Penyaring *Spam* Bayesian
(Bayesian Spam Filtering)

Agastia Cestyakara (18209029)
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
18209029@std.stei.itb.ac.id

***Abstrak*—Penyaring *Spam* Bayesian adalah teori mengenai aplikasi ilmu probabilitas dan statistika dalam bidang sistem informasi, khususnya pada proses penyaringan *spam* pada pesan elektronik. Teori ini merupakan pengembangan lebih lanjut dari Teorema Bayes yang pada dasarnya digunakan untuk menghitung peluang suatu kejadian dengan adanya sebuah data yang diketahui. Pada makalah ini akan dibahas mengenai Teorema Bayes, penerapannya dalam *Bayesian spam filtering*, dan cara kerja penyaring tersebut. Terdapat beberapa kelebihan dan kekurangan dari penerapan teori ini. Selain itu, pada makalah ini juga dijelaskan salah satu aplikasi dan sistem kerja *Bayesian spam filtering* dalam sebuah perangkat lunak komersial, yaitu SpamBayes. Perkembangan lebih lanjut dari teori ini juga dipaparkan untuk melengkapi penjelasan mengenai perangkat ini.**

***Kata Kunci*—teorema Bayes, *spam filtering*, *ham*, probabilitas**

I. PENDAHULUAN

Dalam dunia informasi, isu tentang banjirnya *spam (junk mail)* pada *email* merupakan persoalan yang cukup mengganggu dan meresahkan bagi pengguna. Pengaruhnya yaitu kotak masuk menjadi penuh oleh surat dan pesan yang tidak penting untuk dibaca. Dalam hitungan bulan saja, bisa ribuan *spam* yang masuk. Secara teknis, banjirnya *spam* mengkonsumsi *bandwidth* yang masuk, merupakan ancaman yang dapat meredam arus mail yang sesungguhnya, dan memunculkan kerepotan sistem administrasi yang sedang berlangsung.

"If you compete with slaves, you become a slave" (Norbert Wiener). Sesuai kata mutiara tersebut, apakah kita sebagai pengguna harus terus sabar membuang waktu memisahkan manakah pesan penting dan menghapus ribuan *spam*? Untuk mengatasi masalah ini telah ditemukan aplikasi yang secara umum berguna untuk mengeliminasi pesan surat yang tidak diinginkan itu sehingga dapat mencegah *spam* dalam mengacaukan utilitas dari sistem *email*.

Hubungannya dengan ilmu Probabilitas dan Statistika yaitu dasar dari aplikasi ini merupakan penerapan lebih lanjut dari teorema probabilitas Bayes dengan penyesuaian yang diperlukan.

Variasi *spam filtering* dari teori dasar Bayes ini telah diterapkan di sejumlah karya penelitian dan produk perangkat lunak komersial. Banyak pengguna *email* telah menerapkan penyaringan *spam* Bayes dengan menginstal program penyaringan email terpisah. Penyaring *email* seperti DSPAM, SpamAssassin, SpamBayes, Bogofilter dan ASSP menggunakan teknik penyaringan *spam* Bayesian, dan fungsionalitas ini kadang tertanam dalam perangkat lunak *mail server* itu sendiri.

Program *mail filtering* pertama yang diketahui menggunakan teori Bayes adalah program iFile Jason Rennie, dirilis pada tahun 1996. Program ini digunakan untuk menyortir surat ke dalam folder. Publikasi pertama tentang *spam filtering* Bayesian dilakukan oleh Sahami pada tahun 1998 dalam *spam filter* komersial. Namun, pada tahun 2002, Paul Graham berhasil meningkatkan tingkat positif palsu, sehingga dapat digunakan dengan sendirinya sebagai *spam filter* tunggal. Sebagai contoh aplikasinya, dalam makalah ini secara khusus akan dibahas sistem penyaringan *spam* Bayesian dalam perangkat lunak SpamBayes.

II. TEORI DASAR

A. TEOREMA BAYES

Teorema Bayes merupakan alat untuk menentukan bagaimana peluang yang seharusnya dari sebuah hipotesa (H), dengan adanya sebuah data (D) yang diketahui dari percobaan yang berhubungan dengan hipotesa tersebut. Sebelum dilakukan pengamatan terhadap data percobaan, hipotesa yang akan dicari peluangnya harus diyakini benar terjadi terlebih dahulu. Rumusnya sebagai berikut.

$$\mathrm{P}(H|D) = \frac{\mathrm{P}(D|H)\,\mathrm{P}(H)}{\mathrm{P}(D)},$$

Keterangan:
P(H) = *prior probability;* peluang H benar sebelum data D diamati
P(D|H) = *conditional probability;* probabilitas bersyarat yang melihat data D dengan mengingat atau meyakini bahwa H benar
P(D) = *marginal probability;* peluang terjadinya D
P(H|D) = *posterior probability;* peluang bahwa hipotesis

benar, mengingat data dan kondisi sebelumnya turut meyakini hipotesis

Berdasarkan rumus di atas, jika H adalah kejadian sembarang dalam S dengan P(D) ≠ 0, maka teorema Bayes memungkinkan kita menentukan peluang berbagai kejadian H_1, H_2, ..., Hn yang dapat menyebabkan D terjadi. Rumusnya sebagai berikut.

$$\mathrm{P}(D) = \sum_i \mathrm{P}(D, H_i) = \sum_i \mathrm{P}(D|H_i)\mathrm{P}(H_i).$$

B. *BAYESIAN SPAM FILTERING*

Perangkat lunak pendeteksi *spam* manapun, tidak mengetahui fakta mengenai representasi sebenarnya dari kata yang diberikan, yang dilakukan adalah menghitung peluang bahwa dari kata tersebut dapat diwakilkan bahwa isi keseluruhan pesan adalah *spam* atau bukan. Berikut akan dijelaskan penerapan teorema Bayes untuk menghitung probabilitas yang dapat menentukan apakah sebuah pesan yang berisi kata tertentu adalah *spam* atau bukan.

Dimisalkan terdapat kata "iklan" dalam sebuah pesan surat yang masuk. Dari kata tersebut dapat diindikasikan pesan sebagai sebuah *spam*, seperti pesan yang berisi iklan penjualan produk tertentu. Untuk menganalisa kata tersebut, digunakan rumus sebagai berikut.

$$\Pr(S|W) = \frac{\Pr(W|S) \cdot \Pr(S)}{\Pr(W|S) \cdot \Pr(S) + \Pr(W|H) \cdot \Pr(H)}$$

Pr(S|W) = peluang bahwa pesan surat adalah *spam*, dengan mengetahui adanya sebuah kata "iklan" di dalamnya

Pr(S) = peluang keseluruhan bahwa semua pesan adalah *spam*

Pr(W|S) = peluang kata "iklan" muncul dalam sebuah *spam message*

Pr(H) = peluang keseluruhan bahwa pesan yang masuk bukan *spam* (melainkan *ham*)

Pr(W|H) = peluang kata "iklan" muncul dalam sebuah *ham* message

Statistik saat ini menunjukkan bahwa kecenderungan bahwa sebuah pesan merupakan *spam* dibandingkan *ham* yaitu sebesar 80%, di mana Pr(S) = 0.8 dan Pr(H) = 0.2. Namun sebagian besar pendeteksi *spam* Bayes mengasumsikan bahwa tidak ada alasan yang cukup kuat untuk cenderung menganggap sebuah pesan sebagai *spam*, sehingga *spam* : *ham* = 0.5 : 0.5. Sistem penyaring yang menggunakan hipotesa ini disebut "tak bias", artinya pendeteksi tidak punya praduga terhadap pesan yang masuk. Rumusnya pun menjadi lebih sederhana, yaitu:

$$\Pr(S|W) = \frac{\Pr(W|S)}{\Pr(W|S) + \Pr(W|H)}$$

Untuk penyaring yang menggunakan sistem tak bias, Pr(S|W) dikenal sebagai istilah *spamicity/spaminess* dari kata "iklan", yaitu peluang kecenderungan pesan sebagai *spam* jika terdapat kata "iklan" di dalamnya. Jumlah Pr(W|S) diperoleh dari frekuensi pesan yang mengandung kata "iklan" di antara pesan-pesan yang dianggap *spam* selama proses *training*. Sedangkan Pr(W|H) diperoleh dari frekuensi pesan yang mengandung kata "iklan" di antara pesan-pesan yang dianggap *ham* selama proses *training*.

Agar perhitungan ini masuk akal, dibutuhkan jumlah yang besar untuk pesan yang akan dijadikan bahan *training* sehingga dapat merepresentasikan peluang dengan baik. Tentu saja, menentukan apakah pesan adalah *spam* atau *ham* hanya berdasarkan kehadiran dari kata "iklan" rawan kesalahan, itu sebabnya perangkat *spam* Bayesian mencoba untuk mempertimbangkan beberapa kata dan menggabungkan *spamicities* mereka untuk menentukan probabilitas secara keseluruhan menjadi *spam*. Untuk cara kerja lebih lengkapnya akan dijelaskan pada subbab berikutnya.

C. CARA KERJA DAN DASAR MATEMATIS

Pada dasarnya, setiap kata memiliki probabilitas tertentu untuk muncul dalam sebuah *spam* ataupun pesan sah. Selama proses *training* perangkat lunak, pengguna harus menentukan dulu secara manual apakah pesan merupakan *spam* atau bukan. Dalam proses ini, untuk setiap pesan masuk perangkar *spam filter* akan mempelajari dan menyesuaikan probabilitas setiap kata yang muncul sesuai kategori (jenis *spam* atau bukan) ke dalam *database*nya, sehingga kemudian dapat digunakan untuk mengidentifikasi setiap kata pada pesan selanjutnya.

Setelah proses *training*, *database* probabilitas kata digunakan untuk menghitung peluang sebuah pesan dengan kelompok kata tertentu masuk ke kategori mana. Setiap kata dalam pesan berkontribusi dalam perhitungan peluang *spam* ini (atau hanya kata-kata yang mencolok). Kontribusi setiap kata ini disebut *posterior probability* dan dihitung dengan teorema Bayes. Perangkat juga akan belajar untuk memberi peluang *spam* yang tinggi pada kata-kata mencolok (iklan, dll) dan memberi peluang yang rendah pada kata-kata biasa (nama orang, kata sapaan dalam keluarga). Kemudian peluang *spam* dihitung secara keseluruhan, dan apabila totalnya melebihi batas (misalnya 95%), maka *spam filter* akan menganggapnya sebagai sebuah *spam*.

Selanjutnya, pesan yang dianggap sebagai *spam* dapat secara otomatis dipindahkan ke folder *junk mail* atau bahkan langsung dihapus. Namun beberapa perangkat lunak menyediakan *quarantine mechanism* yang menetapkan rentang waktu di mana pengguna dapat

melihat keputusan perangkat lunak terhadap peasn-pesan terdahulu.

Training awal pada perangkat dapat diperbaiki kemudian ketika keputusan yang salah diketahui (*false positive* atau *false negative*). *False positive* (“fp” atau “FP”) adalah kesalahan di mana pesan yang sah diklasifikasikan sebagai *spam*, sedangkan *false negative* (“fn” atau “FN”)adalah kesalahan di mana pesan *spam* diklasifikasikan sebagai pesan sah. Hal ini akan membantu perangkat untuk beradaptasi secara dinamis sesuai perkembangan *spam*. Beberapa *spam filter* menggunakan kombinasi antara Bayesian *spam filtering* dan heuristic lainnya, dan hasilnya yaitu penyaringan yang lebih akurat.

Untuk konteks *spam*, *email filter* Bayesian menggunakan teorema Bayes dalam beberapa tahap, yaitu:

- Pertama, untuk menghitung peluang bahwa pesan merupakan *spam*, dengan mengetahui bahwa kata tertentu muncul dalam pesan.
- Kedua, untuk menghitung peluang bahwa sebuah pesan adalah *spam*, dengan mempertimbangkan semua kata-katanya (atau bagian yang relevan dari kata-kata tersebut)
- Ketiga, terkadang digunakan untuk berurusan dengan kata-kata yang langka

III. KELEBIHAN DAN KEKURANGAN

Kelebihan dari *spam filtering* Bayesian adalah:

1. Perangkat dapat dilatih pada basis setiap pengguna
2. Kriteria *spam* yang diterima pengguna biasanya berkaitan dengan aktivitas *online* pengguna.
   - Sebagai contoh, pengguna mungkin telah berlangganan Koran *online* dan kemudian akan mempertimbangkannya sebagai *spam*. Koran ini mungkin mengandung kata-kata umum yang umum dipakai dalam bahasa media cetak (merk Koran, alamat asalnya, dll). *Spam filter* Bayesian ini secepatnnya menetapkan probabilitas tinggi berdasarkan pola pengguna secara spesifik.
3. Kriteria pesan sah yang diterima pengguna cenderung berbeda-beda.
   - Sebagai contoh, dalam lingkup perusahaan, nama perusahaan atau nama pelanggan akan sering muncul dalam pesan. Maka *filter* akan menetapkan probabilitas rendah untuk pesan yang mengandung nama-nama tersebut.
4. Probabilitas setiap kata adalah unik untuk setiap pengguna sesuai kebiasannya, dan representasi ini dapat berkembang sepanjang waktu dengan *training* yang tepat kapanpun *filter* salah dalam mengklasifikasikan pesan. Sebagai hasilnya, letepatan *spam filter* Bayesian yang telah di*training* akan menjadi lebih akurat dibandingkan jika aturan probabilitas telah ditentukan sebelumnya.
5. Akurasi. *Spam filter* dapat terhindar dari *false positive* (pesan dianggap sebagai *spam*).
   - Sebagai contoh, apabila pesan mengandung kata “Nigeria” yang biasa dipakai dalam penipuan uang muka, jika pada awalnya di dalam perangkat telah ditentukan aturan khusus, mungkin pesan itu secara langsung akan ditolak. Namun *filter* Bayesian akan menganggap “Nigeria sebagai kata yang mungkin *spam* tetapi juga akan memperhitungkan kata penting lain yang biasa dipakai dalam pesan sah. Sehingga apabila kata yang sah ini lebih kuat dalam mengindikasikan pesan sebagai pesan yang sah, maka penggunaaan kata “Nigeria” dapat diatasi.

Kerugian dari *spam filtering* Bayesian adalah:

1. *Spam filter* Bayesian rentan terhadap *Bayesian Poisoning*, yaitu teknik yang digunakan oleh *spammer* dalam upaya untuk menurunkan efektivitas *filter spam* yang bergantung pada penyaringan Bayesian. *Spammer* akan mengirimkan pesan dengan teks sah dalam jumlah besar (dikumpulkan dari berita yang sah atau sumber sastra) termasuk penyisipan kata acak yang tidak berbahaya yang biasanya tidak berhubungan dengan *spam*, sehingga mengurangi nilai *spam* dari pesan, menyebabkan pesan lebih mungkin untuk tergelincir saat melewati *filter spam*.

2. Teknik lain yang digunakan untuk meperdaya *spam filter* Bayesian adalah mengganti teks dengan gambar, baik langsung disertakan atau melalui *link*. Semua teks dari pesan atau beberapa bagian dari itu, diganti dengan gambar.*Spam filter* biasanya tidak dapat menganalisis gambar ini. Namun, karena banyak *mail server* menonaktifkan tampilan gambar terkait untuk alasan keamanan, *spammer* yang mengirimkan gambar melalui *link* jauh bisa mencapai target yang lebih sedikit. Selain itu, ukuran gambar memiliki bit yang lebih besar dari ukuran teks, sehingga *spammer* membutuhkan *bandwidth* lebih untuk mengirim pesan langsung termasuk gambar. Akhirnya, beberapa *filter* lebih cenderung untuk memutuskan bahwa sebuah pesan adalah *spam* jika memiliki isi mayoritas grafis.

Solusi yang efisien telah diterapkan yaitu dengan menggunakan OCR (*Optical Character Recognizer*) untuk menganalisis setiap foto berukuran menengah ke besar dan juga menganalisis teks di dalamnya.

IV. PERANGKAT LUNAK SPAMBAYES

SpamBayes adalah *spam filter* Bayesian yang ditulis dengan Python menggunakan teknik yang diterapkan oleh Paul Graham dalam esainya "*A Plan for Spam* ". Hal ini kemudian telah diperbaiki oleh Gary Robinson dan Tim Peters.

Perbedaan paling menonjol antara *filter* Bayesian konvensional dengan *filter* yang digunakan oleh

SpamBayes adalah adanya tiga klasifikasi: *spam*, *non-spam* (dalam SpamBayes disebut *ham*), dan klasifikasi tidak yakin. Sistem klasifikasi ini berbeda dengan *filter* Bayesian konvensional yang hanya mempunyai dua klasifikasi.

Proses kerja dalam SpamBayes terbagi menjadi beberapa tahap, yaitu:

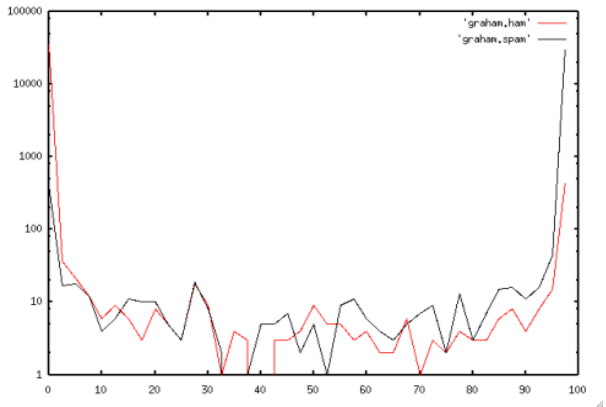
1. *Tokenizing*

Arsitektur dari sistem SpamBayes memiliki beberapa bagian yang berbeda. Perbedaan paling jelas adalah *tokenizer*, yaitu proses mengambil pesan *email* dan mengelompokkannya ke dalam serangkaian token (kata). Pada tahap ini terjadi pembagian kata-kata dari bagian teks pesan, mengeluarkannya sebagai potongan HTML, dan potongan lain (gambar, dll). Selain itu, terdapat pula berbagai penafsiran dan *tokenizing* terhadap *mail header*. Kode untuk melakukan *tokenizing* terdapat dalam `tokenizer.py`, dan komentar di bagian t*okenizer* dari `Options.py` berisi informasi lebih lanjut tentang berbagai pendekatan untuk *tokenizing*.

2. Pemberian Nilai dan Penggabungan

Tahap berikutnya dalam proses *filtering* adalah *scoring* dan penggabungan. Pada tahap inilah perhitungan matematis dan statistika dimulai.

Penggabungan menggunakan skema Graham memiliki sejumlah masalah, yaitu cenderung menghasilkan skor 1 (*spam*) atau 0 (*ham*), dan celah yang sangat kecil di antaranya, namun hal itu tidak sering diklaim sebagai "tidak yakin", sehingga timbul kesalahan. Plot berikut menunjukkan masalahnya:



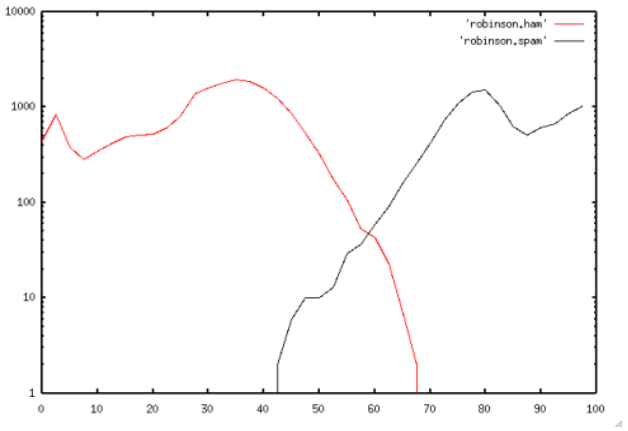
Gambar 1: Skema Graham untuk nilai *spam* dan *ham*

Catatan:

Sumbu X menunjukkan nilai dari pesan, dengan skala untuk 0-100. (Di mana 0 adalah "pasti *ham*", dan 100 adalah "pasti *spam*"), dan sumbu Y menunjukkan jumlah pesan dengan nilainya (skala logaritmis). Perhatikan bahwa plot tidak berasal dari set data yang sama, melainkan plot yang khas dari teknik yang diberikan.

Dalam plot di atas dapat dilihat kebanyakan *spam* mencapai nilai hampir 100, sedangkan kebanyakan *ham* mencapai nilai hampir 0. Keadaan ini masih normal. Sayangnya, terdapat pula beberapa *ham* dengan skor hampir 100 dan *spam* dengan skor hampir 0. Hal ini menunjukkan bahwa sistem secara signifikan melakukan kesalahan penilaian (perhatikan bahwa perbedaan tidak seharusnya terlihat demikian, karena skala grafik merupakan skala algoritmik).

Esai dari Gary Robinson muncul pada tahap ini. Teknik dari Gary menghasilkan skema plot sebagai berikut.

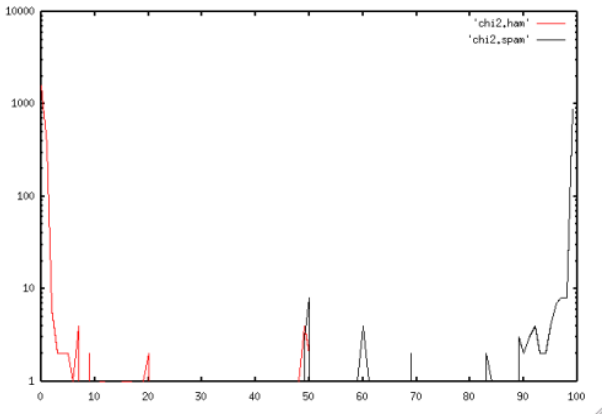


Gambar 2: Skema Robinson untuk nilai *spam* dan *ham* dengan *pendekatan Teorema Limit Pusat*

Gary menghasilkan sejumlah pendekatan alternatif untuk menggabungkan dan mencetak probabilitas kata, antara lain yaitu pendekatan dengan menggunakan Teorema Limit Pusat.

Teorema Limit Pusat menggabungkan skema dan menghasilkan beberapa hal yang menarik, yaitu dua nilai internal, satu untuk *ham* dan satu untuk *spam*.Dua nilai internal ini memmungkinkan sistem untuk memberikan respon keluaran “tidak yakin” ketika nilai *ham* dan *spam* keduanya sangat rendah atau keduanya sangat tinggi. Hal ini menyebabkan kebingungan seperti yang kita mencoba untuk memetakan hasil dengan skor seperti Graham.

Pendekatan lain yang digunakan yaitu dengan menggunakan *Chi-Squared*, namun hasil yang diberikan akan lebih sensitif. Pendekatan chi-squared menghasilkan dua angka; "probabilitas *ham*" ("* H *") dan "probabilitas *spam*" ("* S *").



Gambar 3: Skema Robinson untuk nilai *spam* dan *ham* dengan pendekatan *Chi-Squared*

Sebuah pesan dengan kecenderungan *spam* akan memiliki *S* tinggi dan *H* rendah, sementara pesan dengan kecenderungan *ham* akan memiliki *H* tinggi dan *S* rendah. Dalam kasus di mana pesan yang terlihat sama sekali tidak sesuai dengan bagaimana sistem sudah terlatih, maka nilai bisa berakhir dengan *H* rendah dan *S*. Dalam hal ini kode memberikan respon "tidak tahu". Beberapa pesan juga dapat memiliki baik *H* tinggi dan *S* tinggi, yaitu bahwa pada pesan tersebut banyak terdapat kata yang terlihat terlihat seperti *ham*, tetapi juga sangat banyak seperti *spam*. Dalam kasus ini SpamBayes juga tidak yakin di mana pesan harus diklasifikasikan, dan skor akhir akan dekat 0.5. Plot berikut ini menunjukkan dengan cukup jelas hasil yang didapatkan dari teknik ini.

Jadi pada akhir proses, terdapat tiga hasil yang memungkinan: "*Spam*", "*Ham*", atau "Tidak Yakin". Dan memungkinkan pula untuk mengatur tinggi rendahnya penggalan daerah "tidak yakin" untuk memberikan alternatif antara "tidak yakin" melawan *false positive* atau *false negative*. Pada hasil chi-kuadrat, daerah "tidak yakin" bisa sangat besar, namun tetap menghasilkan jumlah yang sangat kecil untuk pesan "tidak yakin".

3. *Training*

Training adalah proses memasukkan pesan ke dalam *tokenizer* untuk menghasilkan probabilitas yang akan dipakai oleh penentu klasifikasi sehingga kemudian dapat dihasilkan keputusan. Skrip yang digunakan untuk melakukan *training* ini adalah `sb_mboxtrain.py`. Pendekatan mengenai *training* menghasilkan hal-hal berikut:

a) Sistem tetap memberikan hasil yang baik walaupun *training* dilakukan pada pesan dalam jumlah kecil, walaupun tentu saja *training* pada pesan dalam jumlah besar akan memberikan hasil yang lebih baik.

b) Ketidakseimbangan antara jumlah *spam* dan *ham* dalam *training* dapat menimbulkan masalah (masalah yang timbul semakin besar tergantung timpangnya keseimbangan jumlah keduanya). Sistem bekerja dengan baik apabila *training* dilakukan dalam jumlah *spam* dan *ham* yang kurang lebih seimbang.

c) Sangat penting untuk berhati-hati dalam memilih *setting* untuk *training* yang berasal dari satu sumber, misalnya *spam* baru dan *ham* lama. Sistem akan cepat memutuskan jika pesan berasal dari tahun 2001 akan diklasifikasikan sebagai *ham*. Sehingga perlu dipastikan *tokenizer* tetap mendapat petunjuk akan hal ini.

4. Pengetesan

Satu perbedaan besar antara SpamBayes dan banyak proyek *open source* lainnya adalah dilakukannya sejumlah besar pengujian. Sebelum dilakukan perubahan pada *tokenizer* atau algoritma diperiksa, perlu untuk menunjukkan bahwa perubahan itu benar-benar menghasilkan perbaikan bagi sistem. Banyak ide-ide baik yang tampaknya akan memberikan perbedaan positif terhadap hasil ternyata justru tidak berdampak apa-apa, atau bahkan membuat hal-hal bertambah buruk.

Mengenai aturan umum yang sudah ada sebelumnya di dalam sistem, apabila tetap dipertahankan hanya akan membuat kesan "*Stupid beats Smart*" (dalam hal ini, *spam* adalah pihak yang pintar).

Arsitektur *testing* telah dilakukan sejak awal proyek. Skrip tes yang paling menyeluruh adalah skrip validasi silang oleh Tim, yaitu `timcv.py`. Script ini membutuhkan beberapa set *spam* dan *ham*. Skrip ini melakukan *training* pada semua pesan, dari set pertama *spam* dan *ham*, kemudian menghapus set pertama tersebut, dan kemudian memeriksa bagaimana kelanjutannya melalui *rating* mereka. Kemudian mengulangi untuk setiap set sesuai gilirannya, melakukan *training* pada semua pesan, kemudian menentukan set yang diuji, baru kemudian menguji set.

V. DAFTAR ISTILAH

1. Bayesian
   A form of statistical analysis used (in a form) in Paul Graham's initial "Plan for Spam" approach. Now used as a kind of catch-all term for this class of filters, no doubt horrifying statisticians everywhere.
2. *false negative*
   Spam yang salah pengklasifikasian sebagai *ham*, biasa disingkat "fn" atau "FN".
3. *false positive*
   Ham yang salah pengklasifikasian sebagai *spam*, biasa disingkat "fp" atau "FP".
4. *ham*
   Lawan dari istilah *spam*; pada SpamBayes diklasifikasikan sebagai pesan sah berdasarkan kriteria yang sesuai dengan *classifier*.
5. *spam*
   Secara umum disebut sebagai pesan yang tidak diinginkan oleh pengguna yang dikirim secara otomatis kepada banyak orang pada waktu yang sama. Pada SpamBayes diklasifikasikan sebagai pesan sampah berdasarkan criteria yang sesuai dengan *classifier*
6. *training*
   Proses pemasukan pesan (baik *spam* atau ham) ke dalam perangkat SpamBayes untuk diklasifikasikan lebih lanjut.

VI. KESIMPULAN

Teorema Bayes yang merupakan bagian dari ilmu probabilitas dan statistika dapat diaplikasikan dalam bidang sistem informasi. Contohnya adalah sebagai penyaring *spam* pada pesan elektronik. Penerapan Teorema Bayes pada *spam filtering* ini memiliki beberapa kelebihan dan kekurangan selama perkembangannya.

Pada masa ini, telah banyak perangkat lunak komersil yang telah menerapkan penyaring *spam* Bayesian untuk membantu pengguna, salah satunya adalah SpamBayes. Perbedaan utama dari *filter* pada perangkat ini dengan

filter konvensional adalah adanya 3 macam klasifikasi pada hasil penyaringan pesan.

Proses kerja penyaringan dalam perangkat lunak *Spam* Bayes terdiri dari 4 tahap, yaitu *Tokenizing*, Pemberian Nilai dan Penggabungan, *Training*, dan Pengetesan.

VII. REFERENSI

[1] http://spambayes.sourceforge.net/background.html, waktu akses 16 Desember 2010.
[2] http://en.wikipedia.org/wiki/Bayesian_*spam_filtering*, waktu akses 16 Desember 2010.
[3] http://www.paulgraham.com/*spam*.html, waktu akses 16 Desember 2010.
[4] http://en.wikipedia.org/wiki/Naive_Bayes_classifier, waktu akses 16 Desember 2010.
[5] http://spambayes.sourceforge.net/docs.html, waktu akses 16 Desember 2010.
[6] Munir, Rinaldi, Bahan Kuliah II2092 Probabilitas dan Statistik: Beberapa Hukum Peluang, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.
[7]Walpole, Ronald E., dkk. Probability and Statistics for Engineers and Scientists, 8th Edition, Pearson Education, Inc.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 17 Desember 2010

Agastia Cestyakara
18209029