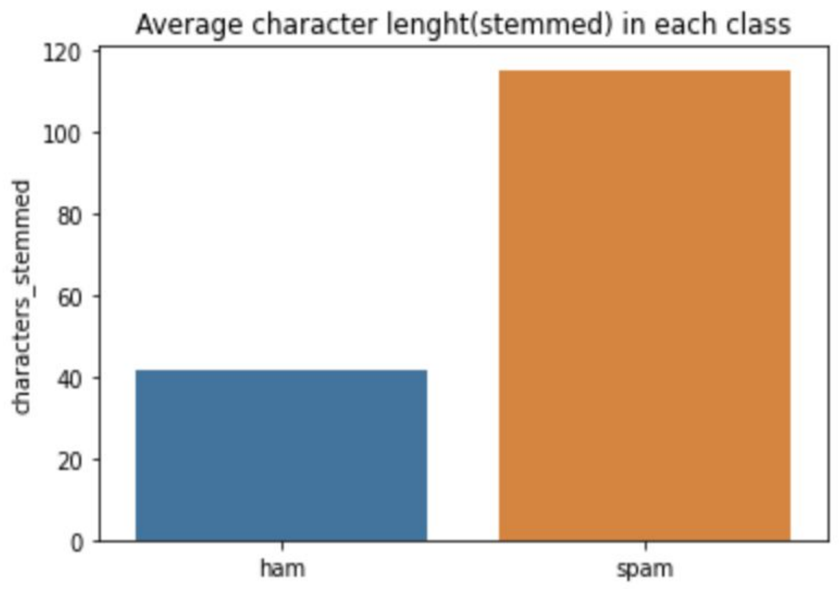
- **Data Exploration:**
  - It is observed that the number of tokens in the spam messages is more than that of the ham(non-spam) messages. The average number of tokens in the spam messages is around 19 whereas for ham is 8. This is for the data that is stemmed and with the stop words removed.



- It is observed that the number of tokens in the spam messages is more than that of the ham(non-spam) messages. The average number of tokens in the spam messages is around 27 whereas for ham is 14. This is for the data that is cleaned but contains stop words and it is not stemmed.

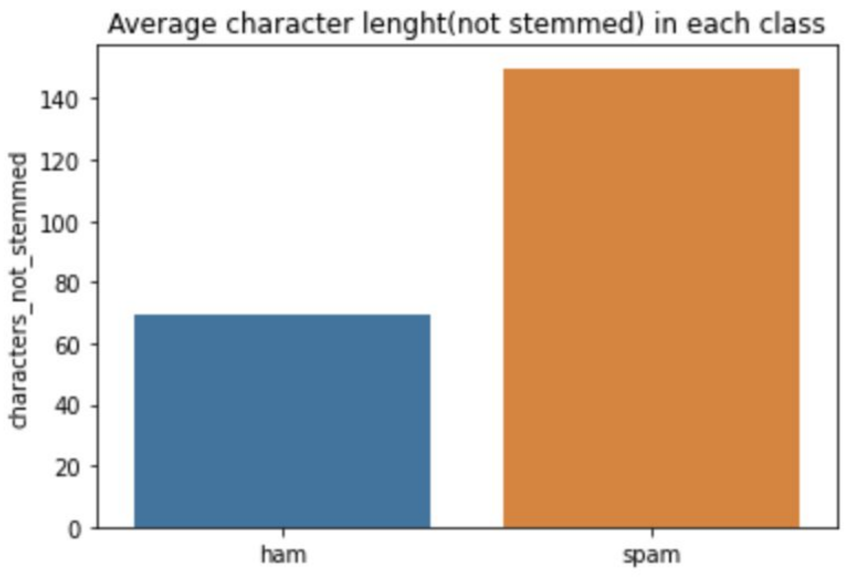


- It is observed that the length of the messages in terms of number of characters in the spam messages is more than that of the ham(non-spam) messages. The average length in the spam messages is around 115 characters whereas for ham is 41 characters. This is for the data that is stemmed and with the stop words removed.



○

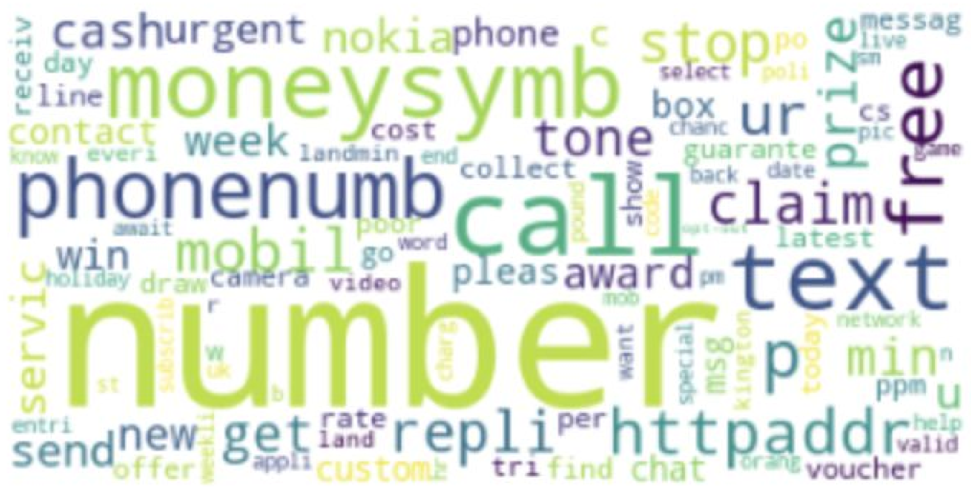
○ It is observed that the length of the messages in terms of number of characters in the spam messages is more than that of the ham(non-spam) messages. The average length in the spam messages is around 149 characters whereas for ham is 69 characters. This is for the data that is cleaned but contains stop words and it is not stemmed.



○

○ The below Word-Cloud is based on the word count of the tokens in the spam messages. It is clear that the spam messages have more occurrences of phone numbers, money symbols, cash, urgent, prize, claim, etc. which are the suspicious words as expected in spam messages.

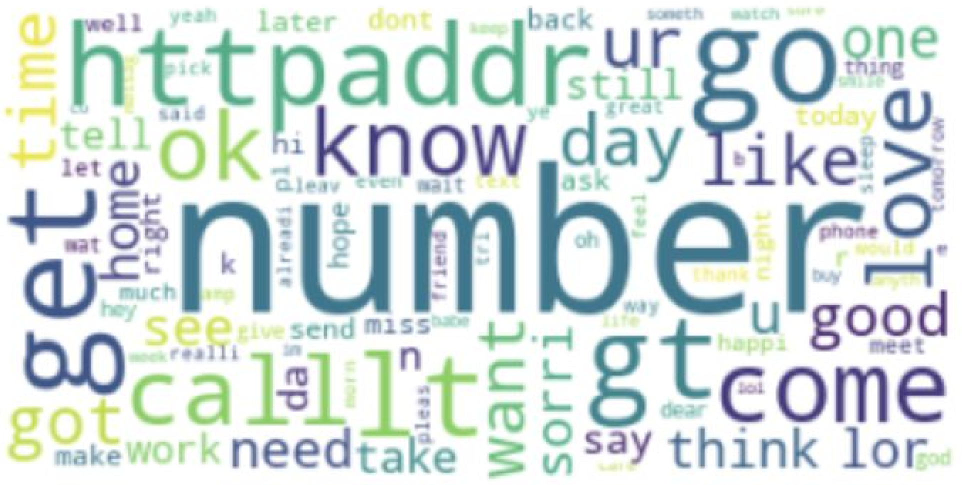
==== top 100 by occurences for spam ====



○

○ The below wordcloud is based on the word count of the tokens in the ham messages. It contains words that occur generally in a normal conversation.

==== top 100 by occurences for ham =====



○

- **Data modelling (Classification)**:
    - Text Analysis:
        - For the data modeling exercise, we used two methods, namely, CountVectorizer and TfIdf vectorizer.
        - The Count Vectorizer creates a sparse matrix of the documents(messages) which contains the counts of the words.
        - The TfIdf Vectorizer also creates a sparse matrix of the documents(messages) which contains the term frequency over inverse documents frequency for each token in the message.
    - Modeling:

- For modelling our dataset and to quantify the results using evaluation metrics, we've used Naive Bayes, Decision Tree and SVM.
- Naive Bayes: Naive Bayes is the most widely used machine learning algorithm used for text classification due to the fact that all the words in the text are conditionally independent of each other.
- SVM: SVM is one of the most popular classification algorithms which discriminatively distinguishes between multiple classes in the dataset. We've used a linear kernel to classify our dataset.
- Decision Tree: Decision tree uses information gain to distinguish between feature subsets and ultimately classifying the instances using their leaf nodes. We've used gini and entropy.
- Results for the frequency (count) vectorized data:
-

Model	Accuracy	F1 score (ham)	F1 score (spam)
Naive Bayes	84.68	90.46	61.09
Decision Tree using entropy	97.42	98.51	90.33
Decision Tree using gini	97.60	98.62	90.95
SVM with linear	**98.38**	**99.06**	**93.90**

- Results for the TfIdf vectorized data:
-

Model	Accuracy	F1 score (ham)	F1 score (spam)
Naive Bayes	84.03	91.10	58.73
Decision Tree using entropy	97.30	98.45	89.60
Decision Tree using gini	97.36	98.48	90
SVM with linear	**98.74**	**99.27**	**95.10**

        - The naive bayes model is used as a baseline model for our results as it is most widely used for spam text classification. Given our text messages are from the informal domain with a lot of noise (even after pre-processing), naive bayes does not achieve results as good as discriminative classifiers due to the fact that a large number of words would be specific to a particular message, hence lowering the conditional probability while getting the joint probability for a particular class.
        - Discriminative classifiers such as decision tree and SVM are better suited to the task as they use various discriminative metrics to distinguish between ham and spam.
        - The classification results using the frequency vector and TfIdf vector are pretty similar.
        - The SVM algorithm with linear kernel on the TfIdf vectorized data gives the best results in classification in terms of accuracy (98.74) as well as f-1 scores for the two classes (99.27) and (95.10).

- Conclusion:
    - In this problem, we use Naive Bayes, Decision Tree and SVM classifiers to classify text messages as ham or spam. We generated the feature vectors of the text messages using count (frequency) vectorizer and TfIdf vectorized as input to our machine learning models. We conclude that discriminative classifiers are more suited to this dataset and get our best results using Tf-Idf vectorizer with SVM (linear kernel).