Detecting Spam on Social Networking Sites

Antonio Lupher
Cliff Engle
Reynold Xin

Question

- How can social networking sites leverage user, profile and message information to curb spam and scams?
- Which features to pick when building classifiers?

Data Set

- Had access to data from InterPals.net, an international social network with 1.4 million users
- Analyzed data set of 2 million "spam" and 2 million "ham" messages from 22,080 and 71,409 accounts, respectively.
- Spam was labeled by users and verified by moderators
- Included advanced fee fraud, romance scams, advertising, etc.

Approach

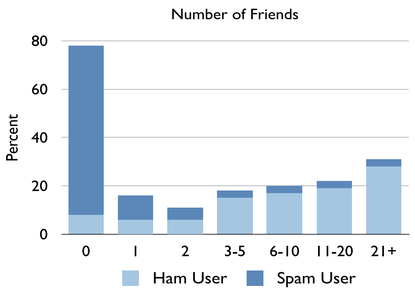
- Examined feature incidence among both classes to determine whether to include in classification.
- Use MySQL, perl, Spark, Lucene to extract features
  - Feature extraction distributed via Spark
  - Produce a feature matrix used to train classifiers

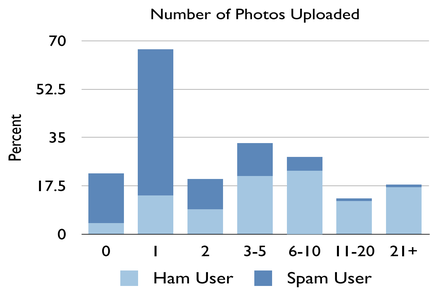
  Stage 1: Generate dictionary sorted by message frequencies in descending order

  Stage 2: Create bag-of-words representation of messages words and "expert features"
- Train and cross-validate Naive Bayes, Linear Regression and SVM classifiers

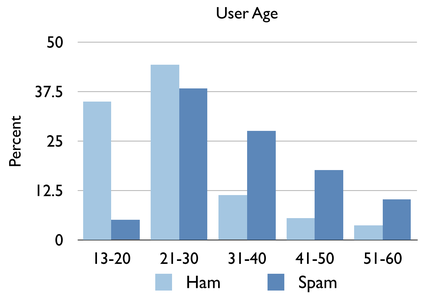
Feature Selection

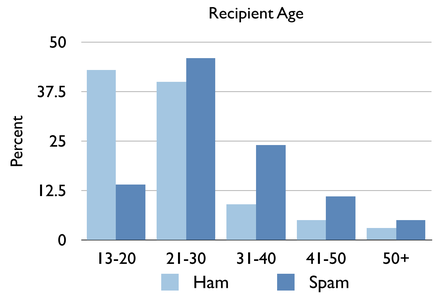
- Examined feature incidence among both classes (full 4 million msg data set) to determine whether to include in classification.

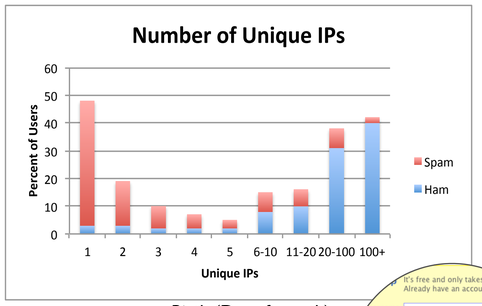


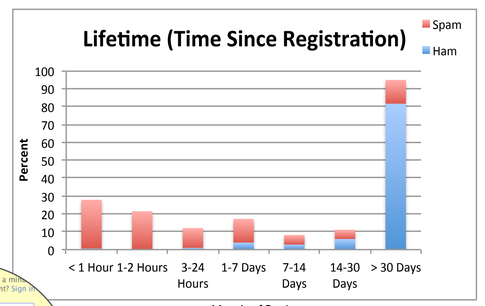


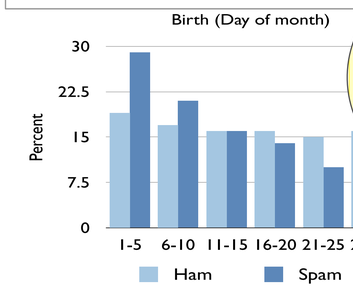
Feature Selection, cont'd

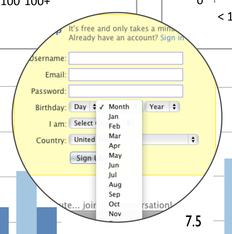


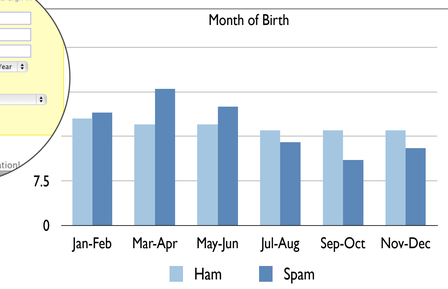


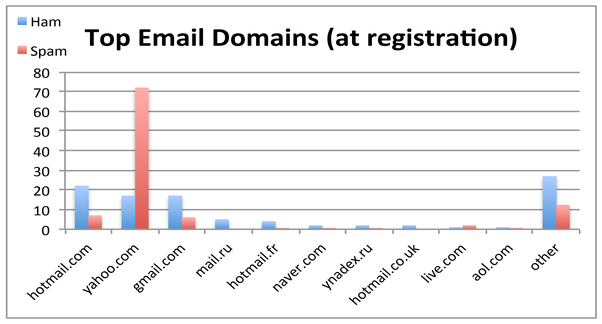


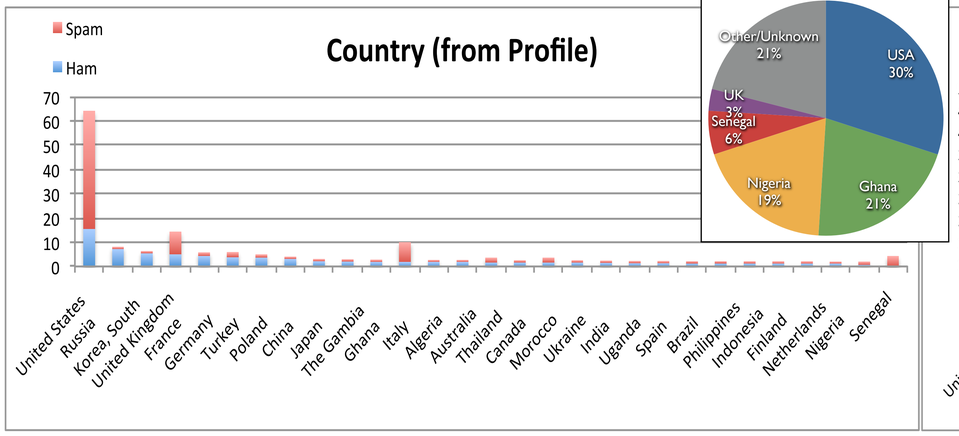


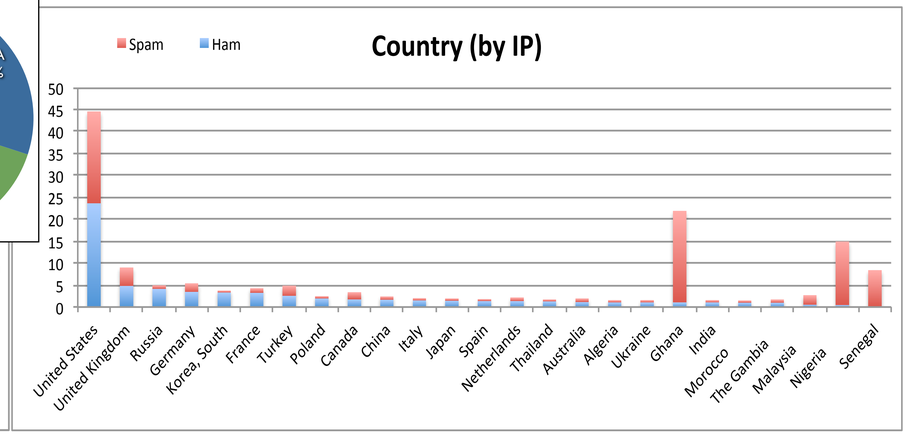








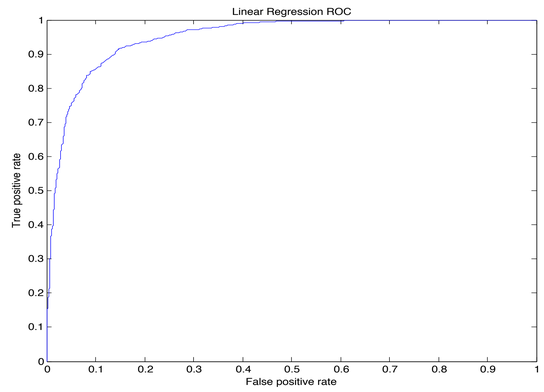




Classification & Results

- Trained & validated on subset of 22k messages.
- Used ScalaNLP for SVM & Naive Bayes; MatLab for Linear Regression

- Naïve Bayes accuracy: 82% (bag of words), 77% (expert features)
- SVM accuracy: 77% (bag of words), 51% (expert features)
- Linear Regression: mean AUC: 0.951, mean Lift: 41.58



Top Ham	Weight	Spam	Weight
hahaha	-0.1314	sex	0.2295
turkey	-0.13	cafe4tune	0.1906
learn	-0.1271	tony	0.1715
fb	-0.124	oky	0.1549
goin	-0.1186	photo	0.1467
kk	-0.1177	hehe	0.1379
uganda	-0.1164	rochelle	0.1346
new	-0.1159	register	0.1327
msn	-0.1156	irene	0.1298
ya	-0.1114	okay	0.1294
thx	-0.11	sexy	0.1278
question	-0.1095	ali	0.1199
xxx	-0.1095	displaying	0.1112
joyce	-0.1081	ghana	0.111
year	-0.1078	correctly	0.1102
hw	-0.1067	ok	0.1093
snail	-0.1067	m®	0.1087
direct	-0.1059	sand	0.1072
bonne	-0.1052	check	0.1058

Top Words (Linear Regression)

Top Ham	Weight	Top Spam	Weight
F-account_lif	-0.98	F-account_lif	0.98
F-friends	-0.37	F-friends	0.37
F-recip_age	-0.21	F-recip_age	0.21
F-age	-0.18	F-age	0.18
i	-0.05	i	0.05
you	-0.04	you	0.05
to	-0.04	to	0.04
and	-0.04	and	0.04
F-month_of_	-0.03	F-month_of_	0.03
F-country_m	-0.3	F-country_m	0.03
me	-0.02	me	0.02
am	-0.02	am	0.02
my	-0.02	my	0.02
?	-0.02	?	0.02
a	-0.02	a	0.02
your	-0.02	your	0.02
F-photo_exis	-0.01	F-photo_exis	0.01
the	-0.01	the	0.01
sex	-0.01	F-sex	0.01

Top Features (SVM)

Lessons Learned

- Naive Bayes very fast, acceptable accuracy, easier to parallelize
- SVM slow & poor results with extra features. We think the accuracy could be significantly improved with a larger data set and by pruning stop words.

Future Work

- Train & validate on entire 2M data set
- Examine other features:

  - textual similarity to previous messages sent by user (data-punctuated token trees, Jaccard index, cosine similarity)
  - browser signatures (e.g. locale, keyboard layout)
  - n-grams
  - photo fingerprinting (e.g. pHash)