machine learning system design

topics and issues face when designing a machine learning system. strategizing how to put together a complex machine learning system

building a spam classifier

prioritizing what to work on

build a classifier with supervised learning to classify emails as spam (1) or not spam (0)
supervised learning:
x = features of email y = spam (1) or not spam (0)
features x: choose 100 words indicative of spam/not spam

e.g. "deal "buy" "discount" "address" "now" etc...

$$y = \begin{cases} 1 \text{ if work } j \text{ appears in email} \\ 0 \text{ otherwise} \end{cases}$$

$$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ \dots \\ 1 \\ \dots \end{bmatrix}$$

address
buy
deal
discount
...
now

$x \in \mathbb{R}^{100}$

```
From: cheapslaes@buystuffromme.com
To: ang@cs.stanford.edu
Subject: Buy Now!

Deal of the week! Buy now!
```

note: in practice, take the most frequently occurring n words (10,000 to 50,000) in the training set, opposed to manually selecting 100 words as above

spending time to lower the training error

- collect large amounts of data ("honeypot projects")
- develop sophisticated features based on email routing information (headers)
- develop sophisticated features from message body (e.g. decision to treat "discount" and "discounts" as the same word; "deal" and "dealer"; punctuation features etc...)
- develop sophisticated algorithm to detect misspellings (e.g. m0rtgage, med1cine, w4tches, etc...)

calculating many different features is possible but it is difficult to determine what features will be most useful in advance
using an analyst' "gut feeling" to choose varying ideas for algorithm development is not a best practice
it is not always advantageous to spend time gathering as much data is possible before trying to initially model the problem

error analysis

the concept of error analysis to systematically improve algorithms

example

m_{cv} = 500 examples in cross validation set

algorithm misclassifies 100 emails

user manually exams the 100 errors and categorizes them based on:

(i) the type of email (pharma, replica, steal passwords, etc…)

(ii) the cues (features) that would have helped the algorithm correctly classify

pharma:	12	→	deliberate misspellings:	5
replica/fake:	4		(m0rtgage, med1cine, etc…)	
steal passwords:	53	→	unusual email routing:	16
other:	31	→	unusual (spamming) punctuation:	32

the improtance of numerical evaluation

should "discount"/"discounts"/"discounted"/"discounting" be treated alike?

will the use of "stemmiung" software (e.g. porter stemmer) benefit?

"universe"/"university"

error analysis may not be helpful for deciding if such is likely to improve performance; only method for conclusion is trial and error

numerical evaluation (e.g. cross validation error) is needed for an algorithm's performance with and without stemming

without stemming: 5% error with stemming: 3% error

distinguish upper vs. lower case (Mom/mom): 3.2%

the recommended approach to perform error analysis is by using the cross validation data to compute $J_{cv}(\theta)$ rather than the test data to compute $J_{test}(\theta)$

if new features are developed by examining the test set, chosen features are likely to work well specifically for the test set; $J_{test}(\theta)$ is no longer a good estimate of how well the algorithm will generalize to new examples

build a "quick and dirty" implementation first