Aberdeen, D., Pacovsky, O., and Slater, A., The Learning Behind Gmail Priority Inbox, In LCCC: NIPS 2010 Workshop on Learning on Cores, Clusters and Clouds, 2010.

羽藤研秋季集中論文ゼミ2日目　#7
2011/10/17 16:20-
発表者：M2柿元

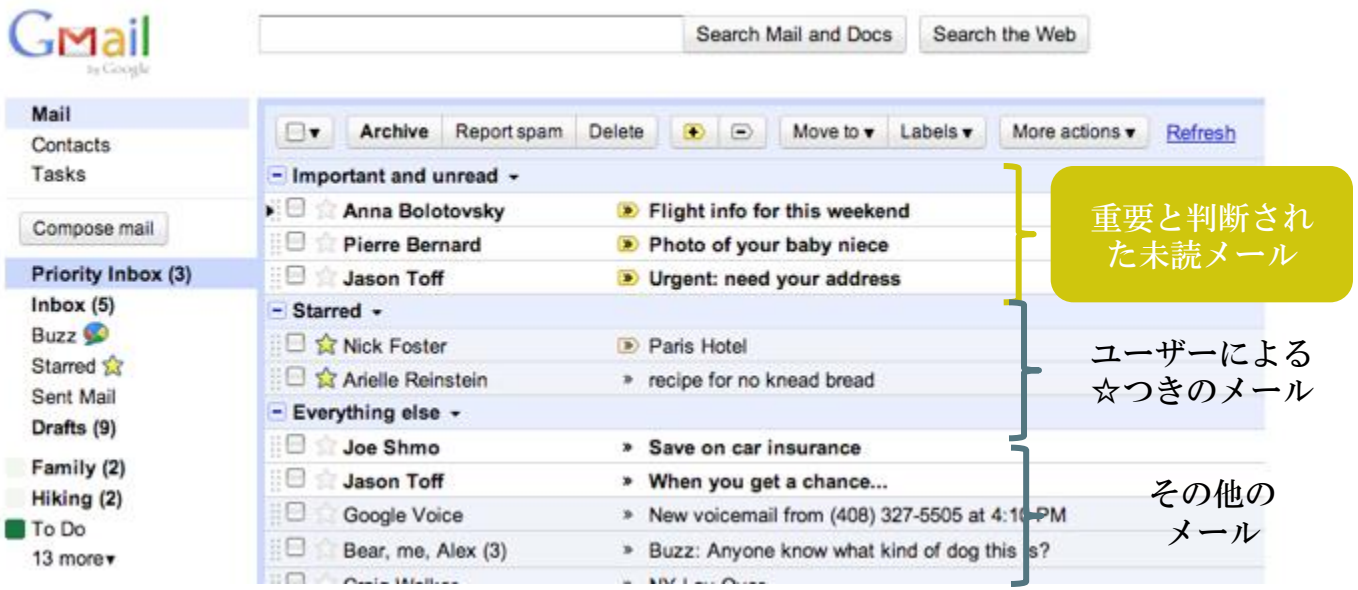
目次

Abstract

Gmail優先トレイは、ユーザーのメールに対する行動を確率的に予測し、ランク付けしている。「重要度」というのは個人性が高いため、ユーザー毎に統計モデルの学習によってこの予測を試みた。ここでは、100万ものモデルのオンライン学習とそれらの効率的な構築を試みている。

Gmail 優先トレイ

- Googleが提供するGmail新機能。
- ユーザーにとって重要なメールを自動的に選び出し、受信トレイのトップに表示する.
- 振り分けられたメールに対して、ユーザー自身が再分類しなおすこともできる.



1. The Gmail Priority Inbox

- Gmail優先トレイは、ユーザーの重要度に応じてメールを振り分けてくれる。
- このような順位づけのタスクは目新しいものではない。
- しかし、リアルタイムでの更新、日々数百万ものモデルを数秒単位で更新し続ける、といった事象でこのタスクが複雑化している。
  - 例えば、
  - ユーザーにとってそのメールが重要かどうか、判断基準がユーザーから明示的に得られない。
  - 非定常でノイズの多い訓練データを取り扱うメソッドを構築すること
  - 訓練データの制約を少なくするモデルを構築すること
  - Terabyteに上るユーザー毎の特徴データを保管し処理すること

2. The Learning Problem

2.1 Features

- メールの特徴は以下の4カテゴリに分類できる。
- Social features
  - メールの受信者・送信者間のやり取りに関する情報（例：メールの返信率）
- Content features
  - ヘッダーや本文に関する情報
- Thread features
  - スレッド情報
- Label features
  - ユーザーが適用しているラベル情報（フィルターなどの使用）
- メールをランク付けするために特徴の評価値を計算し、その後の学習の為にこれらの値を保管しておく。
- 連続的な特徴は自動的に2値の特徴に分割される。
  - Simple ID3 style algorithmで、特徴のヒストグラムに従い分割。

2.2 Important Metric

- 優先トレイの目的はユーザーの明示的なラベリング無しにメールをランク付けすること。
  - ランク付けの基準は、ユーザーがメールに対してどのような行動をとるか。
- ユーザーがメールに対して（メールが届いてT秒以内に）いかなる行動を起こすかを確率的に求めたい。
- 確率モデルの定義

$$p = \Pr(a \in A, t \in (T_{min}, T_{max}) | \boldsymbol{f}, s)$$

時間tの時、aという行動をとる確率はメールの特徴とユーザーがGmailにアクセスしたかどうかに依る.

  - ここで、a…メールに対する行為、
    A…重要度を示す行為の集合。例えば、open, replies, manual correctionsなど。
    t…メールが届いてから行動を起こすまでの時間
    f…メールの特徴を表すベクトル
    s…ユーザーがメールを見る機会を持つことがあったかどうか

2.2 Important Metric

- 予測誤差の定義
  - T_{min}…届いたメールにアクションを起こすまでの最小時間（24時間以内）。
  - T_{max}…やり取りに必要と思われる期間、またメールの保管・処理が可能な期間。

$$e = \begin{cases} 0 & if \neg s \vee t \notin (T_{min}, T_{max}) \\ 1 - p & if\ a \in A; \\ -p & otherwise \end{cases}$$

メールを開ける機会がなかった　or
T_{min},T_{max}間に行動を起こさなかった

メールに対し何らかの行動を起こした

その他

2.3 Models

- ユーザーがメールに対して行動を起こす確率を以下のように定式化。
- 線形ロジスティック回帰モデル(Linear logistic regression model)を使う。
- global model（全ユーザー共通のモデル）とuser model（ユーザー固有のモデル）を足し合わせることで、global modelの膨大なデータとuser modelのデータ不足を解消する。

$$s = \sum_{i=1}^{n} f_i g_i + \sum_{i=1}^{n+k} f_i w_i$$

$$p = \frac{1}{1 + exp^{-s}}$$

ユーザーが何らかの行動を起こす確率

- p: ユーザーが行動を起こす確率
- n: 特徴次元数
- k: ユーザーの特徴次元数
- **g**: global modelの重みベクトル
- **w**: user modelの重みベクトル

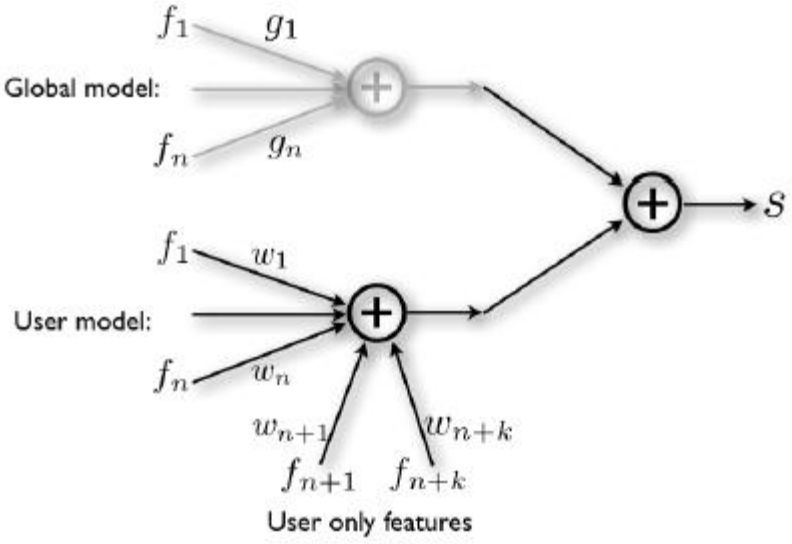
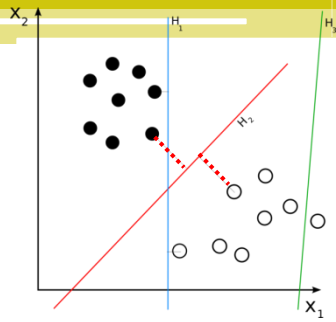


Figure 1: Adding personal and global model scores.

2.3 Models



- 大量のノイズを処理する為に、重みベクトルの学習にはオンライン学習であるPA-IIを利用。
  ※Online Passive-Aggressive Algorithms
- メール一通につき、メール受取時に一度だけglobal modelとuser modelの重みベクトルが更新される。
- i回更新された重みベクトル（user modelの場合）は、

$$w_i \leftarrow w_i + f_i \frac{sgn(e)\max(|e| - \epsilon, 0)}{\|\mathbf{f}\|^2 + \frac{1}{2C}}$$

- e: 誤差値
- C: 正規化パラメータ.更新の”aggressive”さを調整する。
  ラベリングの確信度みたいなものを表すためにメールごとに調節。
- ε: hinge-loss toleranceもしくは “passive”さを示す。

- Cが大きいほど学習による更新幅が大きくなり、直前のメールからの学習に予測がひきづられやすくなる。
  - Cはuser modelの方がglobal modelより大きく、直近のuser modelは大きい値をとる。
  - 十分に学習されていないuser modelはCを大きくしている。（学習を促進させるため）

※補足　Passive aggressive
・オンライン学習の枠組みのひとつ
・損失関数　　マージン（分割する超平面との距離）

$$\ell(\mathbf{w};(\mathbf{x},y)) = \begin{cases} 0 & y(\mathbf{w}\cdot\mathbf{x}) \geq 1 \\ 1 - y(\mathbf{w}\cdot\mathbf{x}) & \text{otherwise} \end{cases}$$

・この設定でround1での更新は、次の条件付き最適化問題となる。

$$\mathbf{w}_{t+1} = \underset{\mathbf{w}\in\mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|^2 + C\xi^2 \quad \text{s.t.} \quad \ell(\mathbf{w};(\mathbf{x}_t, y_t)) \leq \xi.$$

Aggressive　更新を促進する

$$w_{t+1} = w_t + \tau_t y_t x_t$$

$$\tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|^2 + \frac{1}{2C}}$$

Xt 入力データ
l_t 損失関数
C パラメータ（正）

Passive

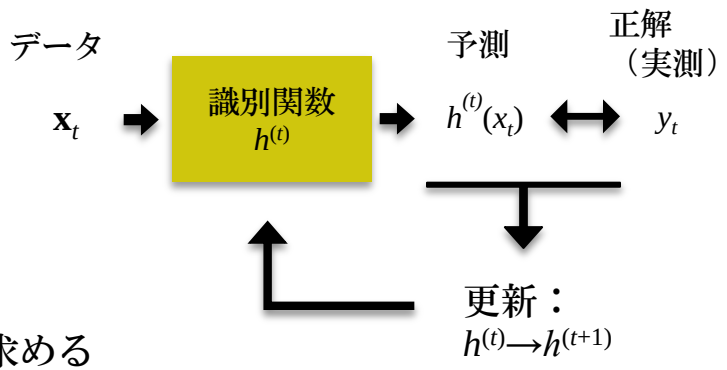
$$w_{t+1} = w_t$$

損失がない場合
τ=0

補足：オンライン学習

- データを１つづつ読み込んで、それまでの学習結果を更新する。
- 2つの利用局面
  1. データ全体は保持しているが、学習を1データ毎に行う
  2. データが１こずつ時系列としてやってくる
     ※この場合はストリームという.
- 訓練データを受け取る毎に簡単なパラメータ更新を行うだけで良いので、計算時間やメモリの効率が良い。

- オンライン学習の定式化
- 以下1,2,3を時刻 t=1,2,…,Tで繰り返す。
1. 時刻tにおいて、
   仮説h_t（ここでは重みベクトル$\mathbf{w}$)、
   入力データx_t、
   正しい結果データy_tが与えられる。
2. 仮説h_t による結果$h^{(t)}(x_t)$を計算し、
   その後でy_tとの比較を損失関数ℓによって行う。
   つまり損失関数$\ell(h^{(t)},(x_t, y_t))$を計算.
3. 損失関数ℓの値に応じて$h^{(t)}$ を更新し、新しい仮説 $h^{(t+1)}$を求める
   最終的な目的は累積損失 $\sum_{t=1}^{T}\ell\left(h^{(t)},(x_t,y_t)\right)$ を最小化すること.



2.4 Ranking for Classification

- さらに、前述したsに閾値を個人毎に設定した。
  - どのメールが重要で、またはそうではないのか判断するため。
  - 個人毎に適切に閾値を設定するのは困難である。
  - メールを開くことは重要度が高いことを示すとしたが(2.2)、実際は「重要だから」開くのではなく「興味を惹かれて」開くことの方が多い。
  - また、重要なメールを重要でないと判断される（誤検出）ことは、ユーザーにとって非常に困る事態である。（逆はそうでもない）
  - 重要なメールに判定されるメールの量は、ユーザーによって大きく異なる。
- ユーザーが閾値を調節できるよう、ある程度の干渉できるようにした。
  - ★マークなどをつける
  - ユーザーが一定の基準で重要マークを使用していることが認められたら、閾値を更新する。

3. Production

- 100万ものユーザーの学習に拡大することは困難。
- モデルを保持・管理するためにbigtable※1に改良を加えた形で利用。

3.1 Prediction Time

※1 big table…Googleの大規模なサーバー上の大量なデータを管理する為に設計されたデータベースシステム

- 全てのデータセンターから、全てのユーザーのスコアリングが可能な設計が求められる。
- どのデータセンターがどのユーザーアカウントを扱うのか予測することは困難。
- bigtableは、ランキングを行うためのモデルの複製・更新を行うために用いられ、モデルの更新に用いられるリソース管理とリアルタイムでの実行を実現した。
- 詳しく言うと、メールの特徴とそのメールに対するユーザー行動を統合。
- ユーザー単位で同じレコードにデータを管理

3.2 Learning

- Sharding(=データを複数サーバーに分割して保持すること）してモデルの学習を行うことで、実行を容易にしている。各コアがuser modelのfractionをそれぞれ更新している。
- Bigtableはスコアリングされたuser:message-idレコードにグローバルアクセスすることで、効率的なデータ管理を実現。
- 100万ものユーザーのモデルの読み書きをネットワーク上で行えないので、RAMに可能な限り多くのモデルをローディングし、更新作業をひとまとまりに処理することが必要。

- 以上の作業により、最終的にコア1台ごとの単位時間あたり（秒）35名を計算可能にした。

4. Results

- Global modelの対数オッズスコアのヒストグラ
  - 緑は「重要な」メール、赤は「重要でない
- 線形回帰モデルを使用すると、自然なランキ
  証。

Combination	*Error*
Global model	45%
User models	38%
User models & thresholds	31%

Table 1: Error rates on user marked mail.

- 閾値の設定により、「検出漏れ」が「誤検出
- メールの重要度の判断基準に関しては、研究
- Each user modelはglobal modelよりはるかに
- Google従業員による実験の結果。Gmail Prio
  メール処理に費やされる時間を6%短縮させ
- 重要でないメールに限ると、13%短縮させた

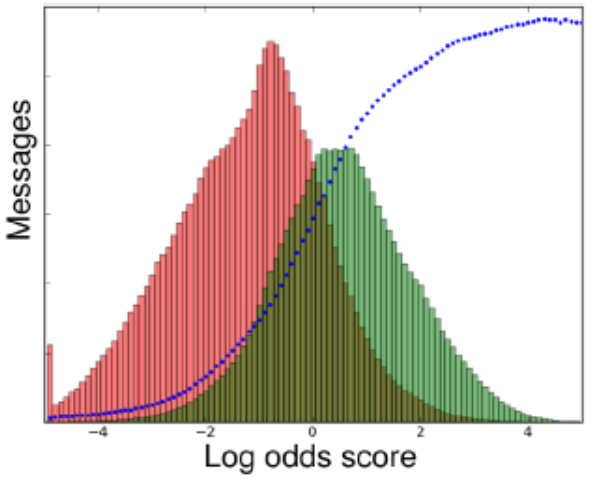


Figure 3: Score distribution for the global model. Dots show ratio of important to not important.

INPUT: aggressiveness parameter $C > 0$

INITIALIZE: $\mathbf{w}_1 = (0, \ldots, 0)$

For $t = 1, 2, \ldots$

- receive instance: $\mathbf{x}_t \in \mathbb{R}^n$
- predict: $\hat{y}_t = \text{sign}(\mathbf{w}_t \cdot \mathbf{x}_t)$
- receive correct label: $y_t \in \{-1, +1\}$
- suffer loss: $\ell_t = \max\{0\ ,\ 1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t)\}$
- update:

1. set:

$$\tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|^2} \qquad \text{(PA)}$$

$$\tau_t = \min\left\{C\ ,\ \frac{\ell_t}{\|\mathbf{x}_t\|^2}\right\} \qquad \text{(PA-I)}$$

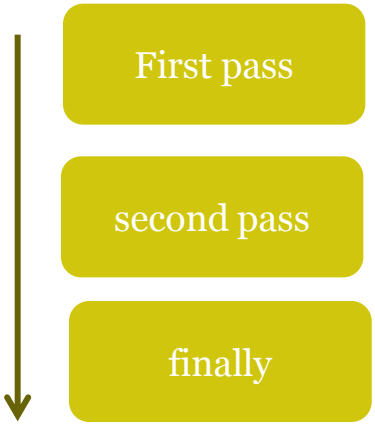
$$\tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|^2 + \frac{1}{2C}} \qquad \text{(PA-II)}$$

2. update: $\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t y_t \mathbf{x}_t$

Figure 1: Three variants of the Passive-Aggressive algorithm for binary classification.

3.2 Learning

- メールを評価する為にユーザーがGmailを最後に立ち上げたのはいつだったのか知りたい。
  - メッセージはuser:message-idに従って並んでいるので、全てのメッセージを読む必要が生じる。
- user modelのshardにごとに、2つの命令を実行する必要がある。



最後のaction timeとshard of usersの統計データを計算※データ量が小さいので早い

メッセージの特徴データ全体をスキャンする

更新されたuser model全てをひとまとめにbigtableにかきこむ

- Fraction of usersが利用可能な各コアに与えられ、このfractionはさらに断片化されて(shards of users）ひとまとめにRAMに入力される。
- 最終的にコア1台ごとの単位時間あたり（秒）35名を消化可能となった。

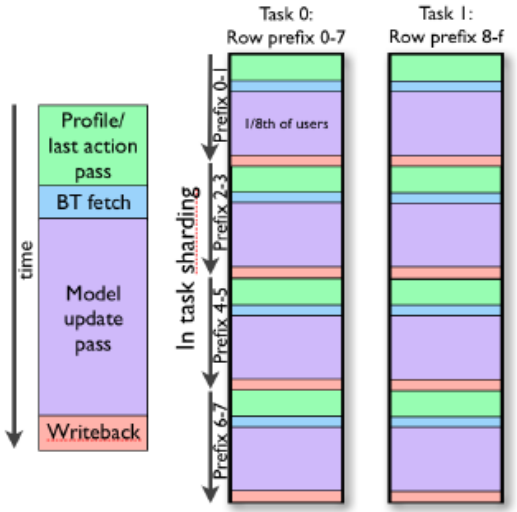


Figure 2: Sharding of user model learning.