

Problem Motivation

アノマリー検出

フラグのないデータセットが与えられ、さらに新しいtestデータが与えられたときに新しいtestデータが先のデータセットの集団に含まれるかどうかを判定する。

- 応用例
 - webサービスのユーザーでスパムや悪質なユーザーを検知する
 - 製造業などで製造した製品の品質が悪くないか検出する
- アルゴリズムの概要 新しいtestデータが先のデータセットに含まれている確率 $p(x)$ を計算し、それが設定した ϵ 以下であるかどうかで判断する

Gaussian Distribution

ガウス分布(正規分布)

正規分布上の点の確率は以下の式で定義される

$$p(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

Algorithm

考え方

すべてのデータセットはそれぞれ正規分布に従うと仮定する。すなわち

$$p(x) = p(x_1; \mu_1, \sigma_1^2)p(x_2; \mu_2, \sigma_2^2) \cdots p(x_n; \mu_n, \sigma_n^2) = \prod_{i=1}^n p(x_i; \mu_i, \sigma_i^2)$$

アルゴリズム

1. 以上ではないと思われるデータセットを集める(個数をmとする)
2. 平均と分散を計算する

$$\mu_j = \frac{1}{m} \sum_{i=1}^m x_j^{(i)}$$

$$\sigma_j^2 = \frac{1}{m} \sum_{i=1}^m (x_j^{(i)} - \mu_j)^2$$

3. 新しいデータに対してノーマルである確率 $p(x)$ を計算する

$$p(x) = \prod_{j=1}^n p(x_j; \mu_j, \sigma_j^2) = \prod_{j=1}^n \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(-\frac{(x_j - \mu_j)^2}{2\sigma_j^2}\right)$$

Developing and Evaluating an Anomaly Detection System

アノマリー検出の精度評価

アノマリーである確率はとても低く、どのデータセット(training,crossvalidation,test)も歪んだデータセットと言える。従って分類の精度はアルゴリズムの精度を表してはいない。

アルゴリズムの精度を示す指標としては

- true positive, false positive, false negative, true negative
- Precision / Recall
- F値

などを用いることができる。crossvalidation-setで ϵ の精度を見てもよい。

Anomaly Detection vs. Supervised Learning

- アノマリー検出を使った法がいい場合

- データセットが歪んでいる場合(アノマリー(陽性なデータがとても少なく、ノーマル(陰性)のデータセットがとても多い場合)
- アノマリーとなる要因がとても多い場合。(陽性のデータセットからアノマリーなものがどんなものか学習できない時)
- 新たなアノマリーな例が生まれうる場合
- 教師あり学習を用いた法がいい場合
 - 陽性のデータも陰性のデータもたくさんあるとき
 - 陽性のデータとはどんなものなのか、陽性のデータから抽出できる場合
- 具体例
- アノマリー検出
 - 詐欺検出(webユーザーの中から悪質なユーザーを検出する)
 - 製造業での不良品検出
 - 機械をモニタリングしているときの不具合検出
- 教師あり学習
 - スパムメール検出
 - 天気予報
 - ガン検出

Choosing What Features to Use

変数の変形

アノマリーに持ついる変数はガウス分布に従うほうがよりよいアルゴリズムになる。octaveのhist関数を使えばヒストグラムを見ることができる。プロットしたときにより正規関数に近づくように累乗してみたり、対数をとってみたりするとよい。

変数の選び方

アルゴリズムに使われる変数は

陰性のデータセットに対しては $p(x)$ が大きく、陽性のデータセットに対しては $p(x)$ が小さくなるようになっていてほしい。

しかし、実際には陰性、陽性ともに $p(x)$ が大きくなってしまい分類に寄与しない場合があるので陰性と陽性を分けやすくなるように変数を組み合わせたりしなくてはならない。

Multivariate Gaussian Distribution

多变量ガウス分布の利用

具体例で考えるデータセットが $x \in \mathbb{R}^n$ のとき、多变量ガウス分布のパラメータは $\mu \in \mathbb{R}^n$, $\Sigma \in \mathbb{R}^{n \times n}$ (共分散行列)である。

$n=2$ のときのパラメータによる分布の変化をみる μ は分布の中心を表す。

$$\text{Sigma} = \begin{matrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{matrix} \text{とする}$$

Σ が単位行列のとき、分布は $(0,0)$ を中心同心円上に広がる

Σ が単位行列に係数をかけた形のとき、その係数は分散をしめし、値が小さいときほど傾きがきつくなり、係数の値が大きいときグラフの傾きがゆるくなる。

s_{11} が x_1 方向の分散、 s_{22} が x_2 方向の分散となる。

s_{12}, s_{21} がともに正のとき x_1, x_2 が同符号のところに分布し、ともに負のときは異符号のところに分布する

Anomaly Detection using the Multivariate Gaussian Distribution

多变量分布の式はパラメータ $\mu \in \mathbb{R}^n$, $\Sigma \in \mathbb{R}^{n \times n}$ に対して

$$p(x; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$$

$$(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}$$

パラメータを求める式は以下の2つ

$$\mu = \frac{1}{m} \sum_{i=1}^m x^{(i)}$$

$$\Sigma = \frac{1}{m} \sum_{i=1}^m (x^{(i)} - \mu)(x^{(i)} - \mu)^T$$

多変量ガウス分布を用いるかどうか

- 多変量ガウス分布を使わない場合
 - よりよいアルゴリズムにするために変数同士を組み合わせたりする必要がある
 - 計算が多変量ガウス分布を使う場合に比べて早い(n (データ数)が大きくて大丈夫)
 - m ((データの変数の数)が小さくてもうまく動く)
- 多変量アルゴリズムを使う場合
 - 変数同士の関係性を素早く見つける自動で見つけることができる
 - 共分散行列の行列式を計算する必要があるので計算に時間がかかる
 - $m > n$ が成り立っていないと共分散行列が不可逆になり、うまくアルゴリズムが回らない($m \geq 10n$ ぐらいだとうまくいく)
- 多変量ガウス分布を使うときに注意すること
 - $m > n$ よりも十分大きいこと
 - 変数同士が線形に従属の関係になっていないかどうか。(同じ意味をもつ変数がデータセットの中に複数あったり、ある変数が他の変数の一次結合などで表されるとき共分散行列が不可逆になってしまう)

Problem Formulation

リコメンド(推薦)システムの構築を考える。レビューサイトやECサイトのようなwebサイトでは全てのユーザーが全てのプロダクトにレビュー、評価するわけではない。従って他の評価の数字から評価していないプロダクトの評価を予想する

Content Based Recommendation

- $r(i,j) = 1$ if user j has rated movie i (0 otherwise)
- $y^{(i,j)}$ = rating by user j on movie i (if defined)
- $\theta^{(j)}$ = parameter vector for user j
- $x^{(i)}$ = feature vector for movie i

For user j , movie i , predicted rating: $(\theta^{(j)})^T(x^{(i)})$

$m^{(j)}$ no.of products(ex.movies) rated by user j

To learn $\theta^{(j)}$

$$\min_{\theta^{(j)}} \frac{1}{2} \sum_{i:r(i,j)=1} ((\theta^{(j)})^T(x^{(i)}) - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{k=1}^n (\theta_k^{(j)})^2$$

To learn $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(n_u)}$

$$\min_{\theta^{(j)}} \frac{1}{2} \sum_{j=1}^{n_u} \sum_{i:r(i,j)=1} ((\theta^{(j)})^T(x^{(i)}) - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^n (\theta_k^{(j)})^2$$

Gradient descent update:

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \sum_{i:r(i,j)=1} ((\theta^{(j)})^T(x^{(i)}) - y^{(i,j)}) x_k^{(i)} (for k=0)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} ((\theta^{(j)})^T(x^{(i)}) - y^{(i,j)}) x_k^{(i)} + \lambda \theta_k^{(j)} \right) (for k \neq 0)$$

Collaborative Filtering

先の例では $x^{(i)}$ が与えられていたが、 $x^{(i)}$ の代わりに $\theta^{(j)}$ が与えられている状況

で考えていく。

- Optimizarion algorithm given $\theta^{(i)}, \dots, \theta^{(n_u)}$ to learn $x^{(i)}$:

$$\min_{x^{(i)}, \dots, x^{(n_m)}} \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} ((\theta^{(j)})^T (x^{(i)}) - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^n (x_k^{(i)})^2$$

以上2つの内容から $x \rightarrow \theta$ と $\theta \rightarrow x$ の学習アルゴリズムがわかる。従ってそれを交互に用いることによってよりよりリコメンドシステムができるのである。

Collaborative Filtering Algorithm

- 協調フィルタリングのアルゴリズム

1. $x^{(1)}, x^{(2)}, \dots, x^{(n_m)}, \theta^{(1)}, \theta^{(2)}, \dots, \theta^{(n_u)}$ を小さい乱数とおいて初期化する(それぞれは別の値だと示すために)

2. 以下のコスト関数を最急降下法を用いて最小化する

$$J(x^{(1)}, x^{(2)}, \dots, x^{(n_m)}, \theta^{(1)}, \theta^{(2)}, \dots, \theta^{(n_u)}) = \frac{1}{2} \sum_{(i,j):r(i,j)=1} ((\theta^{(j)})^T (x^{(i)}) - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^n (x_k^{(i)})^2 + \frac{\lambda}{2}$$

そして、 x, θ を最適化する for every $j = 1, \dots, n_u, i = 1, \dots, n_m$:

$$x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} ((\theta^{(j)})^T (x^{(i)}) - y^{(i,j)}) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} ((\theta^{(j)})^T (x^{(i)}) - y^{(i,j)}) x_k^{(i)} + \lambda \theta_k^{(j)} \right)$$

3. 以上のアルゴリズムを用いて計算した x, θ を用いて、 $\theta^T x$ を計算する

このアルゴリズムにおいては乱数を用いた初期化をしていて、バイアス項は必要ないので $x \in \mathbb{R}^n, \theta \in \mathbb{R}^n$ となる。

Vectorization:Low Rank Matrix Factorization

$$X = \begin{matrix} -(X^1)^T \\ -(X^2)^T \\ \vdots \\ -(X^{n_m})^T \end{matrix}$$

$$\Theta = \begin{matrix} -(\theta^1)^T \\ -(\theta^2)^T \\ \vdots \\ -(\theta^{n_u})^T \end{matrix}$$

とすると $X\Theta^T$ はすべてのユーザーのすべてのプロダクトに対する予想評価を表す行列になる。また、この $X\Theta^T$ は階数が低いため、この計算を low rank matrix factorization(低ランク行列分解)という。

- ある商品Aを買ったときに関連性のある商品として商品を5つ推薦したいとき

$|x_a - x_j|$ が最小となる j を5つ見つけてくれればよい

Implementational Normalization

- 映画のリコメンドシステムの話として考える

新しいユーザが入ってきたときに何を推薦するかを考える。今まで通りに協調フィルタリングを適用させてしまうと、新しいユーザーによる映画の評価はすべて0になってしまって推薦システムは意味をなさなくなる。

そこで新しいユーザーが入ってくる前のすべてのプロダクト(映画)ごとに評価の平均をとる。すべての評価を示す行列からプロダクトごとに求めた平均を引き、協調フィルタリングのアルゴリズムを走らせる。結果として新しいユーザーがこれまでの平均からの差分だけの評価をする。

Detail:Mean

リーナによるノロツント(吹笛)の評議会にてのユーザーがハッピーハウスの前で干場にい
うことになる。