THE UNIVERSITY OF TOKYO

複合情報学特別講義第二

拡がる機械学習の可能性
― グラフとネットワークの機械学習を中心として

東京大学大学院
情報理工学系研究科
鹿島 久嗣

DEPARTMENT OF MATHEMATICAL INFORMATICS

回帰と分類という教師付き学習の２大タスクにおいて、グラフデータの分析手法を紹介します

- 回帰
  - グラフ上での回帰
- 分類問題
  - ネットワーク構造の予測
  - グラフ構造の予測

回帰

回帰問題の定義

回帰は、実数値を予測する教師つき学習

- 回帰は教師つき学習問題の一種
  - 機械学習の問題の中で極めて王道、適用範囲も広い
- 目的は、入力 $x \in \mathcal{X}$ に対し、実数値 $y \in \mathcal{R}$を返すような関数 $f: \mathcal{X} \to \mathcal{R}$を得ることである。
  - たとえば x はある家、yはその家の価格を表す
  - $\mathcal{X}$ はこの世に存在しうる限りの家の集合、$\mathcal{R}$ は実数
- しかし、何の手がかりもなしに f を得ることは難しい。
- ある家Aの価格が2,000万円、家Bの価格が4,000万円、…といったように正解がいくつか与えられていれば、これらをもとに、家とその価格との関係について何らかの法則を見つけることができるかもしれない
  - ベンチマークデータ： UCI Machine Learning Repository / housing

「条件付き分布の推定」という本来の目的は一旦忘れます

- なお、前回までの文脈に則していえば、xが与えられたときのyの条件付き確率分布 $P(y|x)$ を得ること
  - これがわかれば、fは与えられた入力x に対して確率が最大になる出力を返す関数として得られる

    $f(x) = \mathrm{argmax}_y \; P(y|x)$
  - とりあえずは $P(y|x)$ではなく$f(x)$を直接得るのが目的とする

回帰では、入出力ペアの集合（訓練データ集合）を一般化することで、出力未知のデータへの対応を目指します

- 訓練データ：N 個の入力と出力の組 $(\boldsymbol{x}^{(1)}, y^{(1)}), \ldots, (\boldsymbol{x}^{(N)}, y^{(N)})$
  - N 軒の家と、それぞれの価格が分かっているものとする
  - $\boldsymbol{x}^{(i)}$ はD次元の特徴ベクトル
    - 家の価格を予測するのに有効そうな特徴: 部屋数、駅までの距離、地域の犯罪発生率、…
  - $y^{(i)}$ は実数値（家の価格）
- 目的は、訓練データ集合をもとに関数 $f: \mathcal{X} \to \mathcal{R}$ を推定すること
  - 単に、訓練データの入出力を再現するだけならば、訓練データを丸覚えしてしまえばよい → これでは「学習」とはいえない
  - 本当に実現したいのは、訓練データに入っていない $\boldsymbol{x}$ に対しても正しく出力を予測できるような f を得ること（**汎化**）

離散的な特徴は、便宜的に実数値化して使用します

- 離散的な特徴は「便宜的に」実数値であるとして扱う
- 性別のような2値的な特徴は、男ならば1、女ならば-1など
  - これを{0,1}で符号化すると、結果も異なる
- 都道府県などのように複数の可能性があるならば、東京都である(1)かない(-1)かといった特徴を複数個用意するなどで対応する

※ 旧来の（統計的でない）機械学習においては、むしろ離散がデフォルト→後に連続化 という道筋であった
- 統計的機械学習では、むしろ連続的なモデルがデフォルト

多くの機械学習問題における重要な仮定：データは独立であるとします

- ある家Aのデータが、2000年当時のものと、2010年当時のものの2つある場合を考えてみる
  - 2010年での価格は、2000年での価格に依存する
- 大抵の機械学習手法は、データはお互いに独立であることを仮定している
- つまり、特徴ベクトルとラベルの組$(\boldsymbol{\phi}, y)$の上の確率分布 $P(\boldsymbol{\phi}, y)$ からそれぞれ互いに独立に生成(サンプリング)されているものとしている
- データの独立性を仮定しているので、同じ家から2回以上データを取得するのは厳密には若干問題がある
  - 「サンプリングの偏り（バイアス）」問題
- 当面のところ、独立性を仮定する

回帰問題の応用

- 回帰問題の応用例：
  - 価格予測：ある商品 $\boldsymbol{x}$ がいくら（y）で売れるか？
  - 需要予測：ある商品 $\boldsymbol{x}$ がどのくらい（y）需要があるか？
  - 売上予測：ある商品 $\boldsymbol{x}$ がどのくらい（y）売れるか？
  - 活性予測：ある化合物 $\boldsymbol{x}$ がどのくらい（y）活性をもつか？
- ほか、若干抽象度は異なるが（後で述べる）：
  - 時系列予測：ある過去の履歴 $\boldsymbol{x}$ が与えられたときの、次の時点での値 y はいくつか？
  - 分類問題：出力 y が実数値ではなく、離散値を取る場合
    - 分類問題に特化した手法は後の回で紹介する。

線形回帰問題の定式化

回帰問題を、線形回帰問題として定式化してみます

- 回帰問題を定式化するにあたり、定義しなければならないのは
  - モデル：どのような形式で $\boldsymbol{x}$ からyを予測するか？
  - 目的関数：訓練データをどのように用いるか？
- ここでは、最も標準的な定義を用いることにする
  - モデル：線形モデル（線形回帰）
  - 目的関数：2乗損失（2乗誤差）

モデルの定義：線形モデル を考えます

- モデルとしては単純な線形モデルを考えることにする

$$f(x; \mathbf{w}, b) \equiv \mathbf{w}^\top \mathbf{x} = \sum_{d=1}^{D} w_d x_d$$

  - 入力xが与えられたときに、出力の予測値として$f(\boldsymbol{x}; \mathbf{w})$を返す
  - モデルのもつパラメータ（モデルを一意に決定するもの）は
    - ベクトル$\mathbf{w}$：特徴空間と同じD次元のベクトル $\mathbf{w} \equiv (w_1, w_2, \ldots, w_D)$ で、d次元目の値w_dは、d番目の特徴x_dの出力への寄与を表す

目的関数の定義：2乗誤差を用います

- 与えられた訓練データ$\{ (\boldsymbol{x}^{(i)}, y^{(i)}) \}_{i=1}^N$から、パラメータ$\mathbf{w}$を決定するために、回帰を最適化問題として定式化する。
- 我々の持つ情報は訓練データであるから、
  - 与えられた訓練データ$\{ (\boldsymbol{x}^{(i)}, y^{(i)}) \}_{i=1}^N$のそれぞれに対して、
  - モデルの出力$f(\boldsymbol{x}^{(i)}; \mathbf{w})$を、正しい出力$y^{(i)}$になるべく近づける
- 損失関数$l(f(\boldsymbol{x}; \mathbf{w}), y)$：モデルの出力と正しい出力の「遠さ」を具体的な例としては、2乗誤差（L_2損失）がよく用いられる

$$\ell(f(x; \mathbf{w}), y) \equiv (f(x; \mathbf{w}) - y)^2$$

目的関数の定義：2乗誤差を用います

- 目的関数として、損失関数の和を考える

$$L(\mathbf{w}) \equiv \sum_{i=1}^{N} \ell(f(x^{(i)}; \mathbf{w}), y^{(i)})$$

- 損失関数を2乗損失とするならば、

$$L(\mathbf{w}) \equiv \sum_{i=1}^{N} \left(f(x^{(i)}; \mathbf{w}) - y^{(i)} \right)^2$$

- パラメータの推定値$\mathbf{w}$^は、この損失関数の和$L(\mathbf{w})$を最小化するように決定される

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} \; L(\mathbf{w})$$

線形回帰の初等的解法

2乗誤差を用いた線形回帰問題を解いてみます

- 2乗誤差の和は、線形モデルを仮定すると、

$$L(\mathbf{w}) = \sum_{i=1}^{N} \left(\mathbf{w}^\top \mathbf{x}^{(i)} - y^{(i)} \right)^2$$

- これは以下のように書き換えることができる。

$$L(\mathbf{w}) = \| \Phi \mathbf{w} - \mathbf{y} \|_2^2 = (\Phi \mathbf{w} - \mathbf{y})^\top (\Phi \mathbf{w} - \mathbf{y})$$

  - 訓練データの特徴ベクトル集合$\{\boldsymbol{x}^{(i)}\}_{i=1}^N$をまとめて行列とし

$$\Phi \equiv \left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)} \right)^\top$$

    - この行列は**デザイン行列**と呼ばれる
  - 対応する出力集合$\{ y^{(i)} \}_{i=1}^N$をまとめて、ベクトルとして

$$\mathbf{y} \equiv \left(y^{(1)}, y^{(2)}, \ldots, y^{(N)} \right)^\top$$

  - $\| \cdot \|_2^2$は2-ノルム $\| \mathbf{a} \|_2^2 \equiv \mathbf{a}^\top \mathbf{a}$

線形回帰問題の解は閉じた形で得られます

- 目的関数 $L(\mathbf{w}) = \| \Phi \mathbf{w} - \mathbf{y} \|_2^2$ を最小化する$\mathbf{w}$を求めるために

  $\mathbf{w}$で偏微分する $\dfrac{\partial L(\mathbf{w})}{\partial \mathbf{w}} = 2\Phi^\top (\Phi \mathbf{w} - \mathbf{y})$

- これを$\mathbf{0}$とおくことで $\Phi^\top \Phi \mathbf{w} = \Phi^\top \mathbf{y}$
- これを解くと、以下のように閉じた形で解が得られる

$$\mathbf{w} = (\Phi^\top \Phi)^{-1} \Phi^\top \mathbf{y}$$

  - ここで、$\boldsymbol{\Phi}^\top \boldsymbol{\Phi}$は$D \times D$行列
  - 実際に解くには連立方程式を解くなどする
    - MATLABでは $\mathbf{w} = (\Phi' * \Phi) \backslash (\Phi' * \mathbf{y})$

逆行列の解を安定させるための「正則化」

- 解 $\mathbf{w} = (\Phi^\top \Phi)^{-1} \Phi^\top \mathbf{y}$ が存在する条件は$\boldsymbol{\Phi}^\top\boldsymbol{\Phi}$が正則であること、すなわち、フルランクであること
  - 通常、訓練データ数Nは、特徴空間次元数Dよりも大きいため、$\boldsymbol{\Phi}^\top\boldsymbol{\Phi}$はフルランクとなることが多い
- そうでない場合には$\boldsymbol{\Phi}^\top\boldsymbol{\Phi}$の対角成分に小さな正の値を加え、$\boldsymbol{\Phi}^\top\boldsymbol{\Phi}+\lambda \boldsymbol{I}$（ただし、$\lambda$>0は小さな正の値）とすることで、$\boldsymbol{\Phi}^\top\boldsymbol{\Phi}$を正則にする

$$\left(\Phi^\top \Phi + \lambda \mathbf{I}\right) \mathbf{w} = \Phi^\top \mathbf{y} \quad \Rightarrow \quad \mathbf{w} = \left(\Phi^\top \Phi + \lambda \mathbf{I}\right)^{-1} \Phi^\top \mathbf{y}$$

  - あとで述べる「正則化」と密接な関係

リッジ回帰：過学習を防ぐ

訓練データ数よりも特徴空間の次元数が大きい場合には過学習と呼ばれる性能悪化の現象が起こります

- 多くのスジの良いケースでは、訓練データの数Nは、特徴空間の次元Dよりも十分に大きいため、前述の方法で、良い予測精度（汎化性能）を持つモデルが得られる
- しかし、特徴空間の次元が比較的高い場合には、いわゆる**過学習**と呼ばれる問題が起こる。
  - 過学習：訓練データに過剰に適合してしまうことで、むしろ汎化能力を失ってしまう現象
    - 我々の本来の目的は、訓練データの入出力関係を忠実に再現することではなく出力が未知の入力に対して、その出力を正しく出力すること（汎化）
- 特徴空間の次元Dと比較して、訓練データ数Nが大きくない場合には、連立方程式において、$\boldsymbol{\Phi}^\top\boldsymbol{\Phi}$が正則でない、すなわち、フルランクでないことが多く、実質的に、変数の数よりも等式制約の数が少なくなってしまうため、解がいくらでも存在することになる。

モデル選択の一つの基準：なるべくシンプル（≒滑らかな）なモデルを採用せよ

- たくさんある解の中でよいモデルとは何だろうか？
- 「オッカムの剃刀」の教え：なるべくシンプルなモデルを採用せよ
- 「シンプルなモデルを採用することが本当に理論的に良いのか？」という問いに対する答えは後回しにして、とりあえず、その教えを信じることにする
  - なお、単純に経験則としてみても、シンプルなモデルを採用することは大抵の場合良い方向に働く
- 「シンプルなモデル」という気持ちの表現は様々考えられるが、ここでは「滑らかなモデル」とする

モデルのシンプルさはパラメータの2-ノルムで表現する

- モデルの「滑らかでなさ」は、$\mathbf{w}$の2-ノルム$\|\mathbf{w}\|_2^2$で表現する
- 我々が最小化すべき目的関数に$\|\mathbf{w}\|_2^2$を加える

$$L(\mathbf{w}) = \| \Phi\mathbf{w} - \mathbf{y} \|_2^2 + \lambda \| \mathbf{w} \|_2^2$$

  - λは0以上の定数（$\|\mathbf{w}\|_2^2$に対するペナルティの強さを調節）
- λは使用者が決定する必要があり**ハイパーパラメータ**と呼ぶ
  - のちにハイパーパラメータを自動的に決定する方法について学ぶ

2-ノルムをペナルティに用いた線形回帰の解は前述の「$\boldsymbol{\Phi}^\top\boldsymbol{\Phi}$ の正則化」と一致します

- 改めて新しい目的関数を$\mathbf{w}$について最小化してみる

$$L(\mathbf{w}) = \| \Phi\mathbf{w} - \mathbf{y} \|_2^2 + \lambda \| \mathbf{w} \|_2^2$$

- $\mathbf{w}$で偏微分すると $\dfrac{\partial L(\mathbf{w})}{\partial \mathbf{w}} = 2\Phi^\top(\Phi\mathbf{w} - \mathbf{y}) + 2\lambda\mathbf{w}$
  - 2項目が$\|\mathbf{w}\|_2^2$に由来する項
- これを$\mathbf{0}$とおくと $\left(\Phi^\top \Phi + \lambda \mathbf{I}\right) \mathbf{w} = \Phi^\top \mathbf{y}$ （前述の式に一致する）
- 2-ノルム正則化を用いた線形回帰のことを**リッジ回帰**と呼ぶ

正則化：パラメータノルムにペナルティを入れる

- また、パラメータのノルムにペナルティを課すことを **正則化**と呼ぶ
- パラメータの2-ノルム$\|\mathbf{w}\|_2^2$にペナルティを課すことをL2正則化／ティホノフ正則化／リッジ正則化などと呼ぶ
- 2-ノルムの他、よく使われるものとしては **1-ノルム**がある

$$\|\mathbf{w}\|_1 \equiv |w_1| + |w_2| + \cdots + |w_D|$$

  - 1-ノルムを用いた正則化をL_1正則化と呼ぶ
  - L_1正則化を施した線形回帰をラッソ（Lasso）と呼ぶ
  - これはL_2正則化と並び重要であるので、後ほど改めて述べる
- シンプルなモデル
  - 変数の少ないモデル：0-ノルム（凸でない）
  - 重みの小さいモデル：1-ノルム、2-ノルム（凸）

ここまでのまとめ

- 回帰問題は、実数値を予測する問題
- その代表的な定式化は2乗誤差を目的関数として使った線形回帰
- 線形回帰は逆行列（連立方程式）で解ける
- 次元数がデータ数と比較して大きい場合には過学習を防ぐために「シンプルなモデル」を選ぶ正則化を用いる
- 2ノルム正則化（L_2正則化／ティホノフ正則化）を使った線形回帰（リッジ回帰）も逆行列によって解ける

回帰の確率モデル的解釈

回帰の確率モデル的解釈：「損失の最小化」は最尤推定として解釈できます

- 「教師つき学習とは条件付き確率$P(y|x)$を推定する問題である」とひとまず定義し、 **最尤推定**がその推定手段であると述べたが、ここまでの議論にはそのような確率モデル的な観点は現れていない
- ここでは、実はここまでに述べたことは、条件付き確率 $P(y|x)$ の最尤推定と等価である
- 最尤推定は、訓練データ集合$\{(x^{(i)},y^{(i)})\}_{i=1}^N$に対して、対数尤度の和

$$L(\mathbf{w}) \equiv \sum_{i=1}^{N} \log P(y^{(i)}|x^{(i)};\mathbf{w})$$

を最大化するようなパラメータを、推定パラメータ$\mathbf{w}^*$とする考え方

$$\mathbf{w}^* \equiv \underset{\mathbf{w}}{\operatorname{argmax}}\, L(\mathbf{w})$$

線形回帰モデルの確率モデル的解釈：出力にガウス的ノイズが載るものとします

- 回帰問題に対応する条件付き確率$P(y|x)$を定義する
- 出力yが、平均が$f(x;\mathbf{w})=\mathbf{w}^\top \boldsymbol{x}$で分散が$\sigma^2$であるような正規分布に従って発生するものと仮定する。

$$P(y|x;\mathbf{w}) \equiv \mathcal{N}(f(x;\mathbf{w}),\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y-f(x;\mathbf{w}))^2}{2\sigma^2}\right)$$

  - なお、$\mathcal{N}(\mu,\sigma^2)$ は平均μ、分散σ^2をもつ一次元の正規分布

$$y \sim \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right)$$

線形回帰に対する対数尤度の最大化は2乗損失の最小化と一致します

- 回帰問題の場合に対数尤度の和を計算すると

$$\begin{aligned} L(\mathbf{w}) &= \sum_{i=1}^{N} \log\left(\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y-f(x;\mathbf{w}))^2}{2\sigma^2}\right)\right) \\ &= -N\log(\sqrt{2\pi}\sigma) - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y-f(x;\mathbf{w}))^2 \end{aligned}$$

- 1項目が$\mathbf{w}$に依存しないことに注意すると、最尤推定の解は

$$\mathbf{w}^* \equiv \underset{\mathbf{w}}{\operatorname{argmin}}\ \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y-f(x;\mathbf{w}))^2$$

となり、まさに2乗損失を損失関数とした線形回帰に一致する

リッジ回帰は、ベイズ統計的な解釈ができます

- 正則化によって、目的関数の2乗損失にパラメータのノルムを加えることで、過学習を緩和できる
- 目的関数に2乗損失を用いることが最尤推定に対応しているならばこれにパラメータのノルムを加えたものは何に対応しているだろうか？
- 正則化の枠組みは、ベイズ統計の立場から解釈できる

ベイズ統計では「パラメータの上での確率分布」を考えます

- これまで、パラメータ$\mathbf{w}$には「真の値」があるとしており、これをデータから計り知るための方法が最尤推定などであった
- ベイズ統計では、パラメータは一意に決まっているようなものではなく、何らかの確率分布に従って発生するもの、もしくは、パラメータの分布自体がパラメータの性質や意味を示していると考える
- ベイズ統計で重要な役割を果たすのが、**事後分布**と呼ばれる$P(\mathbf{w}|\mathcal{D})$
  - 訓練データ集合$\mathcal{D}$が与えられた時のパラメータの上での確率分布
  - 事後分布$P(\mathbf{w}|\mathcal{D})$は訓練データ集合$\mathcal{D}$を「見た後」に、どういった$\mathbf{w}$が確からしいかを表す

ベイズ統計における「学習」は、訓練データを見る前の事前分布から、観た後の事後分布への変化に対応します

- 事前分布と事後分布
  - 事前分布 $P(\mathbf{w})$ ：訓練データ集合$\mathcal{D}$を**見る前**のパラメータ上の分布
  - 事後分布 $P(\mathbf{w}|\mathcal{D})$：訓練データ集合$\mathcal{D}$を**見た後**のパラメータ上の分布
- 事前分布は、そもそもどのあたりのパラメータがそれらしいかを表す事前知識
- 訓練データ集合を与えられることによって、パラメータの上での確率分布が、事前分布$P(\mathbf{w})$から、事後分布$P(\mathbf{w}|\mathcal{D})$に変化する
  - これが、ベイズ統計における「学習」である
  - このあたりが、ベイズ統計に初めて触れる際、違和感を感じる部分

実は、正則化は事後確率最大化に対応します

- ベイズ統計ではパラメータは点ではなく分布で与えられるが、正則化の枠組みでは、確かに何か一つにパラメータが決まった
- 実は、正則化は、事後分布を最大化するようなパラメータが最良であるとする**事後確率最大化**(MAP; Maximum A Posteriori)という考え方に従ってパラメータを決定していることと等価である

事後確率最大化の目的関数は、最尤推定の目的関数＋事前分布による項 で補正したものと解釈できます

- **ベイズの公式**によって$P(\mathbf{w}|\mathcal{D})$を書き換えてみる

$$P(\mathbf{w}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathbf{w})P(\mathbf{w})}{P(\mathcal{D})}$$

- 形式的に、対数を取ると

$$\log P(\mathbf{w}|\mathcal{D}) = \log P(\mathcal{D}|\mathbf{w}) + \log P(\mathbf{w}) - \log P(\mathcal{D})$$

- 事後確率最大化 $\mathbf{w}^* \equiv \underset{\mathbf{w}}{\operatorname{argmax}} \log(\mathbf{w}|\mathcal{D})$ は、

$$\mathbf{w}^* \equiv \underset{\mathbf{w}}{\operatorname{argmax}} \log P(\mathcal{D}|\mathbf{w}) + \log P(\mathbf{w})$$

  - 1項目は対数尤度
  - 2項目は事前分布の対数を取ったもの
  - 最尤推定の目的関数に、事前分布で補正をかけている
- 最尤推定で求まるパラメータを、事前分布最大となるパラメータに「少し引き戻す」というイメージ

リッジ回帰の事後確率最大化としての解釈

- 事前分布$P(\mathbf{w})$ を各次元w_d が平均$\mathbf{0}$、分散η^2の正規分布とする

$$P(w_d) = \frac{1}{\sqrt{2\pi}\eta} \exp\left(-\frac{w_d^2}{2\eta^2}\right)$$

  - パラメータ$\mathbf{w}$はなるべく$\mathbf{0}$に近いものがよいとする正則化の気持ち
- 事前分布の対数を取ると、$\log P(w_d) = -\log\sqrt{2\pi}\eta - \frac{1}{2\eta^2}w_d^2$ より

$$\log P(\mathbf{w}) = -D\log\sqrt{2\pi}\eta - \frac{1}{2\eta^2}\|\mathbf{w}\|_2^2$$

  - 2項目に正則化項と同じ、パラメータの2-ノルムが現れる
  - 1項目は$\mathbf{w}$を含まないため無視できる
- 事後確率最大化がリッジ回帰に一致する（$\lambda \equiv \sigma^2/\eta^2$）

$$\mathbf{w}^* \equiv \underset{\mathbf{w}}{\operatorname{argmax}} \log P(\mathcal{D}|\mathbf{w}) + \log P(\mathbf{w})$$

$$= \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y - f(x;\mathbf{w}))^2 + \frac{1}{2\eta^2}\|\mathbf{w}\|_2^2$$

回帰の重要な応用

線形回帰の自明でない使い方を2つ紹介します

- 時系列予測
- 分類

時系列予測

- 線形回帰の1つの使い方として、時系列予測に用いることができる
- 時系列とは、時刻 $t=1,2,\ldots$ に関連づけられた、実数値の列 $x_1, x_2, \ldots$ ($x_t \in \mathcal{R}$)
- 時系列予測の目的は、ある時刻tにおける時系列の値 $x_t \in \mathcal{R}$ をそれ以前の値 $x_1, x_2, \ldots, x_{t-1}$ から予測すること
- 時系列予測のための代表的モデルの1つが、**自己回帰モデル**もしくは**AR(Auto Regressive)モデル**と呼ばれるモデル
- D次の自己回帰モデル：

$$x_t = w_1 x_{t-1} + w_2 x_{t-2} + \cdots + w_D x_{t-D}$$

  - ある時刻tにおける値は、過去D時点分の値から決まる
  - モデルのパラメータは $(w_1, w_2, \ldots, w_D)$

自己回帰モデルの学習は、線形回帰としてみることができます

- D次の自己回帰モデルは、

$$x_t = w_1 x_{t-1} + w_2 x_{t-2} + \cdots + w_D x_{t-D}$$

- これは、特徴ベクトルを $(x_{t-1}, x_{t-2}, \ldots, x_{t-D})$、パラメータを$(w_1, w_2, \ldots, w_D)$と考えれば、まさに線形回帰のモデル
- 時刻1から時刻Tまでの時系列の値 $x_1, x_2, \ldots, x_T$ が与えられたときに時刻T+1の値を予測したいものとする
- 時刻Tまでの時系列 $x_1, x_2, \ldots, x_T$ から長さD+1の窓をずらしながらT-D 個の訓練データを作ることができる。
- これらから、線形回帰の方法を用いてパラメータ$\mathbf{w}$を推定し、それをもとに x_T を予測する。
- 時系列予測については、より詳細なモデルや特化した解法がある

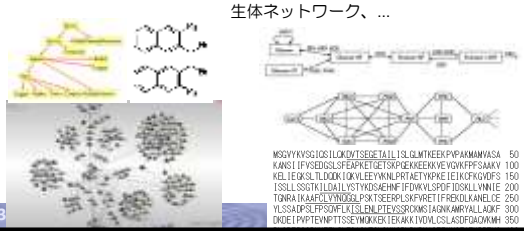
回帰で分類問題を解く

- 回帰を用いて分類問題（例えば、2値分類 $y \in \{+1,-1\}$ の場合）を解く
- 便宜的に各訓練データの出力を $y^{(i)} \in \{+1,-1\}$として回帰を適用する
  - 予測時には、$f(x; \mathbf{w})$ が0以上の値であるなら出力「+1」、そうでないならば「-1」として予測する
- 出力は+1か-1のどちらかなので、回帰の仮定（出力にガウス的ノイズが入る）は成立せず、このような適用は厳密には少しおかしい
- 分類問題をより適切にモデル化する方法は複雑になるため、分類問題を手軽に扱えるという意味で、このやり方は妥協に値する。
- なお、実際は訓練データの出力を $y^{(i)} \in \{+1,-1\}$とするのではなく、$y^{(i)} \in \{+1/|N_+|, -1/|N_-|\}$としたほうがよい
  - 出力が+1の訓練データの数を$|N_+|$、出力が-1の訓練データの数を$|N_-|$
  - フィッシャー判別に対応

ネットワーク上での回帰

世の中には様々な「構造をもったデータ」があります

- 構造を持ったデータとは、データの構成要素とそれらの間の関係によって記述されるデータ：
  - 配列： DNA、タンパク質、自然言語、イベント列、時系列
  - 木構造： HTML/XML、RNA構造、構文解析木、系統樹、ディレクトリ
  - グラフ／ネットワーク構造： 化合物、画像、Web、社会ネットワーク、生体ネットワーク、…

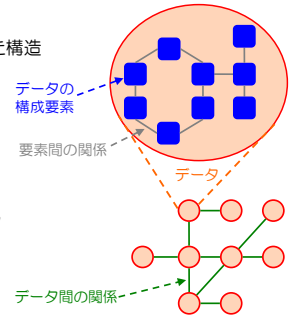


「構造」には内部構造（グラフ）と外部構造（ネットワーク）の２種類があります

- 内部構造（グラフ）：

  データ内の要素の関連を表した構造
  - 半構造データ（HTML／XML）
  - DNA配列
  - 化合物
- 外部構造（ネットワーク）：

  データ間の関連を表した構造
  - Web／文献の参照ネットワーク
  - 社会ネットワーク
  - 遺伝子／タンパク質／薬剤の相互作用ネットワーク



世の中には様々なネットワークがあります

ネットワーク	ノード	リンク
WWW	Webページ	ハイパーリンク
社会ネットワーク	人 コミュニティ	友人関係 所属
生体ネットワーク	遺伝子 タンパク質	制御 相互作用
企業ネットワーク	企業	取引 提携／出資
マーケティング	顧客 商品	購買関係 商品閲覧／評価
創薬	薬剤 標的タンパク質	作用

外部構造（ネットワーク構造）を扱う２つの予測タスク：ノード分類とリンク予測

- 外部構造にからむ２つの予測タスク
  - ノード分類：
    - ネットワーク上のいくつかのノードのラベルが与えられたとき、残りのノードのラベルを予測する
  - リンク予測（ネットワークの構造予測）：
    - ネットワーク上のいくつかのノード間のリンクが与えられたとき、残りのノード間のリンクを予測する

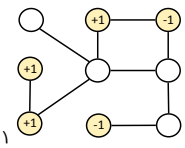
ノード分類

ノード分類では、グラフ上のいくつかのノードのクラスラベルをもとに、残りのノードのクラスラベルを予測します

- やりたいこと：グラフ構造および
  - ラベルつきのノード ○
  - ラベルなしのノード ○

  が与えられたとき
  - ラベルなしのノード ○

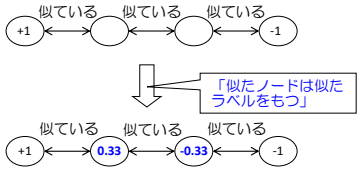
  のラベル（「+1」か「-1」）を当てたい



- 応用例：
  - タンパク質の相互作用ネットワーク上での機能予測
  - WWW上のスパムサイト予測
  - 企業ネットワーク上での格付け

ラベル伝播法は「隣同士は似ている」という制約を用いてラベル無しノードのラベル予測を行います

- 予測の基本原則：お互いに似たデータは、同じラベルをもつ可能性が高い
- ネットワーク上で隣り合うノードを、似ているとみなす
  - 似た者同士が繋がったネットワーク



ラベル伝播法では「隣同士のクラスラベルは似ている」の心を最適化問題として定式化します

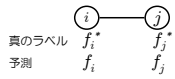
- 予測が満たすべき２つの条件を目的関数で書いてみる
  - ノード i のラベル予測を sign f_i とする
    - $\mathbf{f}$: f_i を並べたベクトル
  - ノード i の正解ラベルを f_i^* とする
    - $f_i^* \in$ {+1,-1,0(ラベル不明)} とする
    - $\mathbf{f}^*$: f_i^* を並べたベクトル
  - 条件 1）隣同士のクラスラベルは似ている

$$F_1(\mathbf{f}) \equiv \sum_{i,j}(f_i - f_j)^2 = \mathbf{f}^T \mathrm{L} \mathbf{f}$$

Lはグラフラプラシアン

  - 条件 2）ラベルつきのノードに対する予測が当たる

$$F_2(\mathbf{f}) \equiv \sum_{i}(f_i - f_i^*)^2 = (\mathbf{f} - \mathbf{f}^*)^T(\mathbf{f} - \mathbf{f}^*)$$



- F_1とF_2の両方を最小化することで予測を得る

参考： グラフラプラシアンLは、グラフの構造を行列で表現したものです

- ノードの次数をもつ対角行列から、隣接行列を引いたもの
  - 対角成分は、ノードの次数を表す
  - 対角以外の成分は、グラフのリンク情報を表す
    - リンクがあるときに-1
  - 対称行列

$$\mathrm{L} = \begin{pmatrix} d_1 & & & 0 \\ & \ddots & & \\ & & d_i & -1 \\ 0 & & & \ddots \\ & & & & d_N \end{pmatrix}$$

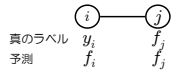
ノードiとノードjの間にリンクがあると(i,j)番目の要素が-1

i番目の対角成分はノードjの次数

ラベル伝播の最適化問題は閉じた形で解が得られます

- F_1とF_2を定数σで線形結合し、目的関数を作る

$$F(\mathbf{f}) \equiv \frac{\sigma}{2} F_1(\mathbf{f}) + F_2(\mathbf{f})$$
$$F_1(\mathbf{f}) \equiv \mathbf{f}^T \mathrm{L} \mathbf{f}$$
$$F_2(\mathbf{f}) \equiv (\mathbf{f} - \mathbf{f}^*)^T(\mathbf{f} - \mathbf{f}^*)$$



- これを解くと、閉じた形で解が得られる

$$\mathbf{f} = (\sigma \mathrm{L} + I)^{-1} \mathbf{f}^*$$

  - 連立方程式を解けばよい

L_1正則化

L_1正則化は、疎な解を与えるため、L_2正則化と並び重要な正則化のひとつです

- 1-ノルムを用いた正則化はL_1正則化と呼ばれる
  - L_2正則化と並び、よく利用される
- L_1正則化を用いた時の線形回帰の目的関数：

$$L(\mathbf{w}) = \| \Phi \mathbf{w} - \mathbf{y} \|_2^2 + \lambda \| \mathbf{w} \|_1$$

- L_1正則化は「疎な解」を与える
  - 目的関数を最小化する$\mathbf{w}^*$において、多くの次元が丁度0になる
  - 特徴ベクトル$\boldsymbol{\phi}$の次元が非常に高く、その一方で、実際に予測に有用な特徴が少ない場合に特に有効
  - 学習済みのモデルを用いて予測を行う際、予測の計算量は$\mathbf{w}$の0でない次元数に依存する

L_1正則化を用いた場合には（L_1正則化の場合と違い）最適化問題の解が閉じた形では求まりません

- L_1正則化を用いた回帰における最適化問題は、リッジ回帰のときと異なり、解が閉じた形で求まらないため、その解法は複雑になる
  - L_2正則化を用いた回帰（リッジ回帰）の解

$$\mathbf{w}^* = \left(\mathbf{\Phi}^\top \mathbf{\Phi} + \lambda \mathbf{I}\right)^{-1} \mathbf{\Phi}^\top \mathbf{y}$$

  - 方法1: 次元ごとの逐次解法
  - 方法2: L_2正則化への帰着

  など、さまざまな方法があるが、リッジ回帰ほど簡単ではない

方法1: 次元ごとの逐次解法 1つのパラメータに注目した最適化を繰り返します

- L_1正則化項 $\| \mathbf{w} \|_1$は：
  - 原点で微分できない
  - 原点が最適解となる場合が多い

  という問題があり、扱いづらい
- パラメータ $\mathbf{w}=(w_1, w_2, \ldots, w_D)$の、ある特定の次元 w_dに注目すると目的関数
  $L(\mathbf{w}) = \| \mathbf{\Phi w} - \mathbf{y} \|_2^2 + \lambda \| \mathbf{w} \|_1$ の最小化は簡単に行える
- そこで、
  1. 次元dを適当に選ぶ
  2. パラメータw_dについての最適化を行う

  を、収束するまで繰り返すアルゴリズムを考える

ある次元のパラメータ（w_d）に注目した目的関数を考えます

- i番目のデータに対するモデルの予測は、パラメータの d 次元目 w_d を特別扱いすれば：

$$\mathbf{w}^\top \phi(x^{(i)}) = w_d \phi_d(x^{(i)}) + \sum_{j \neq d} w_j \phi_j(x^{(i)})$$

- 目的関数を w_d についての関数であると思えば：

$$L(w_d) = \sum_{i=1}^{N} \left(w_d \phi_d(x^{(i)}) - \left(y^{(i)} - \sum_{j \neq d} w_j \phi_j(x^{(i)}) \right) \right)^2 + \lambda |w_d| + \sum_{j \neq d} |w_j|$$

  - 残りのw_j($j \neq d$)は定数であると思う
- 最後の項は w_d に関係ないので無視すると、最小化問題は：

$$w_d^* = \underset{w_d}{\operatorname{argmin}} \sum_{i=1}^{N} \left(w_d \phi_d(x^{(i)}) - \left(y^{(i)} - \sum_{j \neq d} w_j \phi_j(x^{(i)}) \right) \right)^2 + \lambda |w_d|$$

ひとまずは、正則化項を無視して解いてみます

- 以下の1変数最小化問題を解きたい

$$w_d^* = \underset{w_d}{\operatorname{argmin}} \sum_{i=1}^{N} \left(w_d \phi_d(x^{(i)}) - \left(y^{(i)} - \sum_{j \neq d} w_j \phi_j(x^{(i)}) \right) \right)^2 + \lambda |w_d|$$

  - 厄介なのが最後の項 $\lambda |w_d|$
  - この項は$w_d = 0$ において、微分が定義されないために、単純に目的関数の微分を取って 0 とおいて…という方法が使えない
- ひとまず、最後の項を無視して：

$$\tilde{w}_d = \underset{w_d}{\operatorname{argmin}} \sum_{i=1}^{N} \left(w_d \phi_d(x^{(i)}) - \left(y^{(i)} - \sum_{j \neq d} w_j \phi_j(x^{(i)}) \right) \right)^2$$

  を解いてみると、この解は簡単に求まり：

$$\tilde{w}_d = \frac{\sum_{i=1}^{N} \left(y^{(i)} - \sum_{j \neq d} w_j \phi_j(x^{(i)}) \right) \phi_d(x^{(i)})}{\sum_{i=1}^{N} \phi_d(x^{(i)})^2}$$

正則化ナシの解を使って、目的関数を書き換えてみます

- これを用いて損失関数の部分を書き換えてみると、

$$w_d^* = \underset{w_d}{\operatorname{argmin}} \left(\sum_{i=1}^{N} \phi_d(x^{(i)})^2 \right) (w_d - \tilde{w}_d)^2 + \lambda |w_d|$$

  - 定数項は最小化に関係ないので無視した
- ここで、今後の表記を簡単にするために

$$\gamma \equiv \frac{\lambda}{2 \sum_{i=1}^{N} \phi_d(x^{(i)})^2}$$

  とおく
- γ を使って目的関数を書きなおすと：

$$w_d^* = \underset{w_d}{\operatorname{argmin}} \; \tilde{L}(w_d)$$

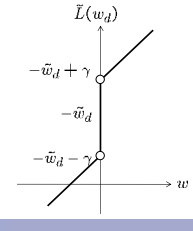
$$\tilde{L}(w_d) = \frac{1}{2} (w_d - \tilde{w}_d)^2 + \gamma |w_d|$$

目的関数の微分を考えてみます（場合分け）

- $\tilde{L}(w_d) \equiv \frac{1}{2} (w_d - \tilde{w}_d)^2 + \gamma |w_d|$ の微分を計算してみると

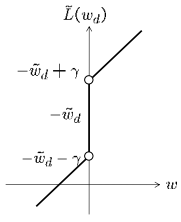
$$\frac{dL(w_d)}{dw_d} = \begin{cases} w_d - \tilde{w}_d + \gamma & (\text{if } w_d > 0) \\ w_d - \tilde{w}_d - \gamma & (\text{if } w_d < 0) \\ \text{undefined} & (\text{if } w_d = 0) \end{cases}$$

  - $w_d = 0$ のとき：
    - $|w_d|$が微分できないため定義されない
  - $w_d > 0$ のとき：
    - 傾き1で切片が$-\tilde{w}_d + \gamma$の一次関数
  - $w_d < 0$ のとき:
    - 傾き1で切片が $-\tilde{w}_d - \gamma$の一次関数
- これが 0 になる w_dを探す



求まった解を見てみると、確かに「疎」な傾向が見えてきます

- グラフが 0 と交わる w_d を探す
- 場合1：$-\tilde{w}_d + \gamma < 0$ すなわち $\tilde{w}_d > \gamma$ のときには解は $w_d^* = \tilde{w}_d - \gamma$
- 場合2：$-\tilde{w}_d - \gamma < 0$ すなわち $\tilde{w}_d < -\gamma$ のときには解は $w_d^* = \tilde{w}_d + \gamma$
- 場合3：$-\gamma \leq \tilde{w}_d \leq \gamma$ のとき
  - $w_d^* > 0$ とすると $w_d^* = \tilde{w}_d - \gamma \leq 0$ となり矛盾
  - $w_d^* < 0$ とすると $w_d^* = \tilde{w}_d + \gamma \geq 0$ となり矛盾
  - 解は必ず存在するので、従って $w_d^* = 0$ でないと困る
- 場合3を見ると、正則化ナシの解 $\tilde{w}_d$ が 0に近いところでは、L_1正則化の解が 0 になる（→ 疎になる）



分類問題

分類問題とは：離散的な出力を持つ条件付き確率分布を推定する問題です

- 教師つき学習は、入力 x が与えられた時の出力yの条件付き確率分布$P(y|x)$を、N 個の入出力ペアである訓練データ集合$\{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$ をもとに推定する問題
- 出力が実数値 $y \in \mathcal{R}$である場合が「回帰」
- y が離散値、出力の取りうる集合$\mathcal{Y}$が：
  - 2クラス分類：+1 と -1 の2値
  - 多クラス分類：$\{1,2,\ldots,C\}$のC通り

  のような場合が「分類」
- 分類問題における出力を、特別に**クラス**と呼ぶ。

回帰手法でも分類問題を解くことは可能ですがそれは必ずしもベストの方法ではありません

- 目標とする出力値を $y^{(i)} \in \{+1, -1\}$とすれば、形式的には2クラス分類に回帰手法を適用することは可能
- 多クラス分類の場合にも、データが C 個のクラスの各々に属するかどうかを{+1, -1}で表し、出力値をベクトル $y^{(i)} = (y_1^{(i)}, y_2^{(i)}, \ldots, y_C^{(i)})$ で書けば（ベクトル出力に一般化した）回帰手法を適用できる
- しかし、2乗誤差を損失関数とする線形回帰では、確率モデルの仮定として「線形モデルの出力にガウス的なノイズが載る」と仮定
  - これは、出力が{+1,-1}のどちらかであるとする分類モデルの背後にあるべき確率モデルとしては適当ではない

より分類に特化したモデルとして、ロジスティック回帰を中心に紹介します

- ロジスティック回帰：より分類という目的を直接的にモデル化したモデル
- モデル（パラメータ）の学習は回帰のときのように、逆行列1回のようなシンプルではなくなる

分類問題の応用

分類の応用は、結構あります

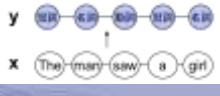
- 2クラス分類：
  - 購買予測：ある人xが商品を購入する(y= +1)か否(y = -1)か予測
  - 活性予測：ある化合物xが、活性をもつ(y = +1)か否(y = -1)か予測
  - 与信：ある人xが、融資したお金を返済してくれる(y = +1)か否(y = -1)か予測
- 多クラス分類：
  - テキスト分類：ある文書xが、どのカテゴリに属するか（$y \in$ {政治, 経済, スポーツ, …}）を判別
  - 画像認識：ある画像xに映っているものが何か（$y \in$ {自動車, 家, 飛行機, …}）を識別
  - 行動識別：ある人に取り付けたセンサーデータxからその人の行動（$y \in$ {走っている, 歩いている, …}） を識別

2クラス分類の複雑なケース：関係の予測

- 2つのデータの間の「関係の有無」を予測するような場合も2クラス分類の特殊なケースとして考えられる
- 2つのデータxとx'の間に関係がある(y = +1)か否(y = -1)かを予測
- たとえば：
  - タンパク質の相互作用予測：2つのタンパク質（データ）の間に物理的な相互作用（チームで働くなど）があるかを予測
  - 購買予測：顧客と商品の間の購買関係を予測
- 入力が2データのペアであるような2クラス分類問題になるので、通常の（1データに対する）2クラス分類問題の一般化になっている
- 関係にも複数種類がある場合には、多クラス分類になる

多値分類の極端なケースとしては、構造データのラベルづけ問題などがあります

- 構造データのラベルづけ問題：自然言語処理における品詞付け問題
  - 文（単語列）x に含まれる単語のそれぞれに対して、その品詞（{名詞, 動詞, 副詞, …}など複数ありうる）を割り当てるタスク
  - 他、固有表現抽出、DNAからの遺伝子発見の問題等色々
- それぞれの単語を独立のものと考えれば、単語の多クラス値分類問題
- 文に含まれる単語全てに対する品詞の組み合わせ（1単語目が「名詞」で2単語目が「動詞」など）を1つのクラスとして考えると、クラスの数は (可能な品詞数)$^{(文中の単語数)}$となり、非常に多くのクラスを持つような分類問題になる



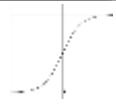
分類のためのモデル：ロジスティック回帰

ロジスティック回帰モデル（2クラスの場合）

- 2クラス{+1,-1}の場合のロジスティック回帰モデル：

$$P(y = +1|x; \mathbf{w}) = \sigma(\mathbf{w}^\top \mathbf{x})$$

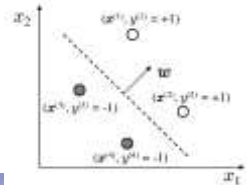
$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$



- 関数 σ は、シグモイド関数（もしくはロジスティック関数）と呼ばれ、実数値を (0,1) の間の値に変換する（＝確率化する）
- σ の中身は、ちょうど線形回帰と同じ形
- 「線形回帰モデルの出力する実数値を確率値に変換している」
- モデルパラメータは線形回帰のときと同じく($\mathbf{w}$, b)のD+1個
- $\mathbf{w}^\top \boldsymbol{x}$ が大きいほど P(y=+1|$\boldsymbol{x}$; $\mathbf{w}$)も大きいので、$\mathbf{w}$の各次元は$\boldsymbol{x}$の各次元（が正の値をもつこと）が、クラス+1への所属にどの程度貢献しているかを表す

ロジスティック回帰モデルは、特徴空間を2つに分割するモデルです

- モデルは $P(y = +1|x; \mathbf{w}) = \sigma(\mathbf{w}^\top \mathbf{x})$
- ロジスティック回帰を用いた予測は、$P(y = +1|\boldsymbol{x}; \mathbf{w}) > 0.5$ であれば +1、$P(y = +1|\boldsymbol{x}; \mathbf{w}) < 0.5$ であれば -1 と予測すればよい
  - それぞれ $\mathbf{w}^\mathsf{T}\boldsymbol{x} > 0$ と $\mathbf{w}^\mathsf{T}\boldsymbol{x} < 0$ に相当する
- D次元の超平面である $\mathbf{w}^\mathsf{T}\boldsymbol{x} = 0$を境にクラス+1と-1が分割されている



分類問題の定式化

我々の目的は、将来のデータに対して、その正解に高い確率（の対数）を与えるモデルを得ることです

- 我々の目的は（一応）条件付き確率分布$P(y|x; \mathbf{w})$を推定すること
- その推定の良さをどのように定義したらよいだろうか？
- 与えられた訓練データ集合$\{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$に対してでなく、出力未知の入力$\boldsymbol{x}^{(N+1)}$に対して、対応する正しい出力$y^{(N+1)}$を出力すること
- いいかえると、入力$\boldsymbol{x}^{(N+1)}$に対する$y^{(N+1)}$の確率（の、なぜか対数）

$$\log P(y^{(N+1)}|\mathbf{x}^{(N+1)}; \mathbf{w})$$

を大きくすることと言える

  - 確率0.1と0.2の違いと、0.8と0.9の違いはどちらも0.1の差であるが我々は前者の差を重く見る（対数をとる＝比で考える）

我々の（最大化すべき）目的関数は、未知データの対数尤度の、真のデータ分布による期待値です

- $\log P(y^{(N+1)}|\boldsymbol{x}^{(N+1)}; \mathbf{w})$を大きくするのが目的とするとして、我々は、次にどのような$(\boldsymbol{x}^{(N+1)}, y^{(N+1)})$が来るかは知らないので、これを直接大きくすることはできそうにない
  - 訓練データ集合$\{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$を使うしかない
- $(\boldsymbol{x}^{(N+1)}, y^{(N+1)})$を生み出している確率分布$Q(\boldsymbol{x}^{(N+1)}, y^{(N+1)})$を考えてみる
- 我々が最大化したいのは、$\log P(y^{(N+1)}|\boldsymbol{x}^{(N+1)}; \mathbf{w})$ の $Q(\boldsymbol{x}^{(N+1)}, y^{(N+1)})$についての期待値であるといえる

$$E_{Q(x^{(N+1)}, y^{(N+1)})}\left[\log P(y^{(N+1)}|\mathbf{x}^{(N+1)}; \mathbf{w})\right]$$
$$\equiv \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} Q(x^{(N+1)}, y^{(N+1)}) \log P(y^{(N+1)}|\mathbf{x}^{(N+1)}; \mathbf{w})$$

未知データの対数尤度の、真のデータ分布による期待値は、訓練データの対数尤度で近似されます

- 訓練データおよび将来のデータ$\{(x^{(i)}, y^{(i)})\}_{i=1}^{N+1}$は、すべて同一の分布 Q から独立に取り出されたものとすると
  - 大数の法則：独立なサンプルの平均は、サンプル数を大きくすると、その期待値に近づく

から

$$\frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w})$$
$$\xrightarrow{n\to\infty} E_{Q(x^{(N+1)}, y^{(N+1)})}\left[\log P(y^{(N+1)}|\mathbf{x}^{(N+1)}; \mathbf{w})\right]$$

- 期待値の代わりに対数尤度の和で代用し、これを最大化するようにパラメータを決定する ＝ つまり最尤推定

なお、事後確率最大化（MAP推定）からは L_2正則化項が出てくるのでした

- 最尤推定によってパラメータを決定すれば：

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w})$$

- 事後確率最大化（MAP）推定では：

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w}) + \log P(\mathbf{w})$$

- 事前分布が正規分布である場合には：

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w}) - \lambda \| \mathbf{w} \|_2^2$$

  - L_2正則化項が出てくる
  - 以降はMAP推定／正則化を前提として話をすすめる

学習アルゴリズム：最急勾配法

分類問題の最適解は閉じた形で求まらないことが多いのでパラメータを少しづつ改善する方法をとります

- MAP推定の目的関数を最大化する方法を考える

$$L(\mathbf{w}) \equiv \frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|\mathbf{x}^{(i)};\mathbf{w}) - \lambda \parallel \mathbf{w} \parallel_2^2$$

  - これが最大になるような $\mathbf{w}$ を求めたい
- 回帰の場合とは異なり、これを $\mathbf{w}$ で微分し $\mathbf{0}$ と置いても、連立方程式のような簡単な形（閉じた形の解）は得られない
- そこで、$\mathbf{w}$ を少しづつ改善していくステップを繰り返す
  - 現時点でのパラメータを $\mathbf{w}^{(t)}$ とすると、これを、目的関数が改善するような $\mathbf{w}^{(t+1)}$ に更新する

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \mathbf{d}^{(t)}$$

最急勾配法：もっとも目的関数が増加する方向にパラメータを更新する簡便な方法です

- **最急勾配法：** 目的関数が最も大きい方へ変化する方向$\nabla(\mathbf{w}^{(t)})$ に向かってパラメータを更新する

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \mathbf{d}^{(t)}$$

$$\Downarrow$$

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \eta^{(t)}\nabla(\mathbf{w}^{(t)})$$

- $\eta^{(t)}$ は学習率と呼ばれる、更新の度合いを決めるパラメータ

学習率 $\eta^{(t)}$ の決定方法：線形探索などを行います

- $\eta^{(t)}$ の簡単な決定方法：
  - 十分に小さい定数に取る
  - $\eta^{(t)} = 1/t$ などとする（ステップ幅が次第に小さくなっていく）
- もう少しきちんと決めたい場合には $\eta^{(t)}$ の線形探索を行う
  つまり、1変数最適化問題を解く

$$\eta^{(t)} = \underset{\eta \geq 0}{\operatorname{argmax}}\ L(\mathbf{w}^{(t)} + \eta\mathbf{d}^{(t)})$$

- $\eta^{(t)}$ の簡便な探索方法としては、η を適当な初期値から初めて：
  1. もし $L(\mathbf{w}^{(t)}+\eta\mathbf{d}) > L(\mathbf{w}^{(t)})$ となるならば、その η を $\eta^{(t)}$ として採用
  2. そうでないならば η を1/2倍してステップ1へ

パラメータの更新式における勾配$\nabla(\mathbf{w}^{(t)})$を具体的に計算してみます

- 目的関数を $\mathbf{w}$ で偏微分すると、

$$\frac{\partial L(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{P(y^{(i)}|x^{(i)};\mathbf{w})} \frac{\partial P(y^{(i)}|x^{(i)};\mathbf{w})}{\partial \mathbf{w}} - 2\lambda\mathbf{w}$$

- モデルとしてロジスティック回帰を用いることにする、
  つまり $P(y^{(N+1)} = +1|\, \boldsymbol{x}^{(N+1)};\mathbf{w}) \equiv \sigma(\mathbf{w}^{\mathsf{T}}\boldsymbol{x}^{(i)})$ とする
- 以下の2つの事実：
  - $\dfrac{\partial P(y^{(i)}|x^{(i)};\mathbf{w})}{\partial \mathbf{w}} = \dfrac{\partial P(y^{(i)}|x^{(i)};\mathbf{w})}{\partial z}\dfrac{\partial z}{\partial \mathbf{w}}$
  - シグモイド関数の微分：$\partial\sigma(z)/\partial z = 1 - \sigma(z)$

より偏微分が計算できる：

$$\frac{\partial L(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{N}\sum_{i=1}^{N} \left(\frac{P(y^{(i)} = -1|x^{(i)};\mathbf{w})}{P(y^{(i)} = +1|x^{(i)};\mathbf{w})}\right)^{y^{(i)}} \mathbf{x}^{(i)} - 2\lambda\mathbf{w}$$

多クラス分類

多クラスのロジスティック回帰モデル：クラスの数だけパラメータベクトルを用意します

- ロジスティック回帰モデルを、多クラスの場合に拡張する
  - 後に配列データのラベル付けモデル（条件付確率場）に拡張される
- クラスの集合 $\mathcal{Y}$ を$\{1, 2, \ldots, C\}$に対するCクラスのロジスティック回帰モデルは：

$$P(y|\mathbf{x};\{\mathbf{w}^{(c)}\}_{c\in\mathcal{Y}}) \equiv \frac{\exp(\mathbf{w}^{(y)\top}\mathbf{x})}{\sum_{c\in\mathcal{Y}}\exp(\mathbf{w}^{(c)\top}\mathbf{x})}$$

  - 各クラス $c \in \mathcal{Y}$ ごとにD次元の$\mathbf{w}^{(c)}$が定義されモデル全体としては C D個のパラメータがあることになる
  - expの中には線形回帰モデルの形が現れており、$\mathbf{w}^{(c)}$ は特徴ベクトルのそれぞれの特徴のクラス c への貢献度を表す
  - 指数を取ることで0以上の値に変換し、さらに正規化することで (0,1) の間の確率値に変換している

分類におけるL1正則化

分類の場合でもL1正則化は有効です

- 回帰の場合と同じく、分類の場合にも、L1正則化によって、疎なパラメータを得たい場合が多々ある
  - パラメータwの1-ノルム：
    $$\| \mathbf{w} \|_1 = |w_1| + |w_2| + \cdots + |w_D|$$
    をペナルティ項として小さくすることで、多くのw_dが0になる効果
- 特に、データ数と比較して、特徴ベクトルの次元が高いような場合
  - 文書分類：文書中に出現する単語を用いた特徴ベクトル（bag-of-words）表現がしばしば用いられる
  - マイクロアレイ診断：各遺伝子の発現量を、患者の特徴ベクトルとして用いる
- これは数千～数十万次元にもなりうるため、大幅な次元削減が必要になることがある

分類の場合は、パラメータの最適解を閉じた形で得るのは難しいのでパラメータ徐々に改善する方式をとります

- L1正則化を用いた場合の目的関数：
$$L(\mathbf{w}) \equiv \frac{1}{N} \sum_{i=1}^{N} \log P(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w}) - \lambda \| \mathbf{w} \|_1$$
- 分類問題の場合、この解を閉じた形で得るのは困難であるので、パラメータ$\mathbf{w}^{(t)}$から$\mathbf{w}^{(t+1)}$への更新によって、解を徐々に改善していく
$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \boldsymbol{\Delta}^{(t)}$$
- 更新量$\boldsymbol{\Delta}^{(t)}$は目的関数$L$をなるべく大きくするように決定する
$$L(\mathbf{w}^{(t)} + \boldsymbol{\Delta}^{(t)}) > L(\mathbf{w}^{(t)})$$

L1正則化分類のアルゴリズム

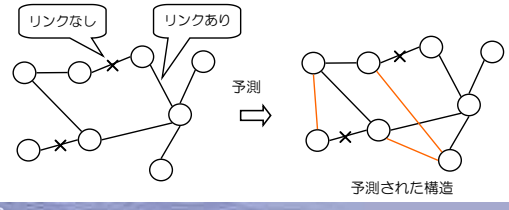
- 以下の2ステップを繰りかえす

1. 最急勾配法を適用し、中間的な解 $\tilde{w}_d$ を得る。
$$\tilde{\mathbf{w}} \equiv \mathbf{w}^{(t)} + \eta^{(t)} \nabla(\mathbf{w}^{(t)})$$
1. 以下の丸め操作によって新しいパラメータ $\mathbf{w}^{(t+1)}$を得る
$$w_d^{(t+1)} = \begin{cases} \tilde{w}_d - \frac{\lambda}{\eta^{(t)}} & (\text{if } \tilde{w}_d > \frac{\lambda}{\eta^{(t)}}) \\ \tilde{w}_d + \frac{\lambda}{\eta^{(t)}} & (\text{if } \tilde{w}_d < -\frac{\lambda}{\eta^{(t)}}) \\ 0 & (\text{otherwise}) \end{cases}$$

データの組に対する予測

リンク予測問題は、部分的に観測されているネットワーク構造から、残りの構造を推定する問題です

- 入力: 一部が欠けたネットワーク構造
  - リンクありのノードペア
  - リンクのないノードペア
- 出力： リンクの有無が未知のノードペアについてのリンク予測



単純に「リンク指標」によってリンク予測を行うこともあります

- 社会ネットワーク研究やスケールフリーネットワーク研究においていくつかの指標が提案されている
- これらは、ネットワークの構造遷移モデルに基づいている
  - 共通の隣接ノードが多いほど、リンクが張られやすいとするモデル「友達の友達は友達」

$$\text{common neighbors} := |\Gamma(i) \cap \Gamma(j)|$$

$\Gamma(i)$ はノード i の隣接ノード集合

  - common neighbors の重み付きバージョン「友達が少ない人ほど付き合いは深い」

$$\text{Adamic/Adar} := \sum_{k \in \Gamma(i) \cap \Gamma(j)} \frac{1}{\log |\Gamma(k)|} \qquad \text{Jaccard's coefficient} := \frac{|\Gamma(i) \cap \Gamma(j)|}{|\Gamma(i) \cup \Gamma(j)|}$$

（こちらは情報検索で使われる指標）

  - 遠距離の影響もとりいれたcommon neighbors

$$\text{Katz}_\beta := \sum_{l=1}^{\infty} \beta^l |\text{paths}_{i,j}^{\langle l \rangle}|$$

$\text{path}_{i,j}^{\langle l \rangle}$はノード i から j への長さ l のパスの集合

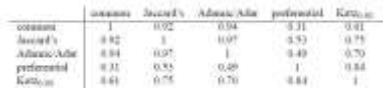
  - バラバシのpreferential attachment モデル （友人が多いほど、より多くの友人を得る）

$$\text{preferential attachment} := \Gamma(i) \cdot |\Gamma(j)|$$

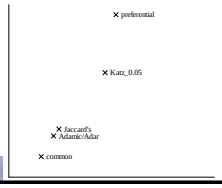
Liben-Nowell & Kleinberg: *The Link Prediction Problem for Social Networks*, CIKM 2004

参考： 各指標の相対的な関係

- 代謝ネットワークデータに対する各手法の予測の Spearman 相関（順序の相関）
  - 大きいほど予測が似ている



  - 相関のMDSによる視覚化
    - preferential attachmentと common neighborsが一番遠い

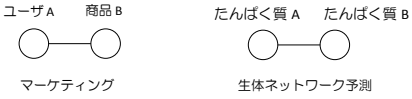


分類問題の一般化として、2つ（複数）の入力をもつ分類問題を考えます

- これまでは入力xに対して出力yを予測するという一対一の関係を扱っていた
- 時によって2つの入力xとx'の組（より一般的にはm個の入力）に対して出力yを予測したい場合がある
- 例：ネットワーク構造の予測問題
  - ネットワーク構造のリンク構造が部分的に与えられた（いくつかのノード対に関して、それらの間にリンクがあるかないかという情報が与えられた）ときに、残りの部分についてのリンク構造の予測を行う問題
    - タンパク質の相互作用予測：2つのタンパク質（データ）の間に物理的な相互作用（チームで働くなど）があるかを予測
    - 購買予測：顧客と商品の間の購買関係を予測

リンク予測のひとつの捉え方は「2つのノードのペアの分類問題」です

- 任意の2つのノードの間のリンクの強さを、予測する
  - マーケティング：ユーザーと、商品の間の購買関係を予測する
    - 買う=リンクあり、買わない=リンクなし
  - 生体ネットワーク予測：2つのたんぱく質の相互作用の有無を予測する
    - 相互作用あり=リンクあり、相互作用なし=リンクなし
- 正解（リンクの有無）は、いくつかのペアについては与えられる
  - これをもとに正解未知のペアについて、リンクの強さを予測する
  - これも教師つき学習の問題のひとつ



マーケティング　　生体ネットワーク予測

2つの入力をもつ条件付き分布を推定する問題を考えます

- リンク予測の一番シンプルな捉え方：2つのノードの2クラス分類
  - 2つのノードx とx' に対し、それらの間にリンクが存在する（+1）かしないか（-1）をクラスラベルとする
- つまり、2つの入力データが与えられた時の、出力の条件付き分布 $P(y|\, x, x')$を推定する
- x と x' は同じ集合$\mathcal{X}$ に属していても良いし、別々のノード集合に属して（xは集合$\mathcal{X}$ に、x'は集合$\mathcal{X}'$に属する）もよい
  - $\mathcal{X}$ を顧客の集合、$\mathcal{X}'$を商品の集合とすると、ある顧客 $x \in \mathcal{X}$ がある商品$x' \in \mathcal{X}'$を購入するかどうかを予測するというマーケティングの文脈での予測問題を考えることができる
- より一般的に、複数種類の関係がある場合（多クラス分類）や、数値的な関係がある場合（回帰）なども考えられる

2入力のロジスティック回帰モデルを考えます

- 入力の組 (x, x') についての分類問題を解くために、入力の組が与えられた時の出力の条件付き確率$P(y|x, x')$をモデル化する
- ロジスティック回帰モデルを2入力に拡張したモデルを考える：

$$P(y = +1|x, x'; \mathbf{w}) \equiv \sigma(\mathbf{w}^\top \psi(x, x'))$$

  - $\sigma(z) \equiv (1+\exp(-z))^{-1}$ ：シグモイド関数
  - $\boldsymbol{\psi}(x, x')$ ： 2つの入力 xと x' の組み合わせに対する特徴ベクトル
  - $\mathbf{w}$：パラメータベクトル
- 通常のロジスティック回帰との違いは、特徴ベクトルが2つの入力の組み合わせに対して定義されているところ
- 2つの入力それぞれの特徴ベクトル$\boldsymbol{x}$ および$\boldsymbol{x}'$ が予め与えられているものとして、これらを利用して$\boldsymbol{\psi}(x, x')$を設計することが重要

組み合わせ特徴ベクトルの定義として良く用いられるのは2つの入力それぞれの特徴の組み合わせを用いる表現です

- 2つの入力ペアに対する特徴ベクトル $\boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{x}')$ をそれぞれの特徴ベクトル $\boldsymbol{x}$ および $\boldsymbol{x}'$ の特徴の組み合わせで構成する
- つまり、 $\boldsymbol{x}$ および $\boldsymbol{x}'$ の次元がそれぞれ D および D' であるとするときに、これらの間の DD' 個の組み合わせ特徴を定義する：

$$\psi(x, x') \equiv x \otimes x'$$

  - $\otimes$ はクロネッカー積と呼ばれる演算子
  - 要素ごとに書けば $\psi_{(i-1)D'+j}(x, x') \equiv x_i \; x'_j$

一旦特徴ベクトルが定義できれば、あとはこれまでと同じように学習を行うことができます

- 訓練データ集合：
  - 各訓練データを2つの入力の組 $(\boldsymbol{x}^{(i)}, \boldsymbol{x}'^{(i)})$ と対応する出力 $y^{(i)}$ とする
  - これを N 組集めたもの $\{(\boldsymbol{x}^{(i)}, \boldsymbol{x}'^{(i)}, y^{(i)})\}_{i=1}^{N}$
- 例えば、2-ノルムに基づく正則化（もしくは正規分布を事前分布とする事後確率最大化）を用いたとすれば、最適化問題：

$$\mathbf{w}^* \equiv \underset{\mathbf{w}}{\operatorname{argmax}} \; \frac{1}{N} \sum_{i=1}^{N} \log P(y^{(i)} | \boldsymbol{x}^{(i)}, \boldsymbol{x}'^{(i)}; \mathbf{w}) - \lambda \parallel \mathbf{w} \parallel_2^2$$

  を解けば、最適なパラメータ $\mathbf{w}^*$ が求まる
- このモデルでは必然的に特徴ベクトルの次元は DD' と高くなってしまうため、パラメータを疎にするために L_1 正則化を用いるのもよい

行列パラメータをもつモデル

2入力のロジスティック回帰は行列を用いて表現できます

- 2つの入力の組に対する特徴ベクトルは、個々の入力の特徴の組み合わせで作られることから、特徴ベクトルを行列で自然に表現することができる
- つまり、 DD' 次元の特徴ベクトル $\psi(\boldsymbol{x}, \boldsymbol{x}') \equiv \boldsymbol{x} \otimes \boldsymbol{x}'$ の定義の代わりに、$D \times D'$ の特徴行列 $\boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{x}')$ を次のように定義する：

$$\boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{x}') \equiv \boldsymbol{x} \otimes \boldsymbol{x}'^{\top}$$

- これに対応して、パラメータのほうも $D \times D'$ の行列 $\mathbf{W}$ として書くことにすると、ロジスティック回帰モデルは：

$$P(y = +1 | \boldsymbol{x}, \boldsymbol{x}'; \mathbf{W}) \equiv \sigma(\mathrm{Tr}\mathbf{W}^{\top}\boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{x}'))$$

  - Tr は行列のトレースを表す
  - 上式におけるTrの中身はちょうど $\mathbf{W}$ と $\boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{x}')$ の要素ごとの積の和

学習の目的関数も同様に行列を用いて書き直すことができます

- 学習の目的関数も同じように書き直すことができる
- 例えば、2-ノルムに基づく正則化の場合には、行列表現を使うと：

$$\mathbf{W}^* \equiv \underset{\mathbf{W}}{\operatorname{argmax}} \; \frac{1}{N} \sum_{i=1}^{N} \log P(y^{(i)} | \boldsymbol{x}^{(i)}, \boldsymbol{x}'^{(i)}; \mathbf{W}) - \lambda \parallel \mathbf{W} \parallel_F^2$$

  - $\| \cdot \|_F^2$ はフロベニウスノルム（行列の全ての要素の2乗和）
    - これは行列をベクトルだと思って2-ノルムをとったときに等しい

入力とパラメータを行列だとみなすことによって、新たな方向性が見えきます

- 入力行列は必ずしもベクトルのクロネッカー積による定義のようにランク1の行列である必要はなく、一般の $D \times D'$ の行列 $\boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{x}')$ であるとしても問題ない
  - 画像は各要素が色の濃淡等を表す行列で自然に表現できる
  - 脳波解析などにおいて、複数の時系列に対する分類を行いたい場合、特徴の定義として、時系列間の相関係数を（対称）行列として表現することがある
- しかし、モデルを行列で表現したところで、これはベクトル表現による以前のモデルと等価であり、これらに伴う最適化問題もまた等価であるため、表現の違いによる本質的な変化はない
- 入力が行列という構造をもつことが意味をもつためには、モデルの学習において行列の構造を明示的に利用する必要がある

「パラメータが行列であること」を明示的に扱うためには行列の複雑さを正則化項に考慮する必要があります

- 入力が行列という構造をもつことが意味をもつためには、モデルの学習において行列の構造を明示的に利用する必要がある
- その行列構造をどこに入れるか？→ 正則化項に入れる
- 最適化問題における正則化項であるフロベニウスノルム（2-ノルム）の代わりに、行列の複雑さを表す別の指標を考えることで、学習において行列の構造を明示的に考慮する

$$\mathbf{W}^* \equiv \operatorname*{argmax}_{\mathbf{W}} \frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|x^{(i)}, x'^{(i)}; \mathbf{W}) - \lambda \| \mathbf{W} \|_F^2$$

⇩

？

行列パラメータの複雑さを測るひとつの基準は、行列のランク（階数）です

- 行列の複雑さの1つの指標として行列のランク（階数）が考えられる
- $\mathbf{W}$のランク：$\mathbf{W}$ の固有値（$\min\{D, D'\}$個ある）を $\boldsymbol{\mu} \equiv (\mu_1(\mathbf{W}), \mu_2(\mathbf{W}), \ldots, \mu_{\min\{D,D'\}}(\mathbf{W})$ としたときの、$\boldsymbol{\mu}$ の非零要素の数
- ランクが小さいこと、つまり、固有値の集合における非零要素が少ないことが、行列の複雑さが低いことを表す
  - $\boldsymbol{\mu}$の非零要素だけを取り出して並べたベクトルを$\boldsymbol{\mu}_+$と書くことにすると、$\mathbf{W}$ は以下のように分解できることが知られている：
    $$\mathbf{W} = \mathbf{U}\mathrm{diag}(\boldsymbol{\mu}_+)\mathbf{V}^\top$$
  - $\mathbf{diag}(\boldsymbol{\mu}_+)$ は $\boldsymbol{\mu}_+$を対角成分としてもつような対角行列
  - $\boldsymbol{\mu}_+$の非零要素の数をRとすると、$\mathbf{U}$と$\mathbf{V}$はそれぞれ $D \times R$および $D' \times R$の行列であり、ランクRが小さいほど、$\mathbf{W}$を小さな行列（$\mathbf{U}$と$\mathbf{V}$）で表現できることがわかる

ランクを落とすための正則化項として、行列の固有値の1-ノルムを用いることができます

- 行列のフロベニウスノルムに代わる正則化項として、行列のランク$R(\mathbf{W})$を用いるのがよいという気がしてくるが、残念ながら$R(\mathbf{W})$は凸関数ではなく、最適化の観点からは正則化項に不向きである
- そこで、ランクの代わりに固有値ベクトル $\boldsymbol{\mu}$ の1-ノルムを用いることを考える（これは凸関数であることが知られている）
- そこで、$\mathbf{W}$の固有値ベクトル$\boldsymbol{\mu}(\mathbf{W})$に対する$L_1$ 正則化項を入れる：

$$\mathbf{W}^* \equiv \operatorname*{argmax}_{\mathbf{W}} \frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|x^{(i)}, x'^{(i)}; \mathbf{W}) - \lambda \| \boldsymbol{\mu}(\mathbf{W}) \|_1$$

- L_1 正則化は結果として得られるパラメータの多くを0にする効果があるため、これを固有値に対して用いることによって、多くの固有値を0にし、結果として行列のランクを低くする

固有値の1-ノルムを用いた正則化は、近年、盛んに利用されつつあります

- 行列の固有値ベクトルに対する1-ノルムはトレースノルムや核(nuclear)ノルムと呼ばれ、これを用いた正則化は、トレースノルム正則化、スペクトル正則化などと呼ばれる
- 協調フィルタリングやネットワーク予測、マルチタスク学習など行列がパラメータとして現れるに様々な場面において、近年盛んに用いられている

構造データのモデリング

構造データのモデリング、とくに配列構造のラベリング問題について紹介します

- 構造データのモデリング
- 配列データのラベル付け問題
- 条件付き確率場（CRF）
- 条件付き確率場における予測
- 条件付き確率場の事後確率最大化学習

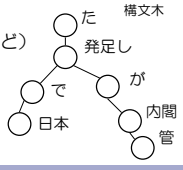
ポイントは「動的計画法」(dynamic programming)

様々な応用分野において、配列や木、グラフなどの構造をもったデータが現れる場面があります

- 様々な分野において、入出力に配列、木、グラフなどの複雑な構造をもったデータを扱う必要がしばしば生じる
  - 自然言語処理： 文、構文木、文書間リンク
  - バイオインフォマティクス：DNA、RNA、タンパク質

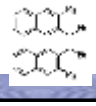
DNA

- しかし、これまでに紹介してきた方法で、これら構造データをどのように扱ったらよいかは自明ではない
- 構造データを扱う学習問題は、入力xのもつ構造の種類（配列、木、グラフなど）の別のほか、出力 y における構造の有無によっても大別される
  - 出力 y に構造が無い場合
  - 出力 y に構造がある場合



出力が構造をもたない場合の例

- 出力 y に構造が無く、入力 x のみが構造をもっている場合
- 出力 y はこれまでに考えてきた回帰や分類などと同じく、1次元（もしくは複数次元）の実数値もしくは離散値をもつ
- 応用としては：
  - HTMLやXMLなどの半構造文書を木もしくはグラフとして表現し文書の構造やレイアウトなどをもとに、文書の性質を予測
  - DNA配列やタンパク質のアミノ酸配列をもとに、それらの機能を予測
  - 化合物の分子構造をグラフとして表現し、その化学的性質を予測



出力が構造をもつ場合の例

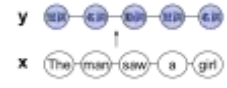
- より複雑な状況としては、出力も構造をもつような場合がある
  - 一般的には、構造から構造に変換する問題として捉えられる
- 配列ラベル付け問題（配列構造の各要素に対して出力値を与える）：
  - 入力 x ：配列構造
  - 出力 y ：同じ長さの配列構造

x_1	x_2	...	x_T
y_1	y_2	...	y_T

- 多くの問題が配列のラベル付け問題として定式化される
  - 自然言語処理：品詞付けや固有表現抽出など
  - バイオインフォマティクス：遺伝子発見やタンパク質の2次構造予測など
- 入出力の構造は必ずしも同じ種類である必要はない
  - 自然言語処理：構文解析（配列→木）
  - バイオインフォマティクス：RNA構造予測（配列→木）

自然言語処理での構造ラベル付与問題の例（列構造）

- 品詞タグ付与タスク
  - **単語列に対して品詞ラベル（動詞、名詞…）を付与するタスク**



- 固有表現抽出タスク
  - **人名・組織名等の固有表現をテキスト中から抽出するタスク**
  - **単語列に対して固有表現の「始まり（B-XXX）」と「続く（I-XXX）」、「それ以外（O）」を示すラベルを付与**

配列データのラベリング問題

配列構造のラベル付け問題とは、入力配列と同じサイズの出力配列を予測する問題です

- 配列構造のラベル付け問題：出力が構造をもつ一番簡単な場合
  - 入力：長さTの配列 $x = (x_1, x_2, \ldots, x_T)$
  - 出力（予測）：同じ長さの配列 $y = (y_1, y_2, \ldots, y_T)$

x_1	x_2	...	x_T
y_1	y_2	...	y_T

- これまでの枠組みでいえば、入力配列 x が与えられたもとでの出力配列 y の条件付き確率 $P(y \mid x)$ を得るのが目的
- 入力配列のそれぞれの要素 $x_t\ (t=1,\ldots,T)$ に対応する出力配列の要素 $y_t\ (t=1,\ldots,T)$ を予測するため、配列のラベル付けと呼ばれる
- 例えば、品詞付け問題は、文が単語の列として与えられたときに、各単語に対して適切な品詞を割り振る問題
  - 入力 x ：単語列（各 x_t は「私」「は」「走る」などの単語）
  - 出力 y ：品詞列（各 y_t は「名詞」「助詞」「動詞」などの品詞）

配列のラベル付け問題は、出力ラベル間の依存関係を考慮するという意味で（多クラス）分類問題の拡張となっています

- 原理上は、配列のラベル付け問題は、入力配列の要素それぞれについて独立した分類問題として考えることも可能
  - 文の品詞付けにおいて、ある単語のみ（例えば「私」）を見て、その品詞を予測するならば、これは通常の（多クラス）分類問題
- しかし、多くの場合、独立性の仮定は成り立たない
  - 上の例では、全体として整合性のある品詞列を予測する必要あり
  - 「名詞のあとには助詞が続きやすい」等の品詞間の依存関係を考慮し、各品詞でなく品詞「列」としてまとめて予測する必要あり
- 配列のラベル付け問題は、通常の分類問題の拡張となっている
  - 通常の分類問題と同じように訓練データ集合としては N 個の入出力の組 $\{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$ が与えられる
  - このとき $x^{(i)}$ と $y^{(i)}$ はそれぞれ長さ $T^{(i)}$ の入力配列と出力配列

条件付き確率場（CRF）

（配列に対する）条件付き確率場のモデル

- **条件付き確率場**（Conditional random field; **CRF**）は、入力 x と出力 y が共に構造をもつ条件付き確率分布 $P(y|x)$ を表現するモデル
- ここでは、最も簡単なケースとして配列データのラベル付けのためのCRFを考える
  - 入出力はともに長さ T の配列 $x = (x_1, x_2, \ldots, x_T)$ と $y = (y_1, y_2, \ldots, y_T)$
  - 出力ラベルの取りうる集合 Σ を $\Sigma \equiv \{1,2,\ldots,C\}$ のように定義し、各 y_t は Σ の要素（ $y_t \in \Sigma$ ）とする
- このとき、配列 x が与えられたときに、これに同じ長さのラベル列 $y_t \in \Sigma^T$ を割り当てる確率を：

$$P(y|x;\boldsymbol{\omega}) \equiv \frac{\exp(\boldsymbol{\omega}^\top \boldsymbol{\varphi}(x,y))}{\sum_{c \in \Sigma^T} \exp(\boldsymbol{\omega}^\top \boldsymbol{\varphi}(x,c))}$$

  - $\boldsymbol{\varphi}(x,y)$ は入出力配列 x と y の両方を考慮した特徴ベクトル

条件付き確率場（CRF）の特徴ベクトルの定義：2種類の特徴を用います

- x と y の両方にまたがるCRFの特徴ベクトル $\boldsymbol{\varphi}(x,y)$ はどのように設計すればよいだろうか？
- 配列ラベル付けのためのCRFにおいてよく用いられるのは：
  - 配列中での位置 t における入力 x_t と出力 y_t の組み合わせによる特徴 と
  - ひとつ前の位置におけるラベル y_{t-1} と現在位置におけるラベル y_t の組み合わせによる特徴 である

x_1	x_2	...	x_{t-1}	x_t	...	x_T
y_1	y_2	...	y_{t-1}	y_t	...	y_T

位置 t における入力 x_t と出力 y_t の組み合わせによる特徴の定義は多クラスロジスティック回帰の別形式のものと同じです

- 配列中での位置 t における入力 x_t と出力 y_t の組み合わせによる特徴
- ロジスティック回帰の別表現でも用いた入力と出力にまたがる特徴ベクトルの定義と同様に、各位置 t に対して以下を定義：

$$\varphi_t^{\mathcal{X}\Sigma}(x_t, y_t) \equiv (\delta(y_t = 1)\phi(x_t)^\top, \delta(y_t = 2)\phi(x_t)^\top, \ldots, \delta(y_t = C)\phi(x_t)^\top)^\top$$

  - $\delta(\cdot)$：括弧の中身が成立するなら1を、しないならば0をとる
- 各 x_t の特徴ベクトル $\boldsymbol{\phi}(x_t)$ を D 次元とすると、これは DC 次元のベクトル

$$\varphi_t^{\mathcal{X}\Sigma}(x_t, y_t)$$

$\phi(x_t)$
y_t

ひとつ前の位置のラベル y_{t-1} と現在位置のラベル y_t の組み合わせ特徴は、連続するラベルの組み合わせを用います

- ひとつ前の位置におけるラベル y_{t-1} と現在位置におけるラベル y_t の組み合わせによる特徴は、隣り合う2つの位置のラベルを各位置 t に対して以下を定義：

$$\begin{aligned}\varphi_t^{\Sigma\Sigma}(y_{t-1}, y_t) \equiv\ & (\delta(y_{t-1}=1)\delta(y_t=1), \delta(y_{t-1}=1)\delta(y_t=2), \ldots, \delta(y_{t-1}=1)\delta(y_t=C),\\ & \delta(y_{t-1}=2)\delta(y_t=1), \delta(y_{t-1}=2)\delta(y_t=2), \ldots, \delta(y_{t-1}=2)\delta(y_t=C),\\ & \vdots\\ & \delta(y_{t-1}=C)\delta(y_t=1), \delta(y_{t-1}=C)\delta(y_t=2), \ldots, \delta(y_{t-1}=C)\delta(y_t=C))^\top\end{aligned}$$

  - これは C^2 次元のベクトル

$$\varphi_t^{\Sigma\Sigma}(y_{t-1}, y_t)$$

x_{t-1}	x_t
y_{t-1}	y_t

2種類の特徴ベクトルを組みあわせ、配列全体で足し合わせたものが配列全体の特徴ベクトルになります

- 以上の2種類の特徴ベクトル $\varphi_t^{\mathcal{X}\Sigma}(x_t, y_t)$と$\varphi_t^{\Sigma\Sigma}(y_{t-1}, y_t)$を並べた：

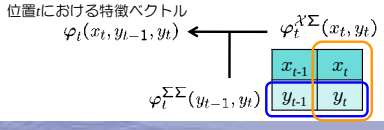
$$\varphi_t(x_t, y_{t-1}, y_t) \equiv (\varphi_t^{\mathcal{X}\Sigma}(x_t, y_t)^\top, \varphi_t^{\Sigma\Sigma}(y_{t-1}, y_t)^\top)^\top$$

を、位置tにおける特徴ベクトルとする

- そして、これを全ての位置について足し合わせたもの:

$$\varphi(x, y) \equiv \sum_{t=1}^{T} \varphi_t(x_t, y_{t-1}, y_t)$$

が、配列全体に対する特徴ベクトルの定義となる



CRFでは、配列上で隣り合う2つのラベルの組み合わせ特徴によって、ラベル間の依存関係を効率的に捉えています

- CRFの特徴ベクトルは全体として$CD + C^2$の長さを持つ
- この特徴ベクトルの構成において特に重要なのは、隣り合う2つの位置のラベルの関係を考慮した特徴 $\varphi_t^{\Sigma\Sigma}(y_{t-1}, y_t)$である
  - 各位置の入出力の特徴ベクトル $\varphi_t^{\mathcal{X}\Sigma}(x_t, y_t)$だけでは、位置間の関係を取り入れていないため、各位置で独立に分類問題を解くのと変わりがなくなってしまう
  - 一方で、長さTの出力ラベル列に含まれるT個のラベルの組み合わせを素朴に捉えてしまうと前述のようにC^T個のパラメータベクトル（DC^T次元）を考えることになってしまう
- CRFの特徴ベクトルでは、ラベル間の関係を、配列上で隣り合う2つのラベルの関係にのみ限定して考えることによって、ラベル間の依存関係を効率的に捉えている

条件付き確率場における予測

条件付き確率場（CRF）の予測は、素朴に行おうとすると指数時間かかってしまうので、動的計画法を用います

- 予測：入力列 x が与えられたときに、最も高い条件付き確率P($y|x$)を与える $y = \hat{y}$ を見つけること：

$$\hat{y} \equiv \underset{y \in \Sigma^T}{\operatorname{argmax}} \frac{\exp(\omega^\top \varphi(x, y))}{\sum_{c \in \Sigma^T} \exp(\omega^\top \varphi(x, c))}$$

- 可能な出力ラベル列 y の候補が C^T 個あるため、これを素朴に実行することは計算時間の面で現実的ではない。
- しかし、特徴の定義が隣り合う2つの変数に対してのみ定義されていることを利用すると、動的計画法によってこれを効率的に（具体的には$O(TC)$の計算量で）行うことができる

予測の式は単純化できます

- 予測の式： $\hat{y} \equiv \underset{y \in \Sigma^T}{\operatorname{argmax}} \frac{\exp(\omega^\top \varphi(x, y))}{\sum_{c \in \Sigma^T} \exp(\omega^\top \varphi(x, c))}$
- 分母は y に依存しない（すべてのyについて和をとっている）ため y についての最大化を行うにあたって分母は無視しても良い
- また、分子のexpは単調増加関数であるため、exp(・)の中身を最大化するyがこれを最大化することがわかる。
- 従って、最大化問題は 以下のように書いて差支えない

$$\hat{y} = \underset{y \in \Sigma^T}{\operatorname{argmax}} \ \omega^\top \varphi(x, y)$$

- さらに条件付き確率場の特徴ベクトルの定義により：

$$\hat{y} = \underset{y \in \Sigma^T}{\operatorname{argmax}} \sum_{t=1}^{T} \omega^\top \varphi_t(x_t, y_{t-1}, y_t)$$

条件付き確率場の事後確率最大化

事後確率の最大化（もしくは最尤推定）によってCRFを学習します

- N個の入出力配列の組 $\{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$が訓練データ集合として与えられたとする。
  - i 番目の訓練データの長さを$T^{(i)}$とする。
- 訓練データ集合に対する事後確率の対数を取ったものの和は：

$$
\begin{aligned}
L(\omega) &\equiv \sum_{i=1}^{N} \log P(y^{(i)}|x^{(i)};\omega) - \lambda \parallel \omega \parallel_2^2 \\
&= \sum_{i=1}^{N} \log \frac{\exp(\omega^\top \varphi(x^{(i)}, y^{(i)}))}{\sum_{y \in \Sigma^{T^{(i)}}} \exp(\omega^\top \varphi(x^{(i)}, y))} - \lambda \parallel \omega \parallel_2^2 \\
&= \sum_{i=1}^{N} \omega^\top \varphi(x^{(i)}, y^{(i)}) - \sum_{i=1}^{N} \log \sum_{y \in \Sigma^{T^{(i)}}} \exp(\omega^\top \varphi(x^{(i)}, y)) - \lambda \parallel \omega \parallel_2^2
\end{aligned}
$$

最急勾配法によって目的関数を最大化します

- これを最大化するために、最急勾配法を用いることにする
- 更新式（現在のパラメータ$\boldsymbol{\omega}^{(t)}$から新たなパラメータ$\boldsymbol{\omega}^{(t+1)}$へ）は：

$$\boldsymbol{\omega}^{(t+1)} \leftarrow \boldsymbol{\omega}^{(t)} + \eta^{(t)} \nabla(\boldsymbol{\omega}^{(t)})$$

  - $\eta^{(t)}>0$ は更新の幅を決定する学習率と呼ばれるパラメータ
  - $\nabla(\boldsymbol{\omega}^{(t)})$は現在のパラメータ$\boldsymbol{\omega} = \boldsymbol{\omega}^{(t)}$における目的関数$L(\boldsymbol{\omega})$の勾配：

$$\nabla(\mathrm{w}^{(t)}) \equiv \left. \frac{\partial L(\omega)}{\partial \omega} \right|_{\omega=\omega^{(t)}}$$

勾配を求める際に、計算の難しい箇所が2箇所あります

- パラメータの更新を行うためには$\nabla(\boldsymbol{\omega}^{(t)})$を計算する必要がある
- そのために目的関数$L(\boldsymbol{\omega})$を$\boldsymbol{\omega}$について偏微分したものを計算する

$$\frac{\partial L(\omega)}{\partial \omega} = \sum_{i=1}^{N} \varphi(x^{(i)}, y^{(i)}) - \sum_{i=1}^{N} \frac{\sum_{y \in \Sigma^{T^{(i)}}} \exp(\omega^\top \varphi(x^{(i)}, y)) \varphi(x^{(i)}, y)}{\sum_{y \in \Sigma^{T^{(i)}}} \exp(\omega^\top \varphi(x^{(i)}, y))} - 2\lambda\omega$$

- ここで問題となってくるのが2つの部分の計算：

$$\sum_{y \in \Sigma^{T^{(i)}}} \exp(\omega^\top \varphi(x^{(i)}, y)) \qquad \sum_{y \in \Sigma^{T^{(i)}}} \exp(\omega^\top \varphi(x^{(i)}, y)) \varphi(x^{(i)}, y)$$

  - 可能な全ての$y \in \Sigma^{T(i)}$についての和を含むため、これを素朴に計算するのは、計算量の面で現実的ではない
- 予測のときと同じく動的計画法を利用できる