IBM research

Tokyo Research Laboratory

A Kernel Path Algorithm for Support Vector Machine
(カーネルパラメータをイロイロ変えながらSVMをとく方法)

G. Wang, D.-Y. Yeung, and F.H. Lochovsky
香港科学技術大学

よむひと：鹿島久嗣（IBM／TRL）

この論文は、カーネル関数のパラメータが変わったときに、SVMの解がどのように変わるか（=最適解パス）を、効率よく追いかける方法を提案しています

- 選んだ理由：最近、なにかとよく聞くsolution path (regularization path)、一体どうやるのかが気になっていたので（津田さんもやってるし…）
  - cross-validationでのハイパーパラメータ選択が、効率的に行える方法？
  - 特にL_1-regularization下で効率がよいらしい → L_1モノを書くときに、とりあえずやっとけば、1セクション書ける？
  - カーネルのハイパーパラメータ変えたときにもできたらいいよね？（この論文）

そもそもの問題： 学習のハイパーパラメータを決めるのは、特に計算量の観点から大きな問題になります

- 一般に学習器の「ハイパーパラメータ」をどのように決めるかというのは難しい問題
- たとえば、パラメータ学習につかう訓練データのほかに、ハイパーパラメータチューニング用の調整用データを使ってチューニングする
- しかし、
  - ハイパーパラメータを固定する
  - 訓練データでパラメータを学習する ←ここが重い
  - 調整用データでの性能を測る

  をくりかえすのは、何度も学習しなければならないので大変
- 何度も学習することなく、いろいろなハイパーパラメータの値のときのパラメータを効率的に計算できるとうれしいハズ
- そこで、最適解パス（solution path）

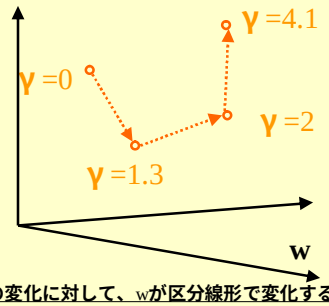
最適解パス（solution path）とは、正則化パラメータなどを変えたときの解（最適パラメータ）の変化を追いかけたものです

- 学習問題を定式化した、最適化問題は、通常
  - 目的関数(w) = 損失関数(w) ＋ γ×正則化項(w)

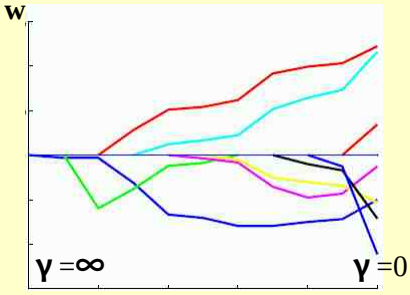
  の形をしている
    - w はモデルパラメータで求めるべきもの
    - γは正則化パラメータで人がテキトーに与えるもの
  - これを最小化することで、最適なパラメータw^*を求める
- γが変われば、最適解w^*も変わる
  - w^*はγの関数になっている
- このときγの変化にしたがって$w^*(\gamma)$が描く軌跡を解パス（solution path）という

最適解パスの一般的な特徴： 最適化パスは区分線形になることが多い（ただ、今回は成り立ちません）

- たとえば、L_1正則化で、最小2乗回帰をする場合など
  - あとで濱田さんが紹介するTsudaさんの論文
- γの変化に対して、wが区分線形で変化する
  - 一旦、最適解の変化する方向がわかってしまえば、次に曲がるまでは、最適解は線形で変化する
  - 従って、向きを求める→次に曲がる点を見つける の繰り返しをおこなう
    - γをちょっと動かしては、最適化を解きなおす必要が無い



γの変化に対して、wが区分線形で変化するイメージ



それぞれのパラメータの最適解パス

この論文でやったこと： カーネルパラメータをハイパーパラメータと思ったときの、SVMの最適解パスを求めました

- 一般的な問題として

「学習器」の「ハイパーパラメータを変えたとき」の「最適解パス」を求める

という問題がある

  - 学習器 ∈ { 線形回帰, カーネルリッジ回帰, ロジスティック回帰, SVM, ... }
  - ハイパーパラメータ ∈ { パラメータのL1ノルムの正則化パラメータ, パラメータのL2ノルムの正則化パラメータ, **カーネルパラメータ**, ... }
- 今回は、
  - 学習器=SVM
  - ハイパーパラメータ=カーネルパラメータ

  としたときの最適解パスを求めた
- これまでとちょっと違うところは、最適解パスが区分線形にならないところ

基本方針：
ハイパーパラメータの変化とともに、KKT条件を追いかけます

- やりたいこと：カーネルパラメータγの変化にしたがうSVM（↓）の解の変化を追いかける
  - SVMの最適化問題（dualのほう）

$$\begin{aligned} \max_{\boldsymbol{\beta},\beta_0} \quad & R_{dual}(\boldsymbol{\beta},\beta_0) = \lambda \textstyle\sum_{i=1}^{n} \beta_i - \\ & \tfrac{1}{2} \textstyle\sum_{i,j=1}^{n} \beta_i \beta_j y_i y_j K_\gamma(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.t.} \quad & \textstyle\sum_{i=1}^{n} y_i \beta_i = 0 \\ & 0 \le \beta_i \le 1, \end{aligned}$$

見慣れた形とちょっと違うけど気にしないでね

βが求めるべきパラメータ

γが今回のハイパーパラメータ（ｶｰﾈﾙﾊﾟﾗﾒｰﾀ）

  - 分類器の形（λのことは気にするな）

$$f(\mathbf{x}) = \frac{1}{\lambda}\left(\sum_{i=1}^{n} \beta_i y_i K_\gamma(\mathbf{x}, \mathbf{x}_i) + \beta_0\right)$$

- 具体的には、SVMの解が満たすべき条件 ＝ KKT条件を追いかける
  - KKT条件は線形の方程式になる
  - ハイパーパラメータが少しだけ変化したときの、KKTの線型方程式の解をアップデートする

補足： SVMのKKT条件のおさらい

- SVM （dualのほう）の最適化問題

$$\begin{aligned} \max_{\boldsymbol{\beta},\beta_0} \quad & R_{dual}(\boldsymbol{\beta},\beta_0) = \lambda \textstyle\sum_{i=1}^{n} \beta_i - \\ & \tfrac{1}{2} \textstyle\sum_{i,j=1}^{n} \beta_i \beta_j y_i y_j K_\gamma(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.t.} \quad & \textstyle\sum_{i=1}^{n} y_i \beta_i = 0 \\ & 0 \le \beta_i \le 1, \end{aligned}$$

βが求めるべきパラメータ

γが今回のハイパーパラメータ（ｶｰﾈﾙﾊﾟﾗﾒｰﾀ）

- 分類器の形

$$f(\mathbf{x}) = \frac{1}{\lambda}\left(\sum_{i=1}^{n} \beta_i y_i K_\gamma(\mathbf{x}, \mathbf{x}_i) + \beta_0\right)$$

- 最適解ではKKT条件が成立する （最適解において、各訓練データ i のパラメタ β_i は以下のどれかを満たす）

- If $y_i f(\mathbf{x}_i) > 1$, then $\xi_i = 0$ and $\beta_i = 0$;
- If $y_i f(\mathbf{x}_i) = 1$, then $\xi_i = 0$ and $\beta_i \in [0, 1]$;
- If $y_i f(\mathbf{x}_i) < 1$, then $\xi_i > 0$ and $\beta_i = 1$.

KKT条件によると、解は3つの集合にわかれます

- KKT条件によれば、訓練データは3つの集合（↓）にわけられる
- ３つの集合が変化しない限りは、次がなりたつ
  - m個の解（簡単のため最初のm個とする）は線型方程式を満たす
  - 残りの解は0か1かが決まっている

解はこの線型方程式を満たす

i=1,..,m がこれを満たすとする

$$\mathcal{E}=\left\{ i : y_i \left(\sum_{j=1}^{n} \beta_j y_j K_\gamma(\mathbf{x}_i,\mathbf{x}_j) + \beta_0 \right) = \lambda, 0 \le \beta_i \le 1 \right\}$$

i=m+1,...,n がこれを満たすとする

$$\mathcal{L}=\left\{ i : y_i \left(\sum_{j=1}^{n} \beta_j y_j K_\gamma(\mathbf{x}_i,\mathbf{x}_j) + \beta_0 \right) < \lambda, \beta_i = 1 \right\}$$

$$\mathcal{R}=\left\{ i : y_i \left(\sum_{j=1}^{n} \beta_j y_j K_\gamma(\mathbf{x}_i,\mathbf{x}_j) + \beta_0 \right) > \lambda, \beta_i = 0 \right\}$$

解はβ=0かβ=1か

新しい解は、連立方程式を解けば求まります

- 前述の3つの集合が変化しないとすると、解βは m+1変数、m式の線型方程式を解けば求まる
  - β_iが0か1に決まらない、m個のβ_iの満たすべき線形方程式
    - m+1変数(β_0も含めて)、m式

$$y_i \left(\sum_{j=1}^{n} \beta_j(\gamma) y_j K_\gamma(\mathbf{x}_i,\mathbf{x}_j) + \beta_0(\gamma) \right) = \lambda$$

  - もともとの制約
    - m+1変数(β_0も含めて)

変数はβ_0〜β_mののm+1個。あとは定数

$$\sum_{i=1}^{n} y_i \beta_i(\gamma) = 0$$

- 従って、
  - γをちょっと変え、カーネルを計算しなおす
  - 上の連立方程式を解く（(m+1)£(m+1)行列の逆行列）

  を繰り返すことで、SVMの2次計画問題を解きなおすことなく最適解を追いかけることができる

最適解パスは区分線形でないため、曲がり角の計算が予めできず、少しづつ探索していくアルゴリズムになります （なので、ちょっとイマイチ）

- γの変化にたいし、カーネルが線形で変化するとは限らないため、解も線形で変化するとは限らない → 曲がり角（３つの集合が変化する点）を計算できない

$$y_i \left(\sum_{j=1}^{n} \beta_j(\gamma) y_j K_\gamma(\mathbf{x}_i, \mathbf{x}_j) + \beta_0(\gamma) \right) = \lambda$$

- なので、ちょっとづつ探索アルゴリズム

1. 適当なハイパーパラメータγからスタート、一回SVMを解く
2. γをちょっと減らす
3. カーネルを計算
4. 連立方程式を解いて、新しい解を求める
5. 3つの集合が変化しないかどうかをチェック
   - 変化しないなら、2.に戻る
   - 変化するなら、γの減らし方をちょっと減らして3.からやり直す
   - どうしても変化してしまう場合には、3つの集合が変化するということだから、3つの集合をアップデートして2.からやり直す

感想:
問題が難しすぎたのかな、という印象

- 探索が必要なのは仕方ないとしても、かなりモロにといてるのに近い
- ハイパーパラメータを変えるたびに、カーネルを計算しなおす必要がある
- 実用的な限界は、パスが区分線形になるあたりまで？
- ただ、パスもの自体にはもう少し注目したい
  - もうちょっといろいろな問題で考えられそうな気がするし
  - ハイパーパラメータチューニングの決定版になる可能性を秘めているかもと期待