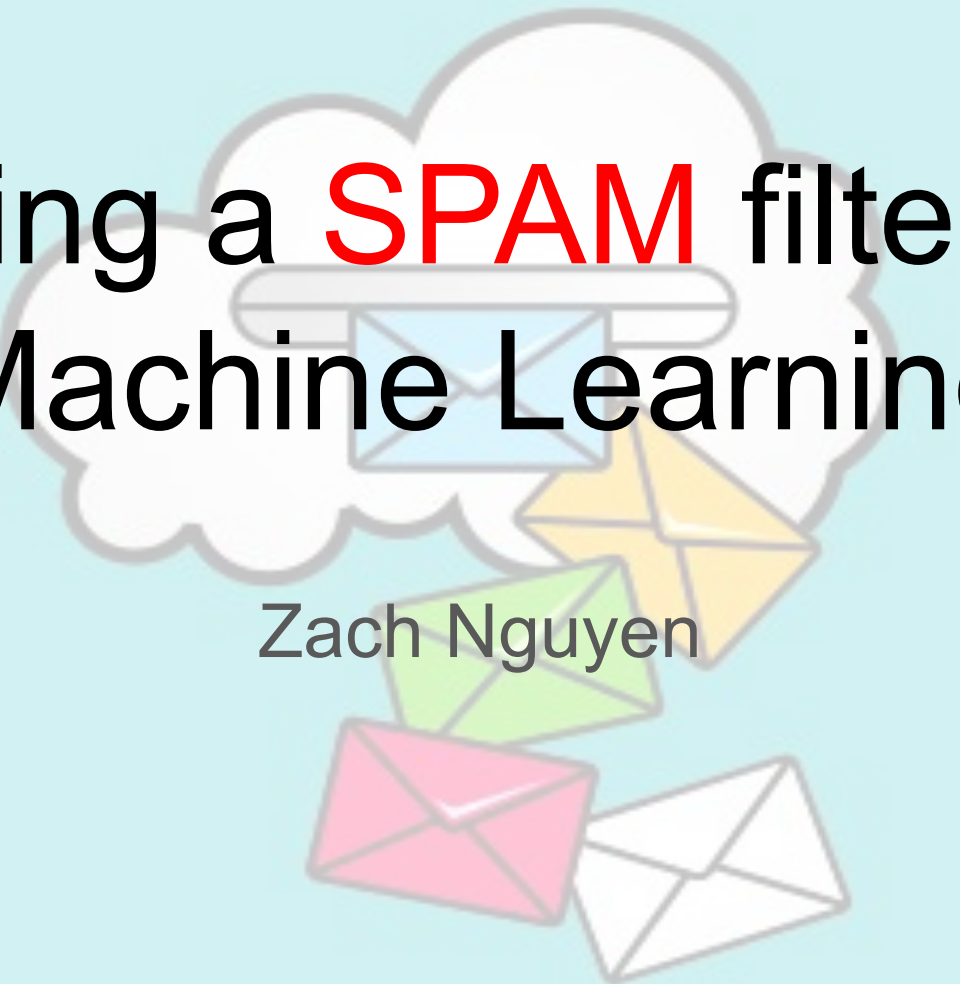


# Building a **SPAM** filter with Machine Learning



Zach Nguyen

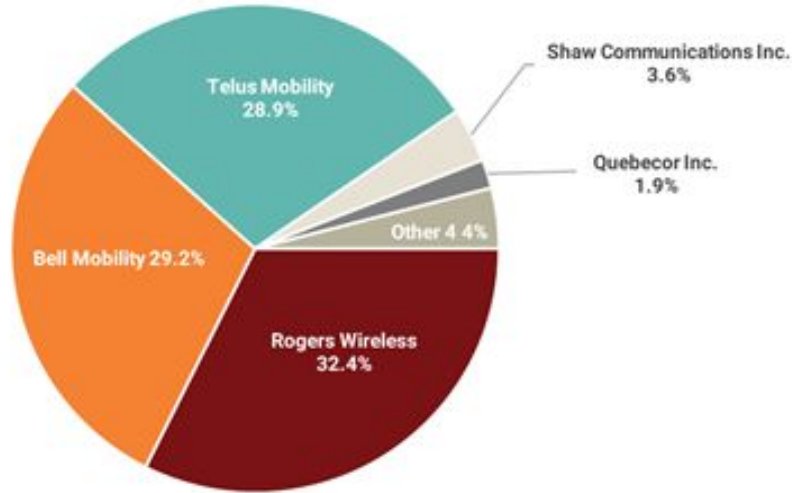
# Part 1: Data Story

Spams poses significant **inconvenience** and **risks** to mobile phone users.



# Companies in the telecom sector look to outcompete and **win over customers.**

## Breakdown: Canada's Wireless Telecom Market (2019)



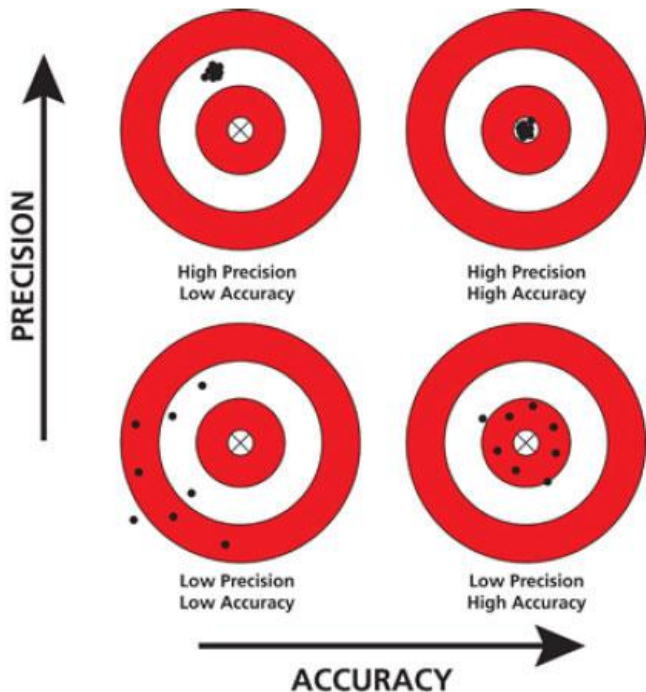
- Oligopoly where top 3 are in fierce competition.
- High penetration prevent new market creation.
- Commodity products with high demand elasticity

UCI's collection of spam datasets provide a good starting point for experimenting with spam classification algorithms.



- Over 5,500 text sms
- Collected from a variety of sources

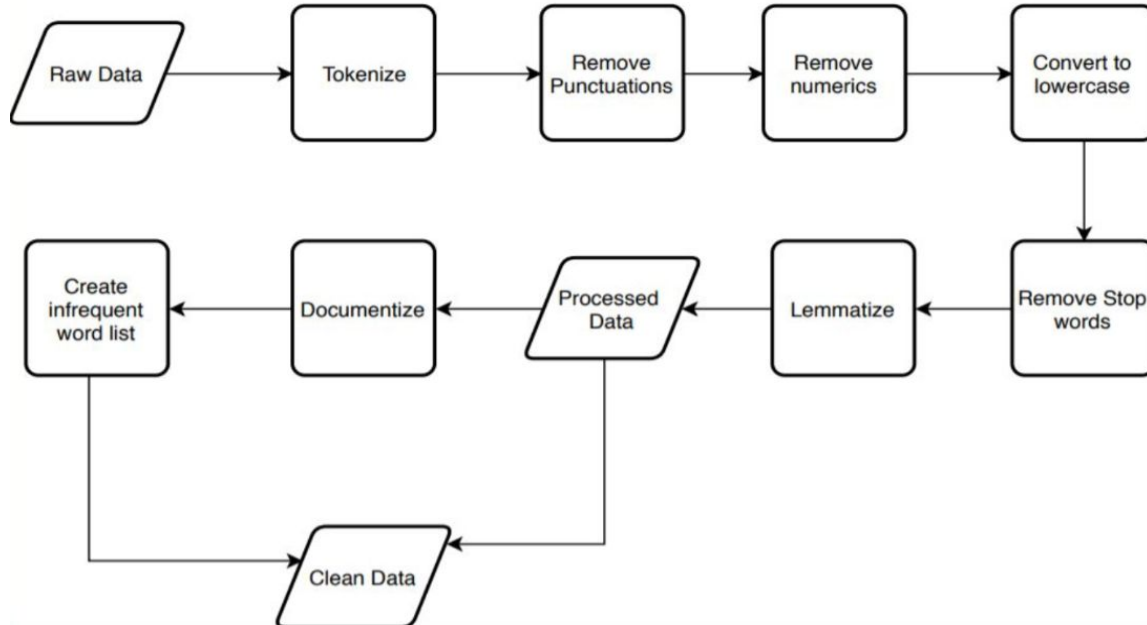
The success of the spam filter is measured by how **precise**, **accurate** and **speedy** it is.



# Part 2: Data Wrangling

# The text messages were transformed with a **text cleaning pipeline**.

**Data Cleaning Pipeline**



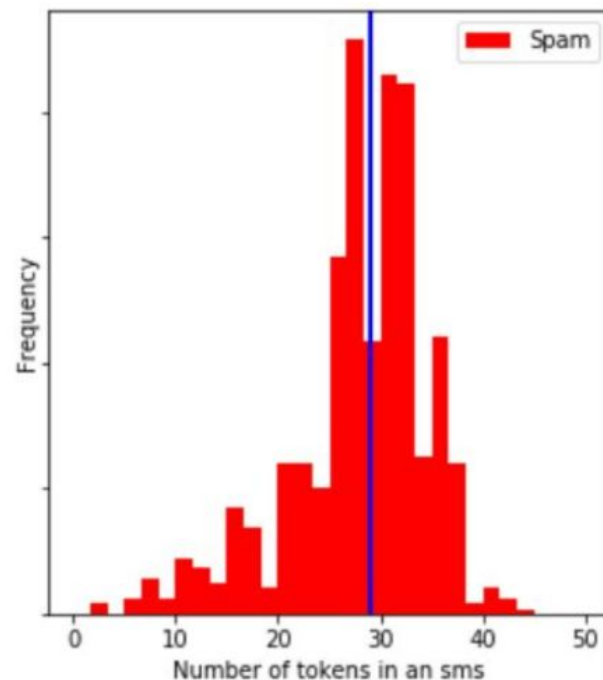
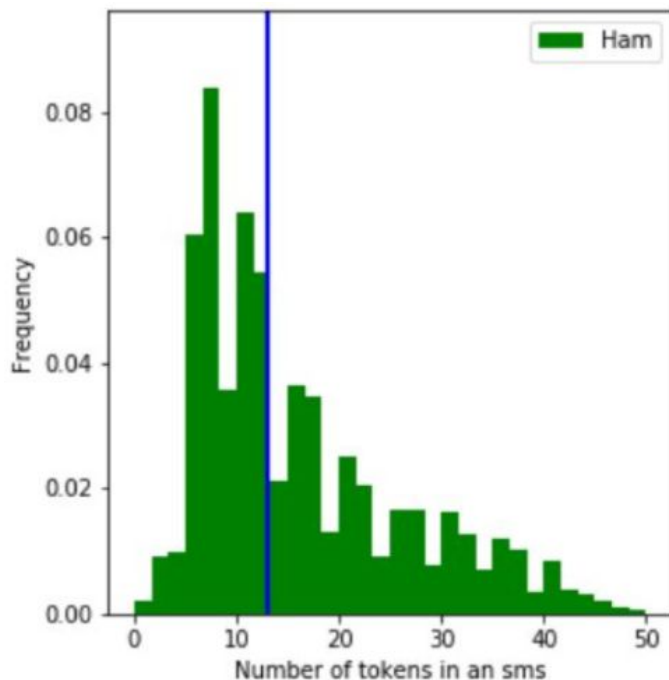


**Meta-Data** was extracted from the text to gain additional information for ML algorithm.

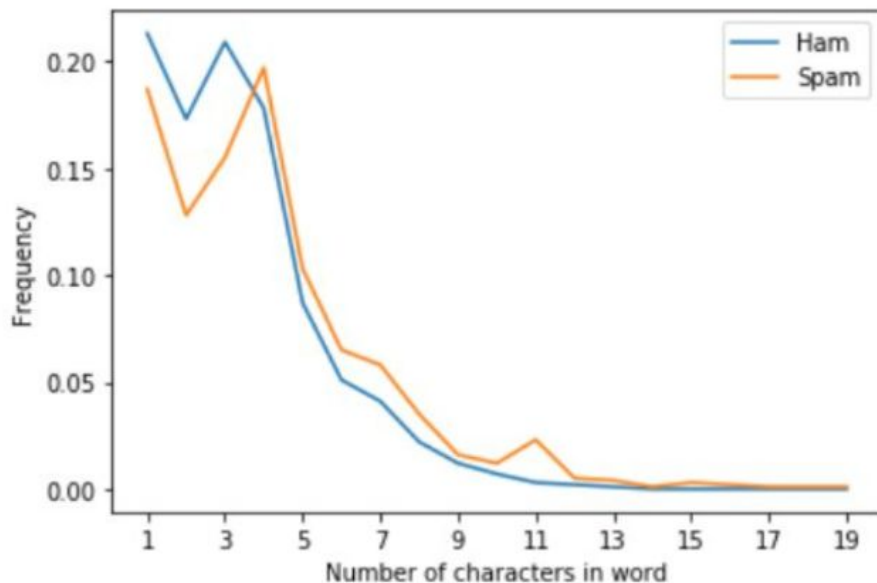
	n_token	avg_wlen	n_num	has_num	n_upper	n_stops	has_email	has_money	has_phone
<b>1978</b>	23	4.000000	2	1	2	6	0	1	0
<b>3989</b>	27	2.833333	0	0	1	5	0	0	0
<b>3935</b>	13	3.285714	0	0	0	2	0	0	0
<b>4078</b>	21	3.785714	0	0	0	9	0	0	0
<b>4086</b>	33	3.625000	0	0	1	9	0	0	0

# Part 3: Exploratory Data Analysis

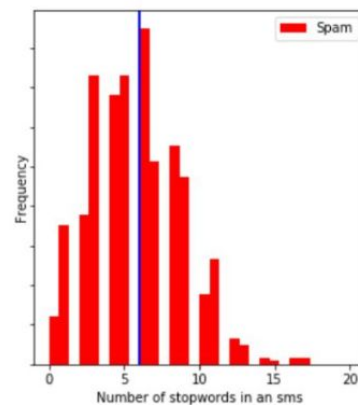
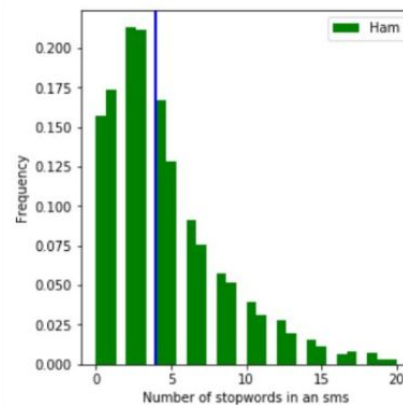
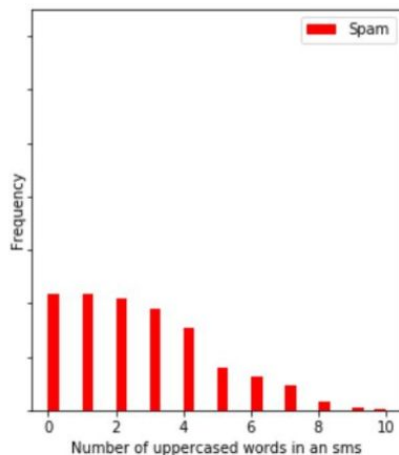
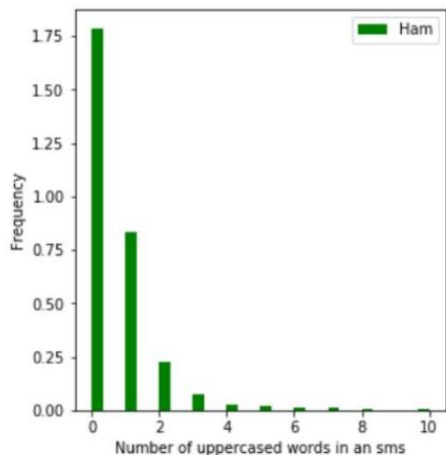
Histograms show that the text messages have very different **lengths**.



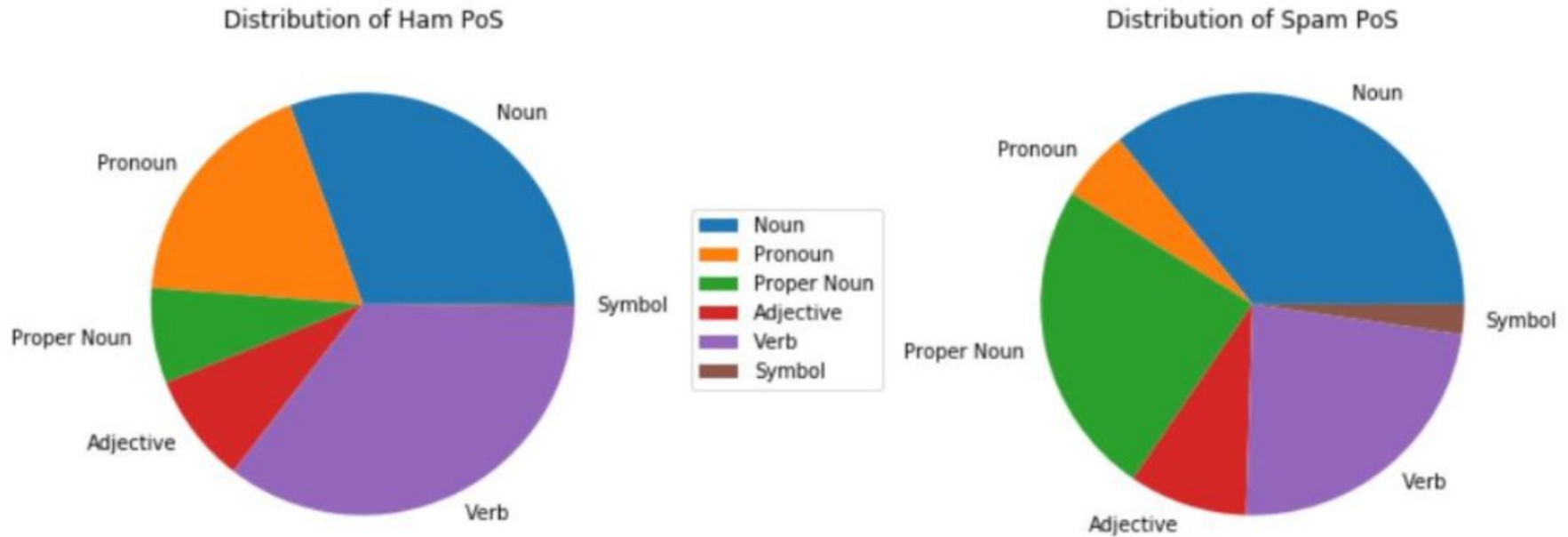
Although not clear on frequency plot, permutation test shows there is a statistically significant difference between **word length** of the 2 classes.



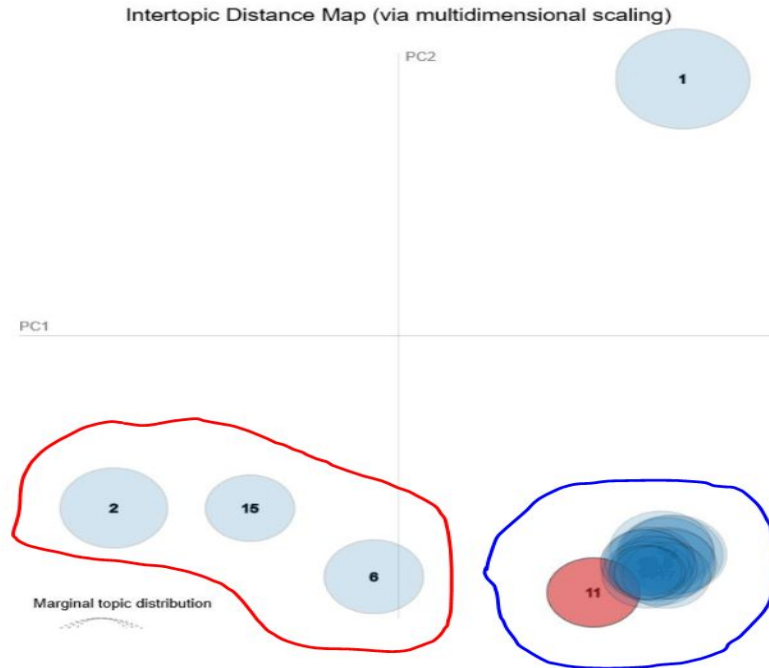
The number **uppercased words** of and number of **stopwords** have some predictive power.



Part of Speech analysis shows spams for likely to use **Nouns**, **Symbols** but less Pronouns.



# Latent Dirichlet Allocation topic modelling shows discrimination across reduced dimension.



# Part 4: Machine Learning



**F-Beta** at Beta = 0.1 (10 fold bias towards precision) was chosen as model metric to reflect business needs.

$$F_{\beta} = \frac{(1 + \beta^2) \cdot (\text{precision} \cdot \text{recall})}{(\beta^2 \cdot \text{precision} + \text{recall})}$$

# A list of algorithms were prepared based on suitability with text classification data

Algorithm	Rationale
Naive Bayes	LDA analysis shows vocabularies fall into topics and may lend well to word probability models
Linear SVM	SVM has been shown to work very well with high-dimensional data such as the Bag-of-word representation
Tree based techniques	Simple and interpretable

# The data was fed onto a **machine learning pipeline** using sklearn's functionality

- Feature space extraction using CountVectorizer() or TfidfVectorizer() with search for min\_df and ngram\_range hyper parameter.
- Parameter grid for hyper-parameter tuning of each algorithm
- 10 Fold-Validation Grid-search

Naive Bayes performed with the highest Fbeta score with 100% precision

	Model	Training_Accuracy	Test_Accuracy	AUC	Fbeta
0	NB_BoW	0.999	0.994	0.983	1.000
1	NB_Tfidf	1.000	0.989	0.992	0.972
2	SVC_BoW	0.999	0.983	0.989	0.991
3	SVC_Tfidf	1.000	0.989	0.992	0.986

Model performed even better, with higher AUC score (general performance under many threshold) when **stacked with meta-data**

	Model	Training_Accuracy	Test_Accuracy	AUC	Fbeta
0	stack_combined	0.989	0.987	0.975	0.992
1	stack_proba	0.999	0.994	0.997	1.000

# Part 5: Conclusion

## **Conclusions:**

- The final model was able to achieve perfect precision in the test set and 7 False Negatives among the 1,100+ test cases.
- When ran through the whole pipeline, a speed of 640 ms per 100 sms was achieved with a local computer. If the company runs in batches of 100 sms, there won't be noticeable effect on users.

## **Limitations/Future considerations:**

- The dataset was merely experimental, so a high score was achieved. Exploration of messier, more modern datasets will help generalize and create a better business case.