Machine Unlearning – June 2019

Data Science for Business

Chi Sheng Jan
Joanne Wong
Piotr Dudojc
Rita Liu
Yongqiang Cao
Alex Jones

Spam or Ham?

The business case for building a spam filter

INSEAD

Introduction

- Spam is a major security risk, primarily through scams and phishing.
- Email spam filters are more common nowadays, but text spam still poses a threat.
- Our client, the National University of Singapore (NUS), wants to protect its community (students, faculty and staff) by designing a spam filter for text messages.
- Dataset comprises 5572 text messages collected from NUS students, faculty, staff and their contacts, classified as spam or ham (legitimate messages).

Data Sample

	A	B
1	Category	Message
2	ham	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
3	ham	Ok lar... Joking wif u oni...
4	spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std tx
5	ham	U dun say so early hor... U c already then say...
6	ham	Nah I don't think he goes to usf, he lives around here though
7	spam	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
8	ham	Even my brother is not like to speak with me. They treat me like aids patent.
9	ham	As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Ca
10	spam	WINNER!! As a valued network customer you have been selected to receivea £900 prize reward! To claim call 09061
11	spam	Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call
12	ham	I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.
13	spam	SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ Tsa
14	spam	URGENT! You have won a 1 week FREE membership in our £100,000 Prize Jackpot! Txt the word: CLAIM to No: 8101
15	ham	I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted a
16	ham	I HAVE A DATE ON SUNDAY WITH WILL!!
17	spam	XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>> http://wap. xxxn
18	ham	Oh k...i'm watching here:)

Data Dictionary

Category : Label for whether the text message was ham or spam.

Message : The actual text message.

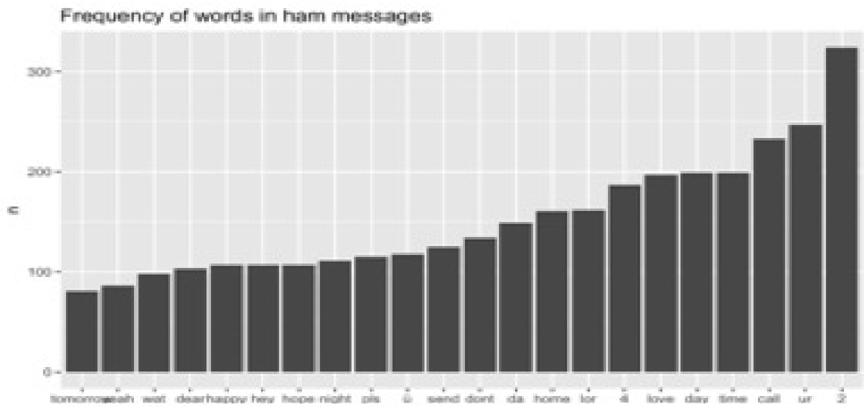
Data preparing and text mining process

1. Data cleanup and encoding verification
   - Correct UTC-8/HTML encoding issues due to previous processing (weird characters e.g. <, >, Ã¼ , Ãœ)
2. Tokenization
3. Filtering "stop words"
4. Language analysis
5. Feature engineering

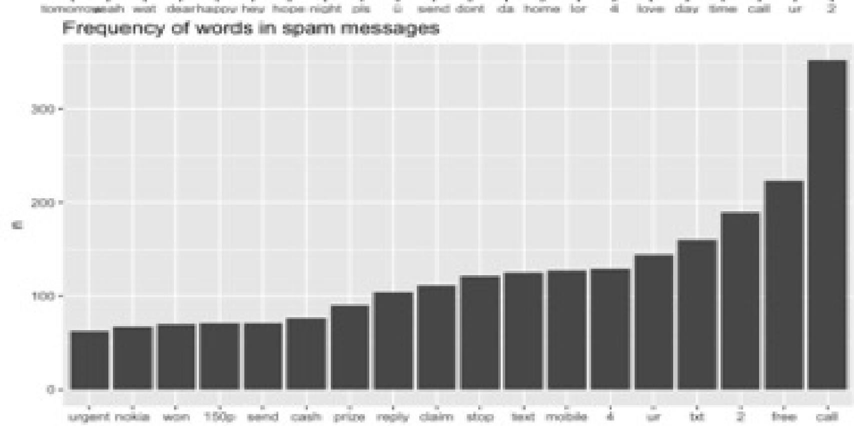
The frequently used words in spam and non-spam messages are different

Word frequency analysis and word count cloud

In ham (non-spam) messages: frequent usage of 2, ur, call, time, day, love, 4



In spam messages: call, free, 2, txt, ur, 4, mobile, text

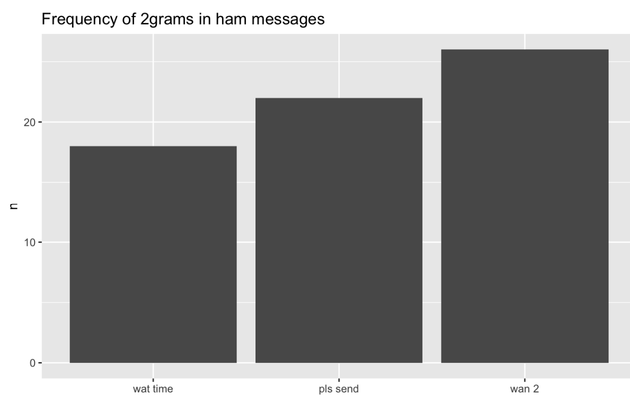


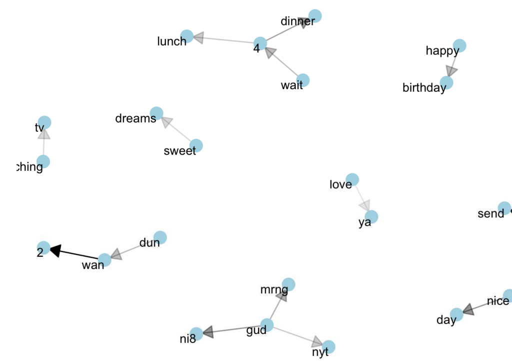
Different feature of ngrams found in spam and non-spam messages

Ngrams frequency and network of bigrams analysis

Ham (non-spam) messages: common 2grams include wan 2, pls send, wait time;

Common sense connection of words

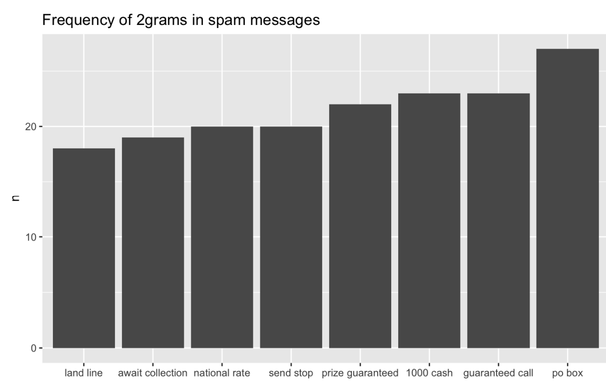


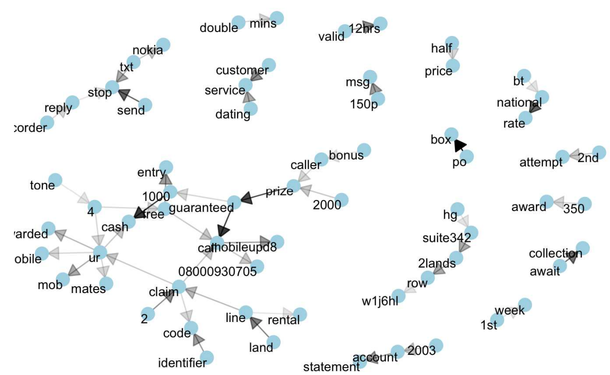


Spam messages:

PO box, guaranteed call, 1000 cash, etc;

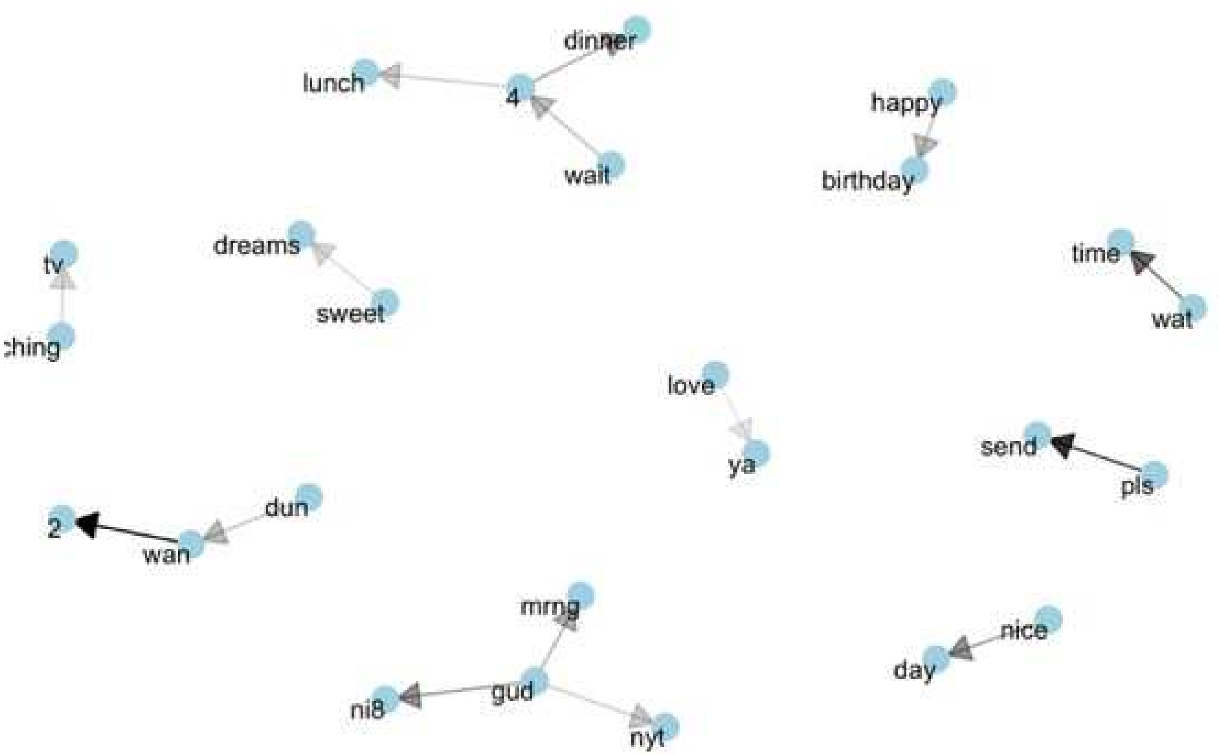
Strange connection of words





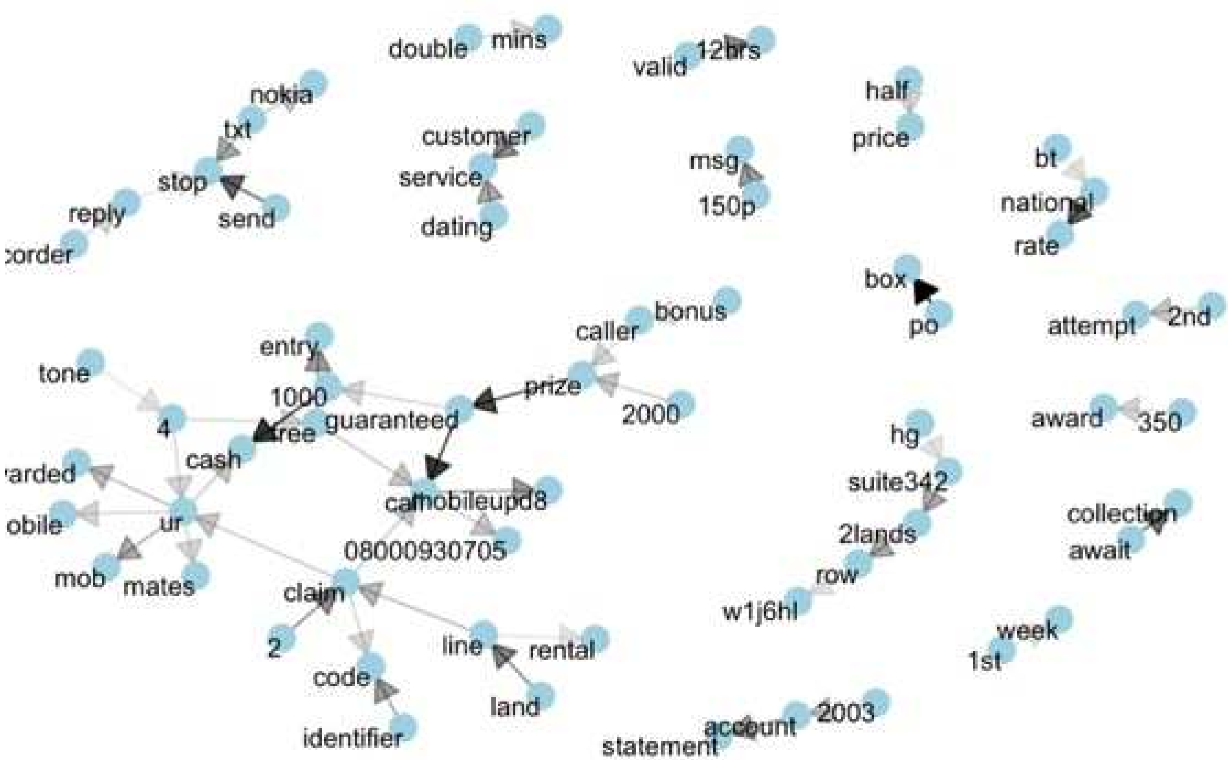
Different feature of ngrams found in spam and non-spam messages

Ngrams network for non-spam messages



Different feature of ngrams found in spam and non-spam messages

Ngrams network for spam messages



Feature engineering

INSEAD

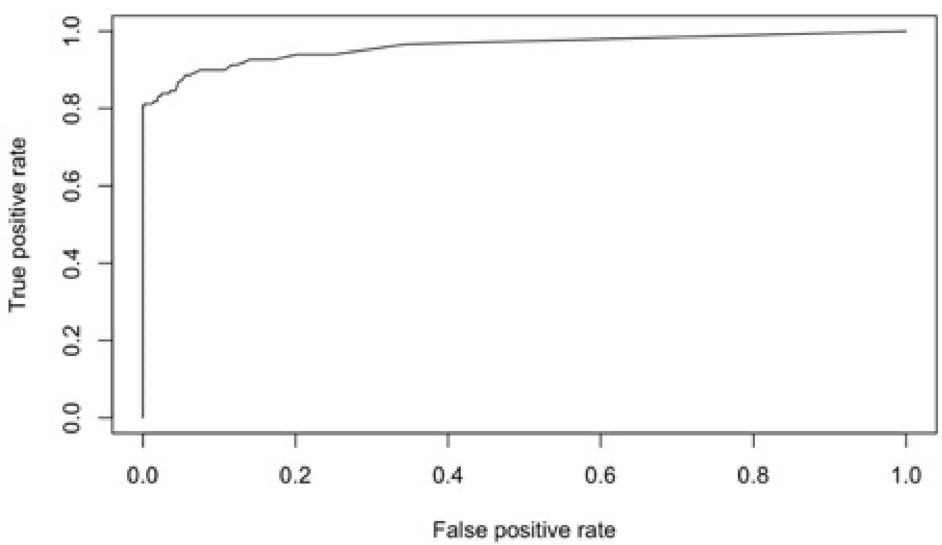
Feature engineering: based on word and ngrams analysis of spam and ham message, new features such as some frequent words, bigrams, and message length and digit numbers are created into the data set

Single words		Ngrams
wat	cash	wan 2
dear	prize	wat time
happy	reply	pls send
hey	claim	gud ni8
hope	stop	gud mrng
night	text	gud nyt
pls	mobile	await collection
u	txt	national rate
dont	free	prize guaranteed
da		1000 cash
home		po box
lor		guaranteed call

Logistic model

Results of logistic model: 0.9678 accuracy and 0.9699 AUC

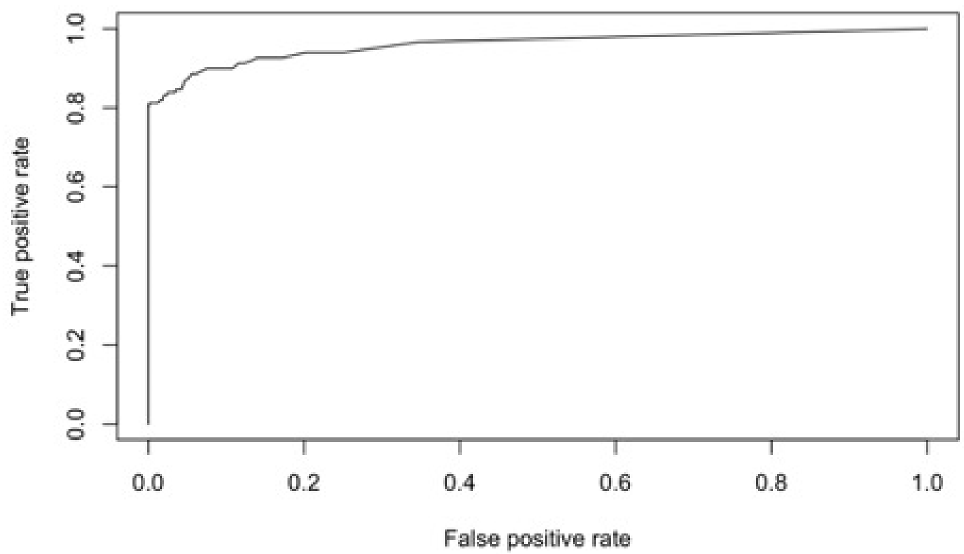
Logistic model with stepwise AIC(both direction) and threshold 0.8




```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 964  33
         1   1  57

               Accuracy : 0.9678
                 95% CI : (0.9553, 0.9776)
    No Information Rate : 0.9147
    P-Value [Acc > NIR] : 2.455e-12

                  Kappa : 0.7538

 Mcnemar's Test P-Value : 1.058e-07

            Sensitivity : 0.63333
            Specificity : 0.99896
         Pos Pred Value : 0.98276
         Neg Pred Value : 0.96690
             Prevalence : 0.08531
         Detection Rate : 0.05403
   Detection Prevalence : 0.05498
      Balanced Accuracy : 0.81615

       'Positive' Class : 1
```

Random forest model

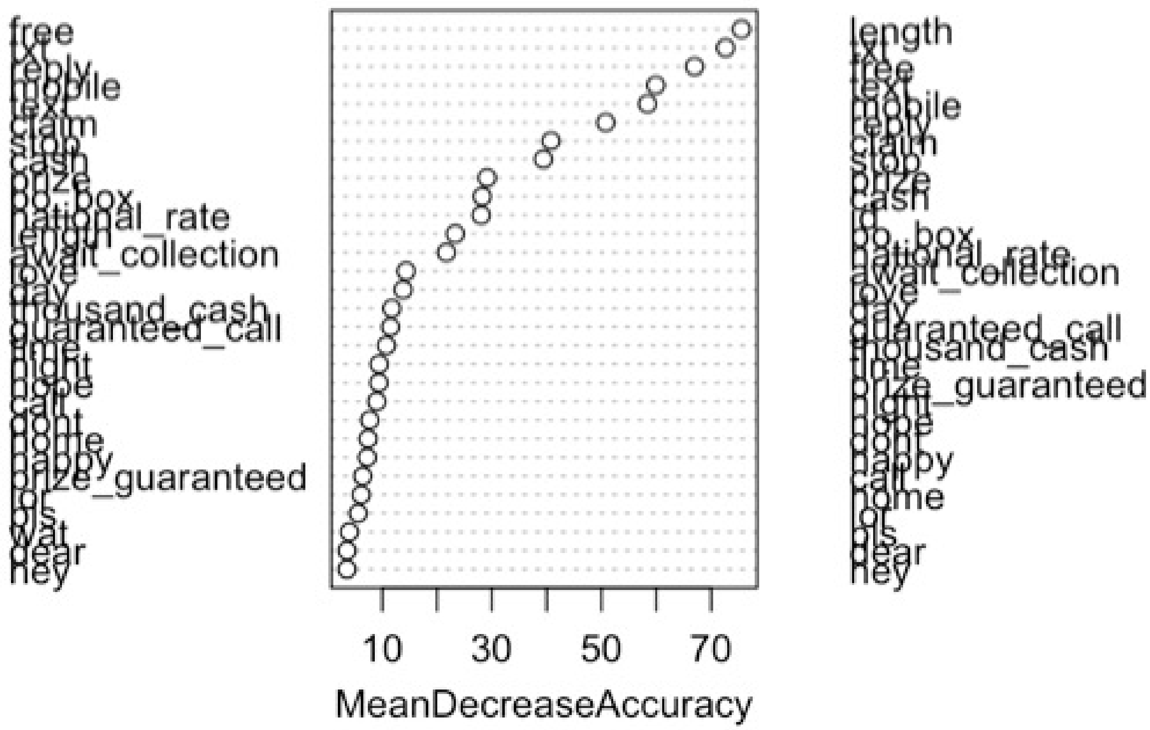
Random forest model produces 0.97 accuracy and 0.96 AUC




```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 965  33
         1   0 116

               Accuracy : 0.9704
                 95% CI : (0.9586, 0.9795)
    No Information Rate : 0.8662
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.859

 Mcnemar's Test P-Value : 2.54e-08

            Sensitivity : 0.7785
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.9669
             Prevalence : 0.1338
         Detection Rate : 0.1041
   Detection Prevalence : 0.1041
      Balanced Accuracy : 0.8893
```

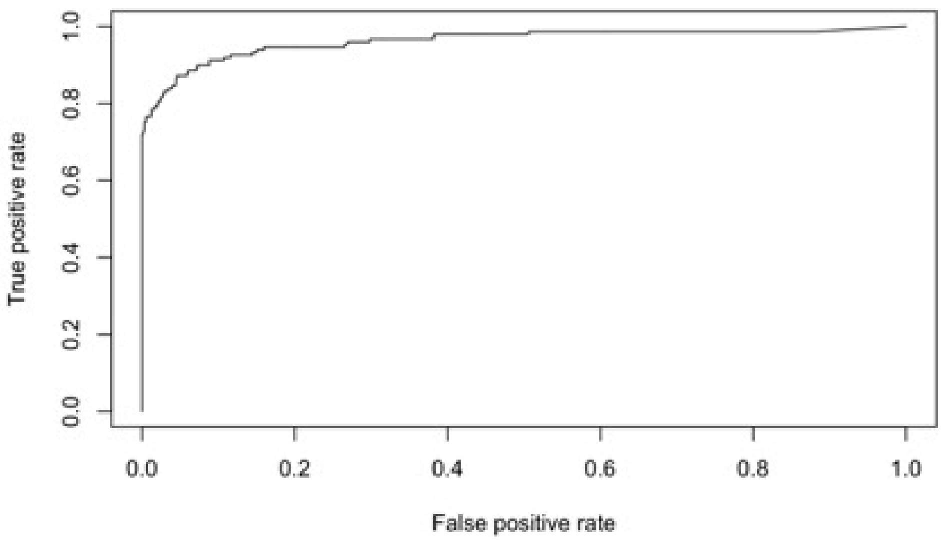
Random forest model

Importance of features



Gradient boosting model

Gradient boosting model produces 0.96 accuracy and 0.96 AUC

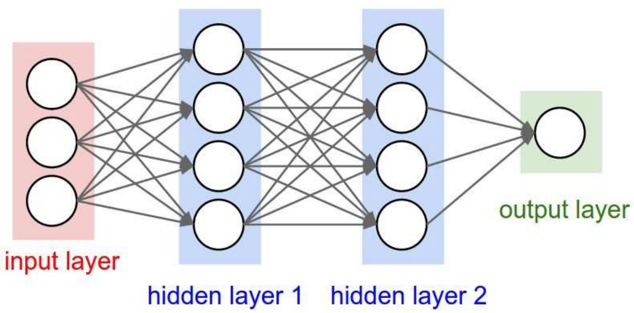



```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 965  44
         1   0 105

               Accuracy : 0.9605
                 95% CI : (0.9473, 0.9712)
    No Information Rate : 0.8662
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8052

 Mcnemar's Test P-Value : 9.022e-11

            Sensitivity : 0.70470
            Specificity : 1.00000
         Pos Pred Value : 1.00000
         Neg Pred Value : 0.95639
             Prevalence : 0.13375
         Detection Rate : 0.09425
   Detection Prevalence : 0.09425
      Balanced Accuracy : 0.85235
```

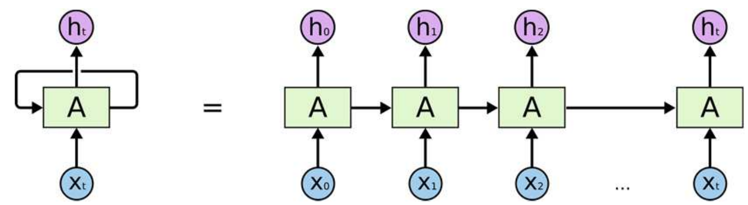
Neural Network model

What is an RNN/LSTM?

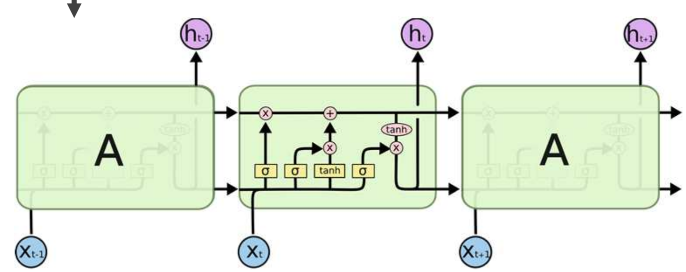
Neural Network



Recurrent Neural Network



Long Short Term Memory



Source: [The Unreasonable Effectiveness of Recurrent Neural Networks] (Andrej Karpathy)
[Understanding LSTM Networks] (Chris Olah)

Neural Network model

Environment

- Tensorflow: open source machine learning library maintained by Google
- Keras: Python library that provides high-level apis, running on top of Tensorflow (among others)
- We won't be using Tensorflow directly, but rather a version of Keras ported to R

Neural Network model

INSEAD

Text tokenization

```
max_words = 1300
max_len = 150
txts <- training$v2
tok <- text_tokenizer(num_words = max_words) %>% fit_text_tokenizer(txts)

sequences <- texts_to_sequences(tok, txts)
data <- pad_sequences(sequences, maxlen = max_len)
x_train <- data
y_train <- training$v1
```

Neural Network model

Building our model

```
input <- layer_input(
  shape = list(max_len),
  dtype = "float32"
)

layer <- input %>%
    layer_embedding(input_dim = max_words, output_
layer <- layer %>%
    layer_lstm(units = 64)
layer <- layer %>%
    layer_dropout(0.5)
layer <- layer %>%
    layer_batch_normalization()
layer <- layer %>%
    layer_dropout(0.5)
layer <- layer %>%
    layer_dense(units = 256, activation = "relu")
layer <- layer %>%
    layer_dropout(0.5)
layer <- layer %>%
    layer_batch_normalization()
layer <- layer %>%
    layer_dropout(0.5)
layer <- layer %>%
    layer_dense(1, activation = "sigmoid")
```

```
model <- keras_model(input, layer)
model %>% compile(
  optimizer = optimizer_adam(),
  loss = "binary_crossentropy",
  metrics = "accuracy"
)
```

Neural Network model

INSEAD

Building our model

```
Model
_________________________________________________________________________________
Layer (type)                                Output Shape                 Param #
=================================================================================
input_2 (InputLayer)                        (None, 150)                  0
_________________________________________________________________________________
embedding_1 (Embedding)                     (None, 150, 50)              65000
_________________________________________________________________________________
lstm_1 (LSTM)                               (None, 64)                   29440
_________________________________________________________________________________
dropout_4 (Dropout)                         (None, 64)                   0
_________________________________________________________________________________
batch_normalization_v1_2 (BatchNormalizationV1) (None, 64)               256
_________________________________________________________________________________
dropout_5 (Dropout)                         (None, 64)                   0
_________________________________________________________________________________
dense_2 (Dense)                             (None, 256)                  16640
_________________________________________________________________________________
dropout_6 (Dropout)                         (None, 256)                  0
_________________________________________________________________________________
batch_normalization_v1_3 (BatchNormalizationV1) (None, 256)              1024
_________________________________________________________________________________
dropout_7 (Dropout)                         (None, 256)                  0
_________________________________________________________________________________
dense_3 (Dense)                             (None, 1)                    257
=================================================================================
Total params: 112,617
Trainable params: 111,977
Non-trainable params: 640
```

Neural Network model

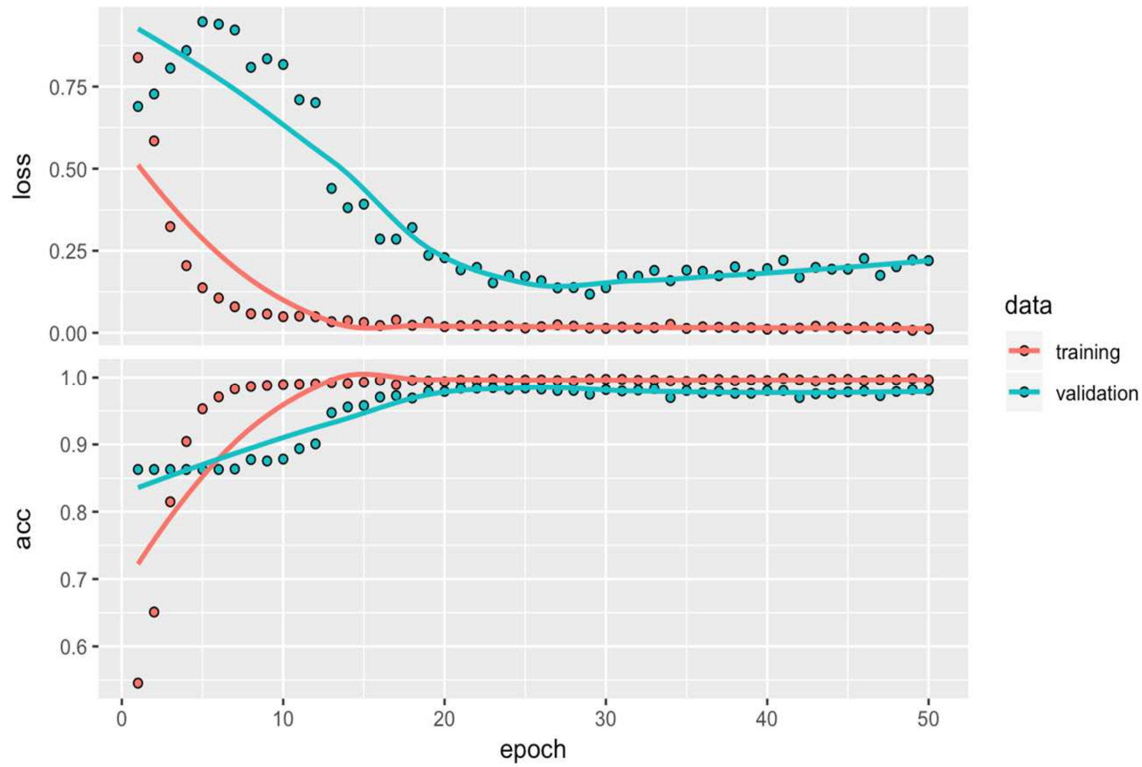
RNN tuning tips from Karpathy

- https://github.com/karpathy/char-rnn#tips-and-tricks
- 0.5MB dataset ~ 0.5M chars ~ same order of magnitude for parameters
- If validation loss ~ training loss, probably underfitting, try increasing model complexity/size
- On the other hand, if validation loss >> training loss, overfitting, try increasing dropout

Neural Network model

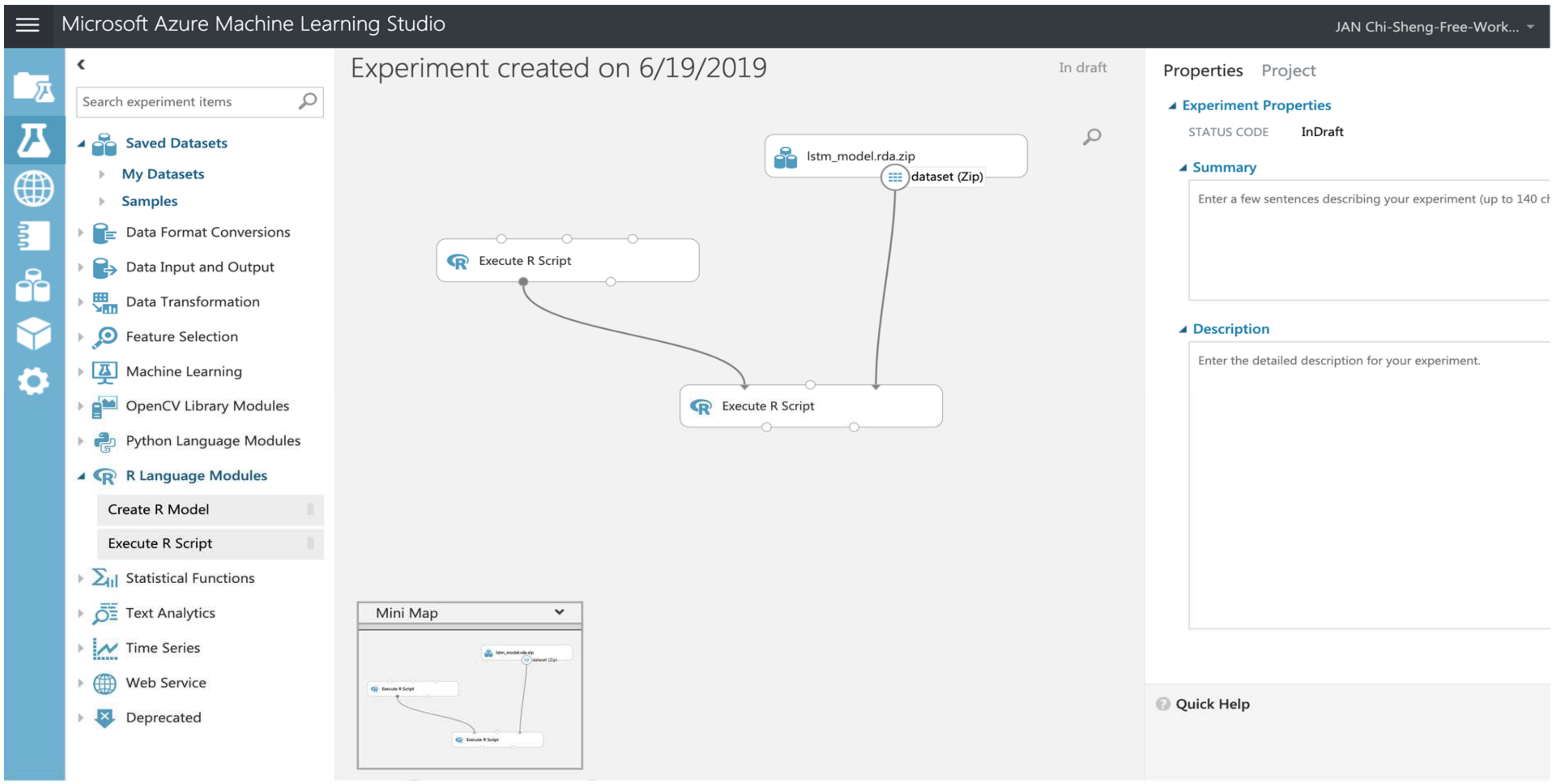
Training our model and testing holdout data

```r
model_out <- model %>% fit(
  x_train,
  y_train,
  batch_size = 128,
  validation_split = 0.3,
  epochs = 50,
  class_weight = list("0" = 1, "1" = 6)
)
```



What else? Moving to the cloud

Azure



What else? Moving to the cloud

Stopping SMS spam (in Alicloud)

Limitations and Implications

INSEAD

Model

- Potential of overfitting
- Make sure 0 false positives: cost to end user of filtering non-spam messages is high (they miss out on a real message)
- Use warning instead of filter for spam

Data

- Dataset size is too small: less than 6000 text messages
- Context based, language usage and way of communication specific to Singapore and certain community (university). Not applicable to other areas and situations

Spammers

- Spammers evolving with their technology and language (possibly also doing data analytics to avoid filtering)

Implementing the model

Conclusion

Our Process

Started with only two "variables": text and ham/spam

Established process for creating spam filter that can be applied across different contexts

1. Data cleaning
2. Text mining
3. Feature engineering
4. Neural Network

Implementation

Implementation of filter or warning app requires buy-in from either carrier, University, or student body.

Data privacy is a concern.

Economic benefits can be considerable, but more about limiting downside risk than financial gain.

Spam also seems to be an extreme externality in the sense that the ratio of external costs to private benefits is quite high. We estimate that American firms and consumers experience costs of almost $20 billion annually due to spam. Our figure is more conservative than the $50 billion figure often cited by other authors, and we also note that the figure would be much higher if it were not for private investment in anti-spam technology by firms, which we detail further on. On the private-benefit side, based on the work of crafty computer scientists who have infiltrated and monitored spammers' activity (Stone-Gross, Holz, Stringhini, and Vigna 2011; Kanich et al. 2008; Kanich et al. 2011; Caballero, Grier, Kreibich, and Paxson 2011), we estimate that spammers and spam-advertised merchants collect gross worldwide revenues on the order of $200 million per year. Thus, the "externality ratio" of external costs to internal benefits for spam is around 100:1.

Sources: The Economics of Spam, Rao & Reiley, Journal of Economic Perspectives 2012