統計的機械学習 2017 年夏学期 シケプリ

EEIC2017 浅井明里

2017 年 7 月 21 日

1 はじめに

2017 年度春学期より開講の鶴岡先生による「統計的機械学習」のシケプリです。基本的には授業スライドのまとめ、省略されている用語の解説や勾配計算、口頭で説明された内容をまとめています。数式についてはできるだけチェックしていますが、もしミスなどあれば教えていただけると嬉しいです。途中で「参考資料に」とか書いてある部分は主にスライドで省略された式変形とかカバーしようとしたのですが今週の試験で体力を持って行かれたのでやりかけです…そのうちひっそり更新するかもしれません… ただ、授業中でも（参考）とつけれていたところが多いので、おそらくカバーしなくても大丈夫なはず…

2 抑えた方が良さそうなこと

今年開講の講義なので何が出るのかはよくわかりません…おそらく基本的なモデルや語句などを抑えておけば単位はくると思います。学習の際におそらくみておいて方がいいことは、

- あるモデルについてパラメータ (SVM における C 等) を変化させた時のバイアスバリアンスの変化。
- 類似したモデルや手法の比較、どのようなデータに対してどのモデルを使うべきか。（あれば）モデルの長所、短所は何か。
- 授業中で手計算のやり方を確認したもの (線形回帰のパラメータ推定、ロジスティック回帰の予測率の計算、リッジ、Lasso 回帰のパラメータ推定)

3 統計的機械学習の基本

3.1 教師あり学習と教師なし学習 (第一講)

統計的機械学習で扱われる問題は教師あり学習」「教師なし学習」「強化学習」に大別できます。統計的機械学習の授業では教師あり学習、教師なし学習についての扱います。

教師あり学習」と「教師なし学習

教師あり学習 (supervised Learning)

入力から出力を予測するモデルを学習、正解が与えられている。

教師なし学習 (unsupervised learning)

予測の対象となる出力 (正解) が与えられない、データの構造や関係を解析

授業で紹介されたアメリカ東海岸エリアの労働者の賃金データである Wage[*1]は それぞれの労働者の age, sex, year, race 等の情報と賃金のデータが与えられているため、このデータを用いてモデルを正しく学習させることで年齢や性別等の情報が与えられた時にその人の賃金を予測するモデルを作ることができます。 このように入力、出力が共に与えられ、入力から出力を予測する回帰、分類等の問題を教師あり学習と言います。

一方、遺伝子の発現データである NCI60 Data[*2]は 特に出力が与えられているわけではなく、データを様々な軸でクラスタリングします。こういった正解ラベルを与えることなく、データからある一定数のクラスに分類する問題は教師なし学習に分類されます。

3.2 記法について

まず p 個の特徴量 (年齢、性別など予測を行うために参照できる情報) を持つ n 個入力データについては、i 番目の観測データの j 番目の変数を x_{ij} とすると、以下のようなベクトル X で表します。

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix}$$

ここで i 番目の観測データを表すベクトルを $x_i^T = (x_{i1}\ x_{i2}\ \dots x_{ip})$ とすると [*3] 以下のように記述することも可能です。

$$\boldsymbol{X} = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{pmatrix}$$

一方、予測の対象は何らかのスカラー値で表されます（分類問題の際にはカテゴリの名前等になることもあります)。

$$\boldsymbol{y} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

学習データはこの入力 X 及び出力 y のペアで以下のように表されます。

$$\{(x_1, y_1), (x_2, y_2), \dots (x_n, y_n)\}$$

以後、入力、出力等のデータについては基本的にはこの記法を用います。

[*1] https://vincentarelbundock.github.io/Rdatasets/doc/ISLR/Wage.html

[*2] https://www.rdocumentation.org/packages/made4/versions/1.46.0/topics/NCI60

[*3] 蛇足かもしれませんが T は行列の転置を意味しています

3.3 統計的学習とは (第 3 講)

統計的学習とは、入力 $X = (X_1, X_2, \ldots, X_p)$ と出力 Y の間にある関係が存在すると過程、入力データ X から Y を推定できることを目指します。

入力と出力

入力

統計的機械学習において、予測行うために与えるデータ。Advertisement データセットにおける TV, Radio, newspaper など。入力変数、予測変数、特徴量（素性）などと呼ばれる。

出力

学習されたデータに入力データを与えた時に返される予測結果。出力変数、応答変数、従属変数などど呼ばれる。

統計的機械学習では、以上の入力 $X = (X_1, X_2, \ldots, X_p)$ と出力 Y の間にある何らかのシステマティックな関係 f を予測することを目指す。

$$Y = f(X) + \varepsilon$$

またここで ε は確率的な誤差 (error) を示しており平均はゼロになります。

入力から出力を予測する際に用いるモデルとして、パラメトリックな手法とノンパラメトリックな手法があります。

パラメトリックな手法とノンパラメトリックな手法

パラメトリックな手法

f の関数形について例えば線形モデルの $f(X) = \beta_0 + \beta_1 X_1$ 等の過程をあらかじめ置いて置き、学習を通して仮定された関数の係数 $\beta_0, \beta_1 \ldots \beta_p$ を求める。

ノンパラメトリックな手法

f の関数形について事前に明示的な仮定を置かない手法。

パラメトリックな手法は f の推定問題を $p+1$ 個のパラメータの値を求めるというシンプルなものに帰着できるという利点がある一方、 仮定されたモデルが真の f を表現できるかどうかは事前にはわからず、表現力の高いモデルを仮定すると過学習 [*4] が起こってしまう恐れがあります。ノンパラメトリックな手法は複雑な f を表現できる可能性があり、また高精度の推定のためにはパラメトリックモデルよりも大量の観測データが必要になるケースが多いです。

x と y の間に存在する関数 f を推定する目的としては、予測及び推論という二つの目的があります。

目的 1： 予測 (prediction)

Y が容易に得られない場合に、X から Y を予測し推定する。 例えば患者に初めての薬を処方する際、その患者が服用して副作用が起こる可能性があるかどうか事前に予測する必要がる。ここで過去の副作用発生事例から 副作用を起こした患者、起こしていない患者の年齢、持病、性別等のデータ X より、この新しい患者がある薬に対して重篤な副作用が起きるリスクいう出力 Y を事前に推定することができる。このように予測が目的である時は推定結果が正確である限り、$\hat{f}$ はブラックボックス化して構わな

[*4] 学習データの中のノイズにも適応し過ぎてしまい、新しい入力に対して正しい予測をできなくなってしまうこと

い。予測誤差には削減可能なものと削減不可能なものがあり、これについては参考資料で補足します。

目的 2： **推論** (inference)

$X_1,\ X_2, \ldots, X_p$ が変化した時に Y がどのように変わるかを知りたいときに推論を行う。これはどの入力変数が Y に関係しているかを知り活用することを目指している。 例えば Advertising データセットにおける学習では、知りたいのは「何らかのシステマティックな関係により売り上げがいくつかになる」 ではなく、例えば TV 広告を増やした時、新聞広告を増やした時でどのくらい売り上げが変わるのかを知るため、 Y と入力変数 X の間にある線形もしくは非線形な関係を知り、どの X が Y に影響を与えているかそのウェイトの大きさをみる。

目的を予測にするか推論にするかにより選択されるべき適切なモデルは異なってきます。 例えば推論を目的とするならば、推定した関数の中身の解釈が比較的容易な線形モデルや、どのように条件を分岐させているかを可視化できる決定木モデルなどがどの 独立変数がどう従属変数に影響を与えているんか容易に確認が可能です。一方線形モデルでは複雑非線形な関係に対応できず、決定機は入力データに左右されるため、 常に高い精度が得られるわけでありません。予測のみを目的とするならば、対象が複雑な現象であっても高い予測精度が得られる可能性がある非線形モデルが適しているかもしれませんが、 非線形モデルは中身の解釈が難しく、推論には使いにくい傾向にあります [*5]。

予測精度と解釈性のトレードオフ

- 表現力の低いモデルでは複雑な形の f をうまく近似できない一方、 Y と $X_1, X_2 \ldots X_p$ の関係がわかりやすい。(予測精度は低いが解釈性が高い)
- 表現力の高いモデルでは多様な f に対応が可能ですが、Y と $X_1, X_2 \ldots X_p$ の関係がわかりづらい。(予測精度は高いが解釈性が低い)

4 回帰と分類 (第 3 講)

4.1 変数の種類

統計的機械学習ではある一つ以上の入力変数 X から、出力 y を推定する f を求めことを目的としますが、入力変数 X には連続した実数値となる量的変数と、 離散的なカテゴリ（ラベル）になる質的変数があります。

[*5] 現在でも広く使用されている機械学習モデルの一つに SVM(サポートヴェクターマシン) というものがあり、これは 非線形な関係にも対応でき一般的なタスクにおいて線形モデルより高い精度を発揮することも多いのですが、線形モデルと比較するとどの入力変数が出力とどの程度影響を与えているか直接知ることはできません。 また機械学習ではないですが、今話題の深層学習もモデル自体をブラックボックス化してしまい、人間が直接的にどうそれぞれの入力変数が影響を与えているのか知るのが難しいと言われています

量的変数と質的変数

量的変数

年齢、身長、収入、株価などの連続的な実数値をとる変数。そのまま入力変数として使用することができる。

質的変数

性別、学歴、Yes/No、カテゴリの種類など、K 個の異なるクラス (カテゴリ) からどれかの値をとる変数。そのままでは入力変数として用いることができず、 後述するダミー変数の使用などにより予測モデルに組み込む。

教師あり学習は回帰問題と分類問題に大別することができ、回帰問題 (regression Problem) では予測変数が金融商品の価格等の量的変数となる一方、 分類問題 (classification problem) では予測対象が債務不履行に陥ったか否かなどの質的変数になります。

補足ですが、質的変数を入力変数として用いた予測モデルを作る際には「女性/男性」などの情報はそのままでは数理モデルに組み込むことができないのでそれぞれの質的変数（質的変数に含まれるカテゴリ) ごとにダミー変数を作成することが一般的です。 例えば「女性/男性」「大人（18 歳以上）/子供 (18 歳未満)」等ある質的変数について完全に 2 値で分けられるときには以下のように「女性であれば 1/男性であれば 2」等のように変数を作ります。

$$x_i^{gender} = \begin{cases} 1 & \text{(if ith person is a female)} \\ 0 & \text{(if ith person is a male)} \end{cases}$$

ではこのように「true/false」の 2 値で分類できない場合はどうするのでしょうか。例えば ethnicity(民族) について、とりあえず Caucasian = 2, African American = 1, or Asian = 0 とか 適当に数字を振ってしまったら [*6] もともと順序や序列の存在しないデータに恣意的に大小関係を加えることになり、モデルの予測を歪めてしまいかねません。 そこでこういった複数の値を取りうる質的変数についてはその全てに対してダミー変数を作り、「Asian ならば 1、そうでければ 0」といった風に数字を割り当てます。

$$x_i^{asian} = \begin{cases} 1 & \text{(if ith person is an Asian)} \\ 0 & \text{(if ith person is not an Asian)} \end{cases}$$

4.2 モデルの精度

モデルの精度の測り方には様々な方法がありますが、回帰問題の場合、平均二乗誤差 (mean squared error, MSE) を、分類問題の場合 error rate を用いることが一般的です。

[*6] どっかの団体に怒られそうな例だ...

平均二乗誤差及び error rate

平均二乗誤差 (MSE)

平均二乗誤差とは n 個の 学習データから構築されたモデルの予測した結果と実際のラベルと差を二乗し、平均を撮ったものであり、以下の式で表される。

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

誤り率 (error rate)

error rate は n 個の学習データに対する予測のうち、誤った予測を行ってしまった数をデータの総数 n で割ったもの。

$$\text{error rate} = \frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

ここで $I(y_i \neq \hat{y}_i)$ は $y_i \neq \hat{y}_i$ ならば 1 を、そうでなければ 0（すなわち正解していれば 0）をとる関数。

ちなみにこの式で表されるのは学習データでの誤差 (training MSE) ですが、本当に最小化したいのは未知データでの誤差（test MSE）です。気をつけなくてはいけないのは、特に過学習をしている場合などは学習データにおける MSE がいくら高くても Test MSE が必ずしも少なくなっているとは限りません。

$$\mathrm{Ave}((y_0 - \hat{f}(x_0))^2)$$

error rate も同様に上の式で得られるのは学習データでの error rate に過ぎず以下の式で表される Test error rate を下げる必要があります。

$$\mathrm{Ave}(I(y_i \neq \hat{y}_i)^2)$$

またモデルの精度を測る指標としてバリアンス (variance) とバイアス (bias) があり、テスト MSE はこれらバリアンス及びバイアスで表すことができます。

バリアンスとバイアス

バリアンス

学習データの違いによってどれだけモデルによって予測される $\hat{f}$ の値が変わるかを示す。未知のデータに対してどのくらい正しく予測できているかを測る。

バイアス

真のラベル f と予測した値 $\hat{f}$ がどれだけずれているかを示す。学習データに対して、どのくらい正しく予測できているかを測る。

平滑化スプライン等の表現力の高いモデルは、学習モデルに対しては 高い精度で予測できる一方、学習データにオーバーフィットしやすく新しい未知データに対して正しく予測できない可能性が高まるためバリアンスが高い傾向にあります。 一方線形回帰などの表現力のあまり高くないシンプルなモデルは、このようなオーバーフィットをしにくく、バリアンスは小さくなる傾向にあるが、学習データに対してフィットしきれずバイアスが大きくなります。このようにバイアスとバリアンスの間にあるトレードオフの関係を The bias-variance trade-off と言います。

The bias-variance trade-off

テスト MSE は以下の 3 つの項の和に分解でき、バイアスとバリアンスにはトレードオフ (バイアスが小さい時はバイアスが大きくなり、バリアンスが小さい時はバリアンスが大きくなる) の関係にあることが確認できる。

$$\text{TestMSE} = E[(y_0 - \hat{f}(x_0))^2] = \text{Var}(\hat{f}(x_0)) + [\text{Bias}(\hat{\text{f}}(\text{x}_0))]^2 + \text{Var}(\varepsilon)$$

- $\text{Var}(\hat{f}(x_0))$: $\hat{f}(x_0)$ のバリアンス (分散)$\text{Var}(\hat{f}(x_0)) = E[\hat{f}(x_0)^2] - E[\hat{f}(x_0)]^2$
- $[\text{Bias}(\hat{\text{f}}(\text{x}_0))]^2$: $\hat{f}(x_0)$ のバイアスの二乗 $E[\hat{f}(x_0) - f(x_0)]$
- データにもともと含まれる ε の分散

このバイアスバリアンストレードオフはよく出てくるので、モデルやモデル内でのパラメータを変えた時、バリアンスやバイアスがどう変化するのか、正しく答えられるようにしましょう。

4.3 ベイズ分類器

ある入力 X が条件付き確率が最大になるようなクラスにあるデータ X_i を分類する方法です。ベイズ最適決定とも呼ばれます。

ベイズ分類器

以下の条件つき確率が最大になるクラスを選び、期待誤り率を最小になることを目指した。

$$P(Y = j|X = x_0)$$

ベイズ分類器の誤り率はベイズ誤り率 (Bayes error rate) と呼ばれ、以下の式で表される。

$$1 - E[\text{maxP}_\text{r}(Y = j|X)]$$

これは回帰問題での削減不可能な誤差に相当している。

しかし実際のところ、条件付き確率を正しく知ることはできないので、これはあくまでも理論上の分類器になります。 ベイズ分類器で分類された結果に基づいてデータの間に引かれた境界線のことをベイズ決定境界といいます。ベイズ分類器とは頃なる方法で分類を行うことを目指すのが次の K 最近傍法です。

4.4 K 最近傍法 (Knearest meighbors method)

K 最近傍法は入力変数からそれぞれの入力が属するクラスを予測する分類問題をとくモデルの一つであり、自分に最も近い上位 K 個の観測データの属するカテゴリで多数決を行い、自身の最近傍 K 個の観測データの中で最も多く現れたラベルをその入力データのラベルとします。これを定式化すると以下のようになります。

K 最近傍法

クラスの条件付き確率を以下のように推定し、この確率が最も大きくなったクラスに分類する。

$$P_r(Y = j|X = x_0) = \frac{1}{K}\sum_{i \in N_0} I(y_i = j)$$

仮にある事例 x_0 を「グループ 1」「グループ 2」の二つに分類したい場合に $K = 3$ の K 最近傍法を行なった

場合、仮にある事例に最も近い上位 3 個 (日本語がおかしいのは仕様です) のうち、二つが「グループ 1」、一つが「グループ 2」ならばこの事例に対する条件付き確率はそれぞれ、

$$P_r(Y = \text{グループ } 1|X = x_0) = \frac{1}{3} \times 2 = \frac{2}{3}$$

$$P_r(Y = \text{グループ } 2|X = x_0) = \frac{1}{3} \times 1 = \frac{1}{3}$$

となり、この事例 x_0 の予測ラベルは $P_r(Y = \text{グループ } 1|X = x_0) > P_r(Y = \text{グループ } 2|X = x_0)$ より、「グループ 1」になります。

この K の値を小さく、例えば $K = 1$ などにすると、それぞれ自身に最も近い観測データのみ考慮するため、決定境界はより入り組んだ、表現力の高いモデルになります。この場合学習データに対しては高い精度を誇るものの、 テストデータでは謝り率が高くなることがありバリアンスが大きなモデルになります。 また $K = 1000$ にしてしまうと、実際にはラベルを予測したい事例よりかけはなれた事例のデータを予測に考慮することになりノイズを与えてしまいかねません。 このようにモデルの表現力が高すぎても低すぎてもテストエラーが高くなってしまうため、K の値には注意が必要で後ほど紹介する交差検証により適切な K を選択します。KNN で求められたラベルに基づいてデータ間に引かれた境界線が KNN 決定境界になります。

図 1 の例で $K = 1$ の場合と $K = 100$ の場合の KNN 決定境界を見ると、実際に K の数が小さいほど境界線が入り組んでいる、つまり表現力が高くなっていることがわかります。 この例では $K = 1$ のケースの方がわずかに $K = 100$ よりも高いテストエラー率になってはいますが、図 2 の K の数に対する学習誤り率とテスト誤り率の推移を見ると、モデルの表現力が高いからといってテスト誤り率は必ずしも下がるわけではないことが確認できます。

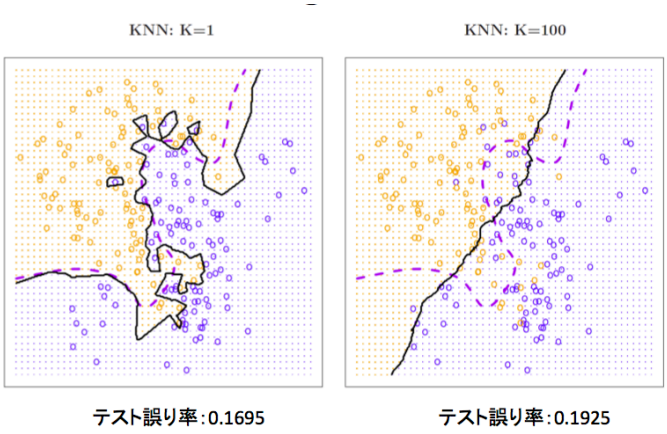

図 1　K=1 及び K=100 の時の KNN 決定境界（黒太線）

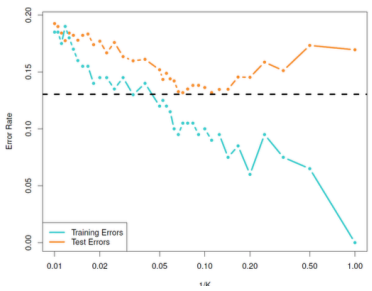

図 2　K の数に対する学習誤り率とテスト誤り率の推移

4.5　線形回帰 (Linear regression)

4.5.1　線形単回帰

線形単回帰とは、X と Y の関係は線形ですと仮定して、一つの予測変数 X によって量的応答変数 Y を予測します。

線形単回帰

X と Y の関係は以下の式のような線形で表されると仮定する。

$$Y \approx \beta_0 + \beta_1 X$$

学習データによって適切なパラメータを予測し、$\hat{y} = \hat{\beta_0} + \hat{\beta_1} x$ をより元の学習データの結果と近づけることを目指す。学習データを $(x_1, y_1),\ (x_2, y_2),\ \ldots,\ (x_n, y_n)$ とし、予測のずれを残差 $e_i = y_i - \hat{y_i}$ とすると、線形単回帰の目的関数は以下のように表される。

$$\text{RSS(残差平方和)} = e_1^2 + e_2^2 + e_3^2 + \ldots e_n^2 = (y_1 - \hat{\beta_0} - \hat{\beta_1} x_1)^2 + (y_2 - \hat{\beta_0} - \hat{\beta_1} x_2)^2 \ldots$$

ここで RSS は学習データ X に対して推定されたパラメータ $\hat{\beta_0}$及び$\hat{\beta_1}$ を用いて線形のモデルで予測した $\hat{y}$ の値が、実際の y との距離を二乗したものの和であり、 これを最小化することによりよりずれの少ないモデルを構築することができます。線形単回帰においてパラメータを推定するときは残差平方和が最小になるようにパラメータを推定すればいいことになります。 目的関数 RSS を最小化するには、RSS を $\hat{\beta_0}$及び$\hat{\beta_1}$ で微分した時に、それが 0 なるような $\hat{\beta_0}$及び$\hat{\beta_1}$ を求めれば良いため

$$\frac{\partial \text{RSS}}{\partial \hat{\beta_0}} = 0,\ \frac{\partial \text{RSS}}{\partial \hat{\beta_1}} = 0$$

を満たす $\hat{\beta_0}$及び$\hat{\beta_1}$ を求めると、以下のようになります。(詳細な変形などは参考資料に)

$$\hat{\beta_1} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{n}(x_i - \overline{x})^2},\ \hat{\beta_0} = \overline{y} - \hat{\beta_1}\overline{x}$$

$$\overline{x} = \frac{1}{n}\sum_{i-1}^{n} x_i,\ \overline{y} = \frac{1}{n}\sum_{i-1}^{n} y_i$$

パラメータの導出については授業で実際に手計算させていたのでもしかしたら聞かれるかもしれません。導出の仕方を押さえておくといいと思います。 この結果求められた二つのパラメータ $\hat{\beta_0}, \hat{\beta_1}$ がどの程度正確なのかは母回帰直線を推定することにより求めることができます。

4.5.2 残差 RSS に関する性質

また、RSS を最小にする β_0, β_1 の導出より、以下の二つの性質が求められます。(導出は参考資料に)

$$\sum_{i=1}^{n} e_i = 0 \tag{1}$$

$$\sum_{i=1}^{n} e_i x_i = 0 \tag{2}$$

式 (1) は残差の合計が 0 であることを、式 (2) は残差 e_i と入力 x はベクトルとして直行であることを意味しています [*7]。

[*7] ベクトル a,b の内積がゼロであるとき二つのベクトルは直行しているため。

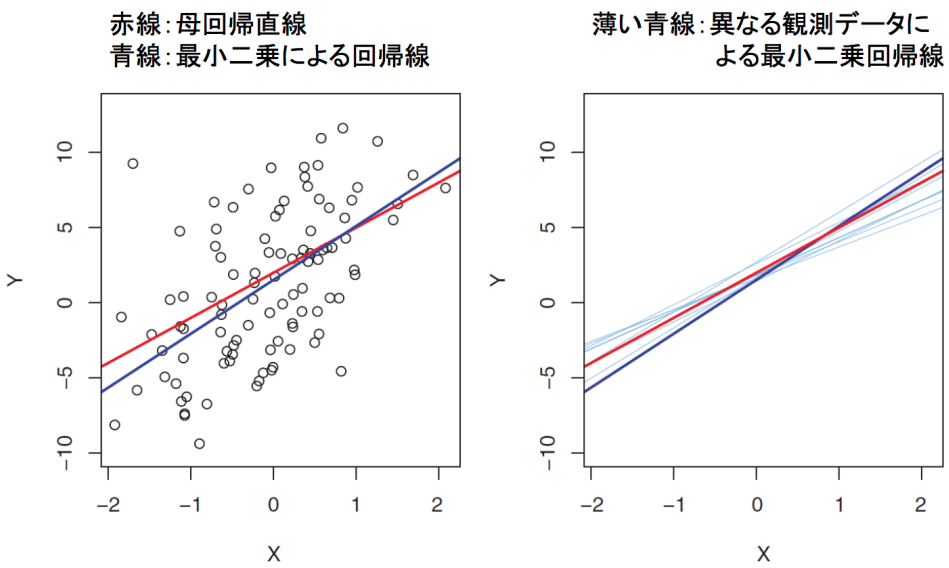

図 3　線形単回帰誤差

4.5.3　線形単回帰の信頼性

線形単回帰においては、母集団において X, Y の間に線形の関係性があると仮定しデータよりそのパラメータ $\hat{\beta}_0, \hat{\beta}_1$ を予測しており、本来的に母集団の β_0, β_1 を予測することはできません。 そこでデータより得られたパラメータがその程度信頼できるのかを確認します。 母集団での X と Y の関係を $Y = \beta_0 + \beta_1 X + \varepsilon$ とすると、ε は母集団における統計的な誤差なので、母集団での回帰直線は $Y = \beta_0 + \beta_1 X$ であらわされます。図 3 では、$Y = 2 + 3X + \varepsilon$ という線形の関係を元に生成した人工データに対する線形単回帰モデルに基づくパラメータ推定の結果を示しており、実際の母回帰直線と最小二乗による回帰直線の結果にややずれがあることがわかります。 また与えられた入力データにおける誤差項により異なる観測データに対する最小二乗回帰直線にはややズレがあることが図 4 右図よりわかります。

求められたパラメータより信頼区間を求めるにはどうしたらいいのでしょうか。そのためにまず各パラメータの標準誤差を求める必要があります。 個々の観測データの誤差は独立で分散が σ^2 と仮定するとパラメータの標準誤差は

$$SE(\hat{\beta}_0^2) = \sigma^2 \left[\frac{1}{n} + \frac{\overline{x}^2}{\sum_{i=1}^{n}(x_i - \overline{x})^2} \right]$$

$$SE(\hat{\beta}_1^2) = \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

ここで σ は未知であるため、RSS(残差平方和) を用いて以下のように観測データから推定され RSE(redidual standard error) と呼ばれます。

$$\mathrm{RSE} = \sigma \approx \sqrt{\mathrm{RSS}/(n-2)}$$

SE の導出、及び次の信頼区間の導出については参考資料にあります。パラメータの信頼区間について、95% の確率でその区間にパラメータの真の値が含まれるための近似的な信頼区間はそれぞれ以下で表されます。

(これについても参考資料に)

$$\hat{\beta}_0 \pm 2\mathrm{SE}(\hat{\beta}_0),\ \hat{\beta}_1 \pm 2\mathrm{SE}(\hat{\beta}_1)$$

4.5.4 線形単回帰モデルがどの程度データにフィットしているか評価する

モデルがどの程度データにフィットしているか評価する指標として、RSD 及び決定係数 R^2 があります。決定係数とは回帰分析によって求められた目的変数の予測値が実際の目的変数の値とどのくらい一致しているかを表している指標です。厳密な定義としては「回帰分析をした結果が目的変数のばらつき分散をどれくらい説明しているか」ということを表しており、 もちろんですが 1 に近いほどデータを正しく説明できていることになります。

RSE(residual standarderror)

誤差 ε の標準偏差の推定量のこと。以下の式で求めることができる。

$$RSE = \sqrt{\frac{1}{n-2}\mathrm{RSS}} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

次に紹介する決定係数も回帰モデルによって実データをどれくらい説明できているか、つまり回帰分析の精度を表す指標です。 こちらも 1 に近いほどモデルがデータをよく表現できていることを意味します。

決定係数 R^2

決定係数とは応答変数 Y の分散を X による回帰方程式で減らした割合。以下の式で定義される。

$$R^2 = \frac{\mathrm{TSS} - \mathrm{RSS}}{\mathrm{TSS}} = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}}$$

ここで TSS(Total sum of square) とはもともとのデータが持っている分散の総和であり、以下の式で定義される。

$$\mathrm{TSS} = \sum_{i=1}^{n}(y_i - \overline{y})^2$$

式からわかるように、第 2 項が小さい時は、すなわち RSS $\ll TSS$ となり元々のデータの持つ変動よりも回帰により予測された値と元データの間の変動の方が小さく、 モデルがうまくデータを説明できていることを意味します。

4.6 多重線形回帰 (第 5 講)

4.6.1 多重線形回帰とは

実際のデータにおいては、一つの入力変数だけでなく、複数の変数を用いて予測をしたいというシチュエーションも多く存在します。複数の単回帰モデルを個別に構築し、組み合わせるだけでは 予測変数同士の関係を考慮することができず、またどう個別のモデルを組み合わせるべきかは明らかになっていません。そこで多重線形回帰は、複数の予測変数を使う単一のモデルで回帰を行うことを目指しています。

多重線形回帰

以下のように複数の X と Y の関係を以下のようなモデルで回帰する。

$$Y \approx \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$$

β_j は他の予測変数を固定して、X_j を 1 単位増やしたときに増える Y 量の量を表している。学習データによって適切なパラメータを予測し、$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$ を予測とし、学習データの残差平方和を以下のように表す。

$$\mathrm{RSS} = \sum_{i=1}^{n} (y_1 - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \hat{\beta}_2 x_{i2} \ldots - \hat{\beta}_p x_{ip})^2$$

要は線形単回帰では一つの変数について、その入力変数が一単位増えたときに増加する Y の量 (パラメータ β_1) がモデルの傾きになっていましたが、 多重線形回帰ではある p 個の入力変数 $X_1, X_2, \ldots X_p$ それぞれにパラメータ β_i を求めてモデルを構築しただけです。多重線形回帰についても単線系回帰の場合と同様に、それぞれのパラメータで目的関数を微分したものが 0 となるよう計算することで求めることができます。 勾配情報を使って再急降下法、 準ニュートン法などを行うことによって計算できます。また、多重線形回帰の結果求められるのは回帰「平面」になります。(次元増えるからそれはそう)

4.6.2 多重線形回帰と F 分布

多重線形回帰においても、線形回帰の時と同様、応答変数と予測変数の間に関係があるかどうかの検定を行うことができます。 ここで基礎統計等で出てきた F 分布について復習します。

F 分布

U_1：自由度 d_1のカイ二乗分布に従う確率変数, U_2：自由度 d_2 のカイ二乗分布に従う確率変数であり、U_1と U_2 は独立であるとした時、F 分布とは確率変数

$$X = \frac{U_1/d_1}{U_2/d_2}$$

が従う確率分布。

多重線形回帰において、応答変数と予測変数の間に関係があるかどうかの検定は以下のように行います。

- 帰無仮説：$H_0 = \beta_{p-q+1} = \beta_{p-q+2} = \cdots = 0$
- 対立仮説：少なくとも一つの $\beta_j (j > p-q)$ がゼロではない。
- 帰無仮説が正しい場合全てのパラメータ β_j が 0 である、すなわち予測変数と関係がない）時。以下の F 統計量は F 分布 $(q, n-p-1)$ に従う。

  $$F = \frac{(RSS_0 - RSS)/p}{RSS/(n-p-1)}$$

  RSS_0 : 最後の q 個の予測変数を使わないモデルで得られた RSS

4.6.3 どの予測変数が重要なのか

多重線形回帰を行う際に、全ての変数を元に Y を求めるても逆にノイズが混じってしまったり計算のコストがかかってしまったりと有効ではありません。 また全ての予測変数の考えうる組み合わせ全てを試して精度を調べるのもコストが大きいです。そこで多重線形回帰では事前にどの予測変数が重要ですかの見当をつけモデルを構築するのが一般的です [*8]。変数選択法には以下のような手法が一般的です。

変数選択法の種類

変数増加法

予測変数を全く入れない状態からモデルの構築を始め、ある一つの変数だけを今あるモデルに加えたとき、その RSS が最も小さい（結果と関係の大きい）予測変数を追加するということを繰り返す。

変数減少法

全ての予測変数を使うモデルから出発し、p 値が最も大きい（つまり結果と関係の小さい）予測変数を削除するということを繰り返す。

変数増減法

変数増加法と変数削減法を組み合わせたもの。

RSS は上の項で紹介した残差平方和でありこれが小さいほどモデルはデータから正しく予測できていると言えます。元々のモデルにある変数を一個加えた複数のモデルの中で最も RSS が小さなモデルの変数を新たに加えていくことにより、 少しずつモデルの精度を上げていくのが変数増加法です。また、変数減少法の p 値とは帰無仮説 H_0 のもとで，統計量（確率変数）T が、データから実際に計算した統計量の値 T_0 よりも「極端」な値を取る確率を意味し、 実用的な意味でいうと p 値が有意水準（5%）とかより小さければ帰無仮説 ($H_0 : \beta_{p-q+1} = \beta_{p-q+2} = \ldots \beta_p = 0$) を棄却します。p 値が大きい時、帰無仮説は棄却されずある入力変数 β_1 にかかる係数が 0 である可能性が高くなるので、この予測変数が結果との関係が小さく、無視してよいとなります（あんまりここの説明自信ない)。

4.6.4 多重線形回帰モデルがどの程度データにフィットしているか評価する

線形単回帰モデルと同様に、多重線形回帰においてモデルがどのくらいデータに適合しているかは決定係数及び RSE によって評価することができます。決定係数は線形単回帰と同様、

$$R^2 = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}}$$

で表すことができますが、多重線形回帰の場合は 1+p だけ自由度が減少するため、以下の式で定義されます。

$$\mathrm{RSE} = \sqrt{\frac{1}{n-1-p}\mathrm{RSS}}$$

4.6.5 多重線形回帰モデルにおける変数同士の相互作用

線形単回帰モデルの時は入力変数は一つだけだったため、単一の入力変数と予測変数の間の関係のみに着目するだけですみました。 しかし多重線形回帰については、各予測変数間の応答変数に対する効果は必ずしも

[*8] Azu ○ eML 勉強会で関係なさそう、 もしくは id などノイズを与えてそうな Housing データの入力変数を除いて見たりした人もいたのではないでしょうか。要はあんな感じです。

独立しているとは言えず（例えば Advertisement データにおける TV と radio という二つの変数はそれぞれ独立して売り上げに貢献しているのではなく、その二つを一緒に行うことによってより宣伝効果をあげているかもしれません)、多数の変数間の相互作用を考える必要があります。 それぞれの変数が独立だとした時のモデルは以下のように関連していると考えられる変数の積を新たな変数としてモデルに追加することで実現できます。

$$Y = \beta_0 + \beta_1 X_1 + \beta_3 X_1 X_2$$

ただ、このように複数の変数の相互作用を考慮してモデルを構築しようとすると、特に入力の次元が大きくなった時には特徴空間の拡大及び計算量の増加や過学習に繋がるので注意が必要です。これについてはのちのサポートベクターマシンの説明でも少し言及します。

4.6.6 多重線形回帰で起こりうる問題

多重線形回帰では線形回帰モデルで解決できない非線形な関係に対応することを目指していました。しかし多重線形モデルでは以下のような問題も発生してしまいます。

誤差項の相関

時系列データ (time-series data) などでは誤差項に 相関があることが多い。

誤差項の分散

Y の値が大きいところでは誤差項の分散も大きくなってしまう。そこで Y を予測するのではなく、$\log(Y)$ を予測するようにするとこれを抑えることができる。

外れ値

計測エラーなどによる外れ値が推論に大きな影響を与える可能性があり、こういった外れ値は回帰直線には影響を与えにくいが決定係数などに大きな影響を与えうる。そこで絶対値が 3 を超えるような値は外れ値の可能性が高いため、 除外をした方が良い。

てこ比

よくわからない [*9]

共線性 (collinearity)

予測変数の間に強い関連がある場合間に強い相関がある場合、そのままではパラメータ推定の信頼性が落ちてしまう。 そのため相関行列により 2 つの予測変数の間の関係を検出する必要があり、仮に二つの予測変数が強い相関を持っていた場合、これらを一つに統合する。

多重共線性

三つ以上の変数の間に共線性がある場合、VIF によって検出が可能である。

$$VIF(\hat{\beta}_j) = \frac{1}{1 - R^2_{X_j|X_{-j}}}$$

4.6.7 多項式回帰 (polynomial regression)

これまでは全ての変数について、ある入力変数 X^p の予測変数 Y にに与える影響は線形的であるとしてモデルを考えてきました。 しかし実際には入力変数と予測変数の間に二次関数的な関係性が存在する可能性もあります。こういった非線形な関係を考慮するためには多項式回帰を行います。 変数同士の相互作用を考慮する場合と同様、X_1^2 等の予測変数を追加し、特徴空間を拡張します。

[*9] http://nlp.dse.ibaraki.ac.jp/ shinnou/zemi2006/mva/mimami1.pdf

4.7 パラメトリックモデルとノンパラメトリックモデルの比較

線形単回帰や多重単回帰のようにデータから目的関数を最小化するようなパラメータを推定し、そのパラメータに基づいてある入力 X に対して予測を行うモデルをパラメトリック、 KNN のようにパラメータを推定するのではなく逐次元のデータを読み込み、予測するモデルをノンパラメトリックモデルと言います [*10]。与えられたデータに非線形性が強いなど、ノンパラメトリックな手法の方が有利に働くケースも存在しないわけではないのですが、 パラメトリックなモデルとノンパラメトリックなモデルの精度が同じくらいなのではあればパラメトリックなモデルの方が良いとされることが多いです。理由としては

- ノンパラメトリック手法では一度学習したデータ [*11] を保持し続ける必要があり、一度学習すればあとはパラメータのみ保持して予測を行うことができるパラメトリックなモデルと比較すると、取り回しのしにくい。
- ノンパラメトリックなモデルの方が次元の呪い（入力変数の数が増える、すなわち予測する空間の次元が大きくなるほど適切な予測ができにくくなる）の影響を強く受けてしまう。
- ノンパラメトリックなモデルの方がより多くのデータを必要とする場合が多い。

ことが挙げられます。

5 分類

5.1 分類問題とは (第 7 講)

分類問題とは応答変数が質的変数である予測問題であり、クラスを予測する問題とも言えます。 例えば患者の症状から疾患を予測する、メールがスパムかどうかを判定するなどはこのような分類問題の例です。 分類問題と回帰問題は別の問題として考える必要があります。これは特に 3 クラス以上の分類などにおいて、線形回帰により分類を行おうと考えるとカテゴリに恣意的に順序をつけ、スコアの大きさに応じて振り分けようとすると、そもそも明確にカテゴリ同士に順序がないはずのものについても 恣意的な仮定を置く必要が出てきてしまい正しい予測をできない可能性があるためです。

5.2 ロジスティック回帰による分類問題

以下では回帰の中でも一般的な、ロジスティックモデルについて説明します。

[*10] パラメトリックモデルとノンパラメトリックモデル http://www.snap-tck.com/room04/c01/stat/stat99/stat0103.pdf

[*11] 実際はデータそのものではなく Condense とかで Weight などを保持し簡略化できるのですが、やはりノンパラメトリックモデルよりは多くのデータを保管する必要があります。

	Coefficient	Std. error	Z-static	p-value
Intercept	−10.6513	0.3612	−29.5	< 0.0001
balance	0.0055	0.0002	24.9	< 0.0001

図 4　ロジスティック回帰により求められた値

ロジスティックモデル

ロジスティックモデルとは、以下のロジスティック関数を用いて確率値を計算するモデル。

$$p(x) = \frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}} = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$$

オッズは確率論で確率を示す値を意味しており、ロジットはオッズの対数をとった値。

オッズ

$$\frac{p(x)}{1-p(x)} = e^{\beta_0+\beta_1 x}$$

ロジット

$$\log \frac{p(x)}{1-p(x)} = \log e^{\beta_0+\beta_1 x} = \beta_0 + \beta_1 x$$

ロジスティック回帰におけるパラメータは確率モデルのパラメータ推定に広く使われる、最尤推定により決定されます。

最尤推定

パラメータの尤度 (学習データが得られる確率) を計算する以下のような関数を尤度関数といい、これを最大にするような β_0, β_1 の値を計算する。

$$l(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y'_i=0} (1-p(x'_i))$$

ロジスティック回帰で求められたパラメータの値から予測値を求める方法も授業では紹介されてたのでほぼスライドと同じですが手順を振り返ります。 以下のようにある人の講座残から債務不履行 (default) する確率を予測するモデルのパラメータをロジスティック推定により求めた結果が図 6 の場合、切片 $(\hat{\beta}_0)$ が-10.6513、balance について 1 単位増加させた時の予測変数の増加分 $\hat{\beta}_1$ が 0.0055 となるので、例えば $1000 の人が default する確率は

$$\hat{p}(X) = \frac{\exp(\hat{\beta}_0+\hat{\beta}_1 X)}{1+\exp(\hat{\beta}_0+\hat{\beta}_1 X)} = \frac{\exp(-10.6513+0.0055\times 1000)}{1+\exp(-10.6513+0.0055\times 1000)} = 0.00576$$

よって balance が $1000 の人がデフォルトする確率は 0.5% と極めて低いことがわかる。質的変数（あるカテゴリに対してそれぞれダミー変数を設定したもの）を用いて予想を行う場合は以下のような学生であるかどうかというダミー変数に対して予測されたパラメータを用いて予測を行う場合、学生である人は $x = 1$、学生で

	Coefficient	Std. error	Z-static	p-value
Intercept	−3.5041	0.0707	−49.55	< 0.0001
student [Yes]	0.4049	0.1150	3.52	0.0004

図 5　ロジスティック回帰により求められた値 (定性情報)

ない人は $x = 0$ とすれば良いので、学生のデフォルトする確率は

$$\hat{p}(X) = \frac{\exp(\hat{\beta}_0 + \hat{\beta}_1 X)}{1 + \exp(\hat{\beta}_0 + \hat{\beta}_1 X)} = \frac{\exp(-3.5041 + 0.4049 \times 1)}{1 + \exp(-3.5041 + 0.4049 \times 1)} = 0.00431$$

上記の式で x=0 とした時の確率が 0.0292 より、学生の方がややデフォルトする確率が高いとこのモデルからは予測できます [*12]。またロジスティック回帰において応答変数（予測するラベル、default するかどうか、患者の病気の症状など）が多数の値を取る場合は、ソフトマックス回帰により、最も確率が高くなるラベルを選択肢します。 この定義からもわかりますが、ソフトマックス回帰の結果得られる、各クラスに対する予測確率の合計は必ず 1 になります。

ソフトマックス回帰

ソフトマックス関数は、n 次元の実数ベクトル $x = (x_1 \dots x_n)$ を受け取って以下で定義される n 次元実数ベクトル $y = (y_1 \dots y_n)$ を返す関数である。

$$y_i = \frac{e^{x_i}}{e^{x_1} + e^{x_2} \dots + e^{x_n}}$$

ソフトマックス回帰は

$$\text{softmax}(x)_k = P(Y = k|X) = \frac{e^{x_k}}{\sum_{k=1}^{n} e^{x_i}}$$

ここで $e^{x_k} = \exp(\sum_p \beta_{kp} X_p)$ を表し、入力が X であるときに予測変数が $k \in K$ である確率を表す。

5.3　複数の予測変数を用いたロジスティック回帰

線形単回帰と同様に、ロジスティック回帰も複数の予測変数を取り入れたモデルに拡張することができます。複数の予測変数を利用したロジスティック回帰を多重ロジスティック回帰といいます。

多重ロジスティック回帰 (multiple logistic regression)

多数の予測変数を用いたロジスティック回帰モデルを多重ロジスティック回帰という。

$$\text{ロジット} = \frac{p(X)}{1 - p(x)} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \dots \beta_p X_p$$

$$p(X) = \frac{\exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2 \dots \beta_p X_p)}{1 + \exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2 \dots \beta_p X_p)}$$

[*12] 実際には後述するように必ずしも学生がデフォルトしやすいわけではなく交絡によりそう見えているだけだったりします。

	Coefficient	Std. error	Z-static	p-value
Intercept	−10.8690	0.4923	−22.08	< 0.0001
balance	0.0057	0.0002	24.74	< 0.0001
income	0.0030	0.0082	0.37	0.7115
student [Yes]	−0.6468	0.2362	−2.74	0.0062

図 6　多重ロジスティック回帰

多数ロジスティック回帰についても、計算方法は入力変数を一つにしたロジスティック回帰と基本的には同じです。先ほどと同様に、balance, income, student を使ってある人が default するかどうかを予測してみます。balance が $1,500、income が　$40,000 の学生が default する確率は、上の表で切片 $\beta_0 = -10.8690$、balance にかかるパラメータ $\beta_1 = 0.0057$、income にかかるパラメータ $\beta_2 = 0.0030$、student にかかるパラメータ $\beta_3 = -0.6468$ より、

$$\hat{p}(X) = \frac{\exp(-10.8690 + 0.0057 \times 1500 + 0.0030 \times 40,000 + (-0.6468) \times 1)}{\exp(-10.8690 + 0.0057 \times 1500 + 0.0030 \times 40,000 + (-0.6468) \times 1)} = 0.058$$

ちなみに式からわかるかもしれませんが、このモデルにおいて学生かどうかを表すダミー変数にかかる $\beta_3 < 0$ となっているため、 同じ balance, income であれば学生ではない方が default する確率は低いと予測されます。これが先ほど軽く補足した交絡 (confounding) による効果であり、 交絡とは統計モデルの中の従属変数と独立変数の両方に（肯定的または否定的に）相関する外部変数が存在することを意味します。例えば「飲酒者と非飲酒者では飲酒者の肺癌発生率が高くなる」というのがデータからは確かに観測されるのですが、 これは直接的に飲酒が肺がんの要因になっているわけではなく [*13]、非飲酒者と比較して飲酒者における喫煙者の割合が高いために、 結果的にデータから「飲酒者の方が肺がんになりやすい」という傾向がでてしまうことになります。今回の問題では実際には学生だから default を起こしやすいのではなく、学生の方が非学と比べ income が少ない、credit card balance が大きい等の要因によりデフォルト率が高くなっています。

5.4　線形判別分析（第 7, 8 講）

ロジスティック回帰のように直接 $\Pr(Y|X = x)$ をモデル化するのではなくある入力変数 X が与えられた時にそれがある応答変数 Y となる確率を予測する別のモデルに「線形判別分析 (LDA)」があります。

[*13] もしかしたら何か関係あるのかもしれないけどそういうのは医学部の人に聞いてください

線形判別分析 (linear discriminant analysis)

線形判別分析とは、$\Pr(Y|X=x)$ を正規分布でモデル化し、ベイズの定理によって $\Pr(Y|X=x)$ を計算するモデル。K をクラスの数、π_k をラス k の事前分布、$f_k(x)$ をクラス k のデータの密度関数とすると、$\Pr(Y|X=x)$ は以下のように表される。

$$\begin{aligned}\Pr(Y|X=x) &= \frac{\Pr(Y=k)\Pr(X=x|Y=k)}{\Pr(X=x)} \\ &\frac{\Pr(Y=k)\Pr(X=x|Y=k)}{\sum_{l=1}^{K}\Pr(X=x,Y=l)} \\ &= \frac{\Pr(Y=k)\Pr(X=x|Y=k)}{\sum_{l=1}^{K}\Pr(Y=l)\Pr(X=x|Y=l)}\end{aligned} \tag{3}$$

$$p_k(x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K}\pi_l f_l(x)} \tag{4}$$

$f_k(x)$ が正規分布だと仮定すると、$f_k(x)$ は以下の式で表される。

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{1}{2\sigma_k^2}(x-\mu_{x_k})^2\right) \tag{5}$$

分散が全てのクラスで等しい場合、ある入力 x が与えられた時、それが k である確率は式 (5) を式 (4) に代入することで求めることができ、

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{1}{2\sigma_k^2}(x-\mu_{x_k})^2\right)}{\sum_{l=1}^{K}\pi_l \frac{1}{\sqrt{2\pi}\sigma_l}\exp\left(-\frac{1}{2\sigma_l^2}(x-\mu_{x_l})^2\right)} \tag{6}$$

式 (6) より、$p_k(x)$ は $\pi_k \frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{1}{2\sigma_k^2}(x-\mu_{x_k})^2\right)$ に比例するため、 これが最も大きくなるクラスに入力データを分類すればいいことがわかります。等分散性より $\sigma_k = \sigma$ を仮定し式 (6) の右辺分母について対数をとると、

$$\log(\text{式 (6) 右辺分母}) = \log\pi_k + \log 1 - \log\sqrt{2\pi}\sigma + \left(-\frac{1}{2\sigma^2}(x-\mu_k)^2\right)$$

となり、予測に関わるのは π_k を含んだ項のみとなるため、

$$\delta_k(x) = \log\pi_k + x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2}$$

として、この $\delta_k(x)$ を最大にするクラス k に事例 x を分類すると考えることができます。すなわち K 個のクラスに対して μ_k及びσ_k^2 がわかれば分離境界 x を求めることができます。
例えば $K=2, \pi_1=\pi_2,$ 等分散性より$\sigma_1=\sigma_2=\sigma$ の場合を考えると、分離境界において $\sigma_1(x)=\sigma_2(x)$ が成立するため、

$$x\frac{\mu_1}{\sigma^2} - \frac{\mu_1^2}{2\sigma^2} + \log\pi_1 = x\frac{\mu_2}{\sigma^2} - \frac{\mu_2^2}{2\sigma^2} + \log\pi_2$$

$$x\frac{\mu_1-\mu_2}{\sigma^2} = \frac{\mu_1^2-\mu_2^2}{2\sigma^2}$$

$$x = \frac{\mu_1+\mu_2}{2}$$

実際には μ_k及びσ_k^2 は未知なので、以下のようにデータから推定し判別関数を求めます。

$\hat{\mu}_k$及び$\hat{\sigma}_k^2$ の推定による線形判別分析法による分類

n を学習データの全事例、n_k を クラス k に属する事例の数とすると、

$$\hat{\mu}_k = \frac{1}{n_k}\sum_{i:y_i=k}^{k} x_i,\ \hat{\sigma}_k^2 = \frac{1}{n-K}\sum_{k=1}^{K}\sum_{i:y_i=k}(x_i - \hat{\mu}_k)^2$$

また π_k も未知である場合は以下のようにデータから推定できる。

$$\hat{pi}_k = \frac{n_k}{n}$$

線形判別分析ではこれらの推定値を用いて、以下の判別関数 $\hat{\sigma}_k$ が最も大きくなるクラスに分類する。

$$\hat{\sigma}_k = x\frac{\hat{\mu_k}}{\hat{\sigma}_k^2} - \frac{\hat{\mu_k}^2}{2\hat{\sigma}_k^2} + \log(\hat{\pi}_k)$$

予測変数が 2 以上の場合は密度関数が異なるので判別関数が異なるのですが基本的な求め方が前述の方法と同じなのと、あの式を入力するのがめんどくさいのでスライドで確認していただけると嬉しいです。

5.5 二次判別分析

線形判別分析では各グループは多変量正規分布をしており、かつ等分散性 (全てのグループが同じ共分散行列を持つ) を持つことを仮定していました。二次判別分析ではクラスごとに異なる共分散行列を想定し、判別を行います。

二次判別分析 (quadratic discriminant analysis, QDA)

クラスごとに異なる共分散行列を想定し、以下の判別関数の与えるスコアが最も大きくなるクラス k に分類する。

$$\delta_k(x) = -\frac{1}{2}x^T\sum_{k}^{-1}x + x^t\sum_{k}^{-1}\mu_k - \frac{1}{2}\mu_k^T\sum_{k}^{-1}\mu_k - \frac{1}{2}\log|\sum_{k}| + \log\pi_k$$

5.6 LDA と QDA の比較

QDA ではクラスごとに分散共分散行列のパラメータを求める必要があり、計算のコストが大きくなることまた学習パラメータが小さいときなどは学中データにフィットしすぎて過学習を引き起こす可能性があります (バリアンスが大きく、バイアスが小さい)。 一方 LDA はバリアンスは小さいものの、そもそも共分散行列が各クラスで同じという過程をおいているため、そもそもこの過程が成立していなかった場合、データにフィットせずバイアスが大きくなる恐れがあります。

5.7 分類手法の比較

では分類問題においてはこれまで取り扱った「ロジスティック回帰」「線形判別分析」「虹判別分析」「K 最近傍法」のうち、どの判別器を用いるのが良いのでしょうか。まずロジスティック回帰及び LDA では共に境界面が直線（もしくは超平面）になるため、そもそも線形分離のできないデータには十分に対応できません。仮にデータが線形分離が可能な場合はデータの分布が各クラスそれぞれ正規分布である時は、これを仮定にお

いてモデルを構築する LDA の方がやや精度が高くなります。 KNN、QDA は共に非線形な決定境界を実現可能ですが、KNN はより複雑な境界線を構築することができる一方、前述の通り多くのデータを必要としかつ結果からどのパラメータが重要なのか等がわかりにくいため、学習データが少ない時は QDA の方が良いというべきかもしれません。

分類手法の比較についてのまとめ

LDA

真の決定境界が線形かつデータの分布が各クラスそれぞれ正規分布に従っている。また分散共分散行列が等しい。

ロジスティック回帰

真の決定境界が線形かつデータの分布が各クラスそれぞれ正規分布に従っていない。

QDA

真の決定境界が非線形かつデータの分布が各クラスそれぞれ正規分布に従っている。分散共分散行列が等しくない。

KNN

決定境界の非線形性か特に゛強い場合もしくは非線形性があり、データが正規分布に従っていない時。

5.8 正解率以外の評価指標

分類問題においては、単なる正解率 (accuracy) 以外にもいくつか評価指標があり、クラスによりデータの数が大きく異なる時 [*14] など、問題によっては正解率以外の評価指標がモデルの評価基準として適していることがあります。授業で紹介されていたのは sensitivity と specificity 及び ROC 曲線でありこれらの評価指標は主に二値分類で用いられます。

5.8.1 感度 (sensitivity) と特異度 (specificity)

それぞれ、実際のラベルが Yes であるもの、No であるものに対して正しく予測ができているのかを測る指標です。 図 7 のようにデータを実際のラベルと予測結果で分割したものを混同行列 (confusion matrix) といいます。

[*14] 例えば婚活アプリのチャットのログからどういったやりとりをしている二人が実際に付き合っているのかを予測したい時、全体の中で実際にカップル成立したログのデータが 10% 以下しかなかったとすると、 「全ての組み合わせが結局付き合うに至らなかった」と予測するようなでたらめなモデルでも正解率を見ると 9 割以上になってしまいます。しかしここで知りたいのはデータの中で少数派の「どういったやりとりをした二人が実際に付き合ったのか」であり、 この少数派をきちんと予測できるモデルを評価できる指標を用いる必要があります。

		True default status		
		No	Yes	Total
Predicted default status	No	True Neg. (TN)	False Neg. (FN)	TN+FN
	Yes	False Pos. (FP)	True Pos. (TP)	TP+FP
	Total	TN+FP	TP+FN	TP+TN+FP+FN

図 7　混同行列

sensitivity と specificity

感度 (sensitivity)

実際のラベルが「Yes」であるものに対してきちんとモデルが「Yes」と予測できているかどうか。(Yes であるものを正しく検出できた割合)

$$\text{sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

特異度 (specificity)

実際のラベルが「No」であるものに対してきちんとモデルが「No」と予測できているかどうか。(No であるものを正しく検出できた割合)

$$\text{sensitivity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

混同行列のそれぞれのブロックに対応したカテゴリ名がややわかりにくいかもしれませんが、「予測結果=実際の分類」であれば True が頭につき、「予測結果が Yes」であれば Positive が次に、「予測結果が No」であれば Negative が後につきます。 つまり「本来のラベルは No だが Yes だと予測されてしまった」ケースについては、実際のラベルと予測が異なるので頭に「False」が予測結果が Yes なので後ろに Positive がつき「False Positive(偽陰性)」になります [*15]。実際に感度、特異度を求めたい場合はここにそれぞれの値を代入するだけでいいです。

では実際にこのような指標を用いるのはどんな場面なのでしょうか。「Yes」が全体の中で閉める割合の低いデータについてはモデルにより「Yes」が想起される確率が低くなります。 これをそのまま放置すると「全てを No で分類するモデル」になりかねないので、あえて閾値 0.5 ではなく 0.2 等に下げることで本来の二値分類では「No」に分類されていた事例も「Yes」に分類されやすくなり感度を高くすることができます。 ただこの場合当然ですがエラー率は上がってしまうため、目的に応じてこういった指標を使い分ける必要があります。sensitivity、specificity 以外に図 9 のようにもいくつか指標はありますが、授業で詳しくは触れていないので特に大丈夫だと思います。

5.8.2　ROC 曲線 (receiveroperatingcharacteristicscurve)

上のように特にデータの中で偏りがある場合などに閾値をあえて 0.5 ではなくそれ以下もしくは以上にすることにより目的とする結果をえられるモデルを作りたい時があります。 これも交差検証で適切な閾値の結果

[*15] 個人的に忘れにくい覚え方は「結婚相手を探す時に False Positive」だけは引っかかってはいけない」（本来ならうまくいくはずのない相手と誤って結婚してしまい、そういう結果になること）です。

名称	定義	別名
False Positive rate	FP/N	Type I error, 1 – Specificity
True Positive rate	TP/P	1 – Type II error, power, sensitivity, recall
Positive Predictive value	TP/P*	Precision, 1 – false discovery proportion
Negative Predictive value	TN/N*	

図 8　様々な評価指標

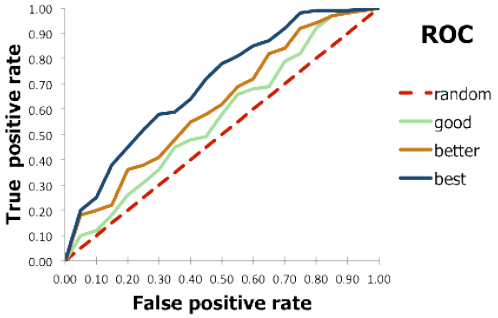

図 9　ROC 曲線

を探す必要があるのですが、その場合はこれまでのように正解率ではなく AUC(Area Under the Curve) という指標を用います。 まず、False Positive Rate(= 1 - Specificity) と True Positive Rate(= Sensitivity) をそれぞれ xy 平面にプロットします。この結果えられた曲線を ROC 曲線といい、 この ROC 曲線の下の部分の面積を AUC といいます。この AUC 値が 1 に近いほど判別能が高いことを示しています。図 10 からもわかるように ROC 曲線の下の面積が大きいものほど優れたモデルになります。閾値を変えて予測を行った際に AUC が最もよくなる閾値を選択します。

6　リサンプリング法 (第 9 講)

リサンプリング法とは、手持ちの限られたデータをある固定された割合で分割し、訓練データとテストデータに分けて学習するのではなく、 ランダムサンプリングと学習によるパラメータ推定を複数回行い、データの分割の仕方による予測モデルのばらつきなどを少なくすることを目指しています。 試行回数が増える分計算コスト画像化するため、かつてはこれがネックになっていましたが、計算機の能力向上によりこういったコストの問題は小さくなっています。 以下で代表的なリサンプリング手法について紹介します。

6.1　検証セットによる手法

データセットをランダムに「学習用データセット (training set)」と「検証用データセット (validation set)」の二つにランダムに分割し、 学習用データでモデルのパラメータを推定し、検証用データで精度を評価するという、最もベーシックな方法です。シンプルな一方で、 データをどう分割するかによって精度の推定値やパラメータの予測値は大きく変わってしまうこと、 またデータ自体が少ない時にデータ全体の数割から半分ほどのデータを学習に用いることができないため、精度も本来達成可能な精度より低めに推定される可能性があります。

6.2　交差検証

検証セットによる精度測定法を補うため、交差検証法はデータセットの全てのデータを用い、また繰り返し学習したパラメータの平均値を取ることでデータの分割の仕方によるばらつきの影響を排除することを目指しています。

Leave-One-Out Cross Valdation(LOO 交差検証)

N 個の学習データ (x_1, y_1), $(x_2, y_2) \dots$ (x_n, y_n) のうち、1 つをのぞいた $N-1$ 個で学習を行い、毎回残された 1 個のデータで性能を評価、 これを全ての事例に対して行いその MSE の平均をとることで、最終的な評価指標の推定値とする。

$$\mathrm{MSE}_1 = (\hat{y}_1 - y_1)^2$$

$$\dots$$

$$\mathrm{MSE}_\mathrm{n} = (\hat{y}_n - y_n)^2$$

$$\mathrm{CV(n)} = \frac{1}{n}\sum_{i=1}^{n} MSE_i$$

LOO 交差検証には以下のようなメリットデメリットが存在します。

LOO 交差検証のメリット

学習には毎回 n-1 個のデータが使用され n 個全てのデータを用いた際とほぼ同じ性能のモデルを作ることができる。 また分割方法の違いによる性能評価のばらつきもない。

LOO 交差検証のデメリット

n 回の学習が必要になり、かつ学習データと検証データに一定割合で分割する際と比べ学習に 使用するデータの数も増えるため、データの数が多いほど、学習に必要なコストが大きく増加する。

このように、LOO 交差検証は学習データと検証データに分割する際のデメリットを克服できる一方、 計算コストが大きく増加してしまいます。そこで、一つだけを性能評価用に残し、n 回学習を行うのではなく、データをある数 k に分割し、一つのグループ以外のデータで学習、残された一つのデータで性能検証を行うことを繰り返すことで 計算量を抑えつつも性能の向上を目指すのが次の k 分割交差検証です。

6.3 K 分割交差検証

k-fold Cross Valdation(k 分割交差検証)

データを k 個のグループに分割し一つを検証データ、残りの $k+1$ 個を学習データとして MSE を計算し、これを k 回繰り返して平均を取り、 この平均をテスト MSE として推定する。一般的に $k = 5, 10$ 等が用いられる。

$$CV(k) = \frac{1}{k}\sum_{i=1}^{k} \mathrm{MSE}_i$$

LOO 交差検証では一般に n 回の学習を行う必要がありましたが、k 分割時交差検証では学習回数はたかだか k 回ですむ一方、LOO 交差検証と同様、 k 回の学習を通して全ての入力データをみることができます。また LOO 及び k 分割交差検証の間には以下のような Bias-variance trade-off が存在します。

- K 分割交差検証の場合、学習に使えるデータの数は LOO 比べ少なくなるので、バイアスが大きくなる可能性がある。
- LOO 交差検証には学習に使うデータが毎回ほとんど同じなので出力間に強い相関があり、バリアンス

が大きい。

6.4 ブートストラップ

また、こういったデータを分割し、繰り返し学習左折以外に、ブートストラップという方法もあります。

ブートストラップ

標本からの復元抽出によるランダムサンプリングを繰り返すことで行う統計的推論手法。確率モデルのパラメータの信頼区間などを簡単に求めることができる。

復元抽出とは、抽出を行う際に一度抽出したサンプルが再び抽出の対象となりうる方法のことです。例えば受験数学でよくでてくる「赤い玉 3 個、白い玉 4 個入った袋から中身を見ずに一個取り出し色を確認して戻す作業を 3 回行った時、 全て赤い玉であった確率は」などは復元抽出の例であり、「一度出した玉は元に戻さない」であればこれは非復元抽出になります。具体的なブートストラップ法の利用としては、

1. 手持ちのデータセットからランダムサンプリングによって新たなデータセットを生成。
2. この抽出されたデータを元にモデルのパラメータを推定する。
3. これを多数回繰り返す。
4. 得られたパラメータの分布より信頼区間などを計算する。

という手順になります。

7 正則化（第 10 講）

7.1 パラメータ推定法と縮小推定法

最小二乗法、最尤推定法などのパラメータ推定法では、モデルがデータにフィットするようにパラメータを最適化します。 一方リッジ回帰、Lasso 回帰は縮小推定に分類され、各パラメータの絶対値が大きくならないようにします。これらは正則化とも呼ばれています。 最小二乗法による線形回帰パラメータ推定は、以下の RSS を最小化にするパラメーラ $\beta_0, \beta_1 ..., \beta_p$ を決定しますが、 これは学習データへのフィティングしか考慮していないため、過学習してしまいやすく未知のデータに対して適切に予測できない恐れがあります。 そこでリッジ回帰や Lasso 回帰などの縮小推定法により過学習を回避し、未知データに対しても頑健なモデルを構築します。

7.2 リッジ回帰

リッジ回帰

リッジ回帰の目的関数は以下で表される。

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\underline{\lambda\sum_{j=1}^{p}\beta_j^2}$$

下線で示され項がパラメータの絶対値が大きくなることに対するペナルティ（L2 正則化項）となる。
λ : チューニングパラメータ。

L2 正則化項の λ を大きくするほど、各パラメータの値 $\beta_0, \beta_1 ..., \beta_p$ はゼロに近づいていくことになります。またリッジ回帰においては、値の大きな予測変数が大きな影響力を持つようになるため、予測変数のスケールが性能に大きな影響を与えてしまいます。 そのため前処理として以下のような式で予測変数の標準化を行う必要があります。

$$\widetilde{x_{ij}}=\frac{x_i j}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i j-\overline{x_j})^2}}$$

λ を大きくするとバリアンスは小さくなり、バイアスは大きくなります。これは λ が小さければ、リッジ回帰の目的関数はパラメータ推定法における目的関数と近く、つまり学習データへの 過学習によりバリアンス (入力データによる出力結果のばらつき) が大きくなる傾向にある一方、λ が大きければモデルは過学習を抑え新しい学習データに対しても頑健になるためバリアンスは小さるためです。 その一方で、λ 項を大きくすると元の学習データに対するフィットは少なくなるためバイアスが大きくなります。ここでも以下の式で表されるバイアスバリアンスのトレードオフ関係が存在します。バイアス、バリアンスが共に大きくなりすぎず、MSE が最小になるちょうどいい λ を交差検証等を行って選択することにが大事です。

7.3 Lasso 回帰

Lasso 回帰は、正則化項をパラメータの二乗ではなく、絶対値とした目的関数を最小化することによりパラメータを求める方法です。

Lasso 回帰

Lasso 回帰の目的関数は以下で表される。

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\underline{\lambda\sum_{j=1}^{p}|\beta_j|}$$

下線で示され項がパラメータの絶対値が大きくなることに対するペナルティ（L1 正則化項）となる。
λ : チューニングパラメータ。

Lasso 回帰の目的関数はリッジ回帰と比べ、最後の正則化項が異なっているだけに見えますが、この式から得られるパラメータの性質は大きく異なります。 リッジ回帰ではパラメータ β が 0 に近づくにつれ、L2 正則化項はそれぞれのパラメータを二乗したものの総和なので、より急速に 0 に近づきます。一方 Lasso 回帰で

- Credit データセット

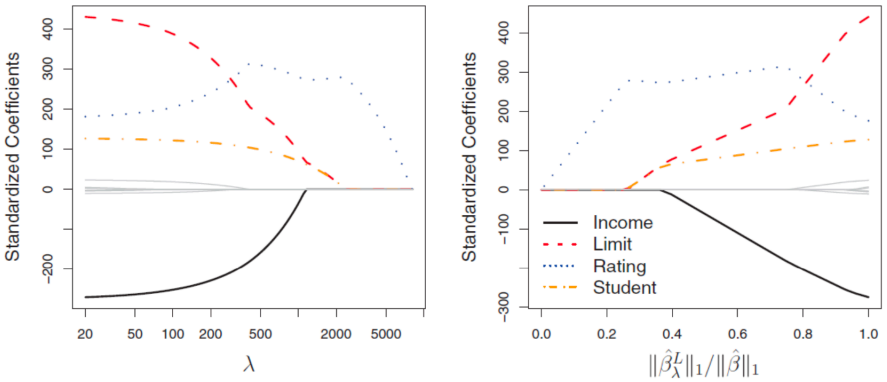

図 10　Credit データセットに対して Lasso 回帰

は、正則化項は $beta$ の絶対値の和になるため、 L1 正則化項は β に比例して小さくなり残り続けます。結果として Lasso 回帰では λ の値を大きくするにつれ、多くのパラメータが急速に 0 に近づくため、より重要なパラメータ以外の 枝葉のパラメータが自動的に除かれることになります。特に多くのパラメータを保持したくない場合（例えば携帯端末に保存するモデルを作るときなど）はこちらの方が適していると言えます。

図 10 は前述の Credit データセットに対して Lasso 回帰を行い、λ を大きくした際のパラメータと出力の Coefficient を示しており、λ が大きくなるにつれて、 多くのパラメータが急速に 0 に近づいていることがわかります。（ここら辺は授業でやった手計算で自分で確認してみるとわかりやすいと思います。参考資料で補足します。）

7.4　リッジ回帰、Lasso 回帰の比較

リッジ回帰、Lasso 回帰どちらが適切なのかはデータの性質によって異なります。仮に入力データの次元が大きく、余分なパラメータを落としてしまいたい場合は Lasso が適切かもしれません。 一方で、Lasso 回帰の最適化 (パラメータ推定) の際には、L1 正則化の方が絶対値項を持っているために勾配の計算が難しくなります。リッジ回帰の正則化項は二乗の総和でしかないので勾配計算が簡単であり、Lasson 回帰の方が最適化が難しくなってしまうという短所があります。 特にデータサイズが大きい時は、Lasso 回帰での勾配の計算はかなり面倒になります。またどちらの場合でも、適切な λ の選択については交差検証によってもっとも Test MSE が小さくなる λ を選択する必要があります。

7.5　高次元データでの学習 (第 11 講)

入力データによっては、予測変数（特徴量）p が学習事例の個数 n よりも大きくなることが大きくなることがあります。例えば、血圧の予測では、特徴量として年齢、性別、BMI の他に遺伝子情報（SNPs）を特徴量として入れる場合、遺伝子情報は個々人によって少しずつ異なり、それぞれにダミー変数を振り分けていくと予測変数の数は膨大に増加する（$p \approx 500000$）。仮に患者データが 200 人分ほど ($n \approx 200$) であれば $p \gg n$ となってしまいます。 他にもスパムメールの分類で単語を特徴量として用いる時、すでにスパム、スパムでな

いのラベルがついたメールが 1000 件前後しかない時に、そこに含まれている全ての単語を特徴量として用いれば [*16]、 特徴量の数はデータの数をたやすく上回ってしまいます。$p \geq n$ のとき、必要ではないノイズな予測変数（例えばスパムメール内でも実際には分類に大きく関係しない単語は多々存在しています）でもフィットしてしまうため、 過学習に陥る可能性が高くなります。

これに対する対処法としては、正則化や変数選択などによりモデルの柔軟性を下げ、必要ではない予測変数を減らします。しかしこの方法にはいくつか注意点があり、例えば多重共線性 [*17] により、変数選択によって選ばれた変数以外にも有用な変数が存在する可能性があります。 また学習データで得られる RSS や決定係数は、実際のデータでも完全に当てはまるとはいえず、モデル の性能の指標としては無意味であることも注意が必要です。

8 決定木 (decision tree)

木構造によって回帰や分類のための条件を表すモデルであり、目的とする問題に応じて回帰木（regression tree)、分類木 (classification tree) があります。

8.1 回帰木

回帰木とは、ある数値（連続値）の推定のルールをツリーで表現したものです。Hitters データセット [*18] は、選手の在籍年数（Years）と打数 (Hits) より Log(Salary) を予測するデータセットです。回帰木を用いて Salary を予測する時、図 9 のようにまず「Years < 4.5」なら選手の Log(Salary) は 5.11 に決定され、 そうでない、かつ「Hits < 117.5」ならば 6.00、「Years < 4.5」かつ「Hits ≥ 117.5」ならば 6.74 となる。

回帰木は予測変数の空間を K 個のリージョンに分割することにより、分岐条件を求めます。回帰木の大まかの作り方は以下のようになります。

1. 予測変数の空間を J 個の領域 $R_1, R_2, \ldots, R_j$ に区分けをする。
2. 予測変数の値が領域 R_j に含まれる入力に対し、R_j に含まれる学習データの応答変数の平均値を出力する。

予測変数の空間を J 個の領域 $R_1, R_2, \ldots, R_j$ に区分ける方法として、RSS を最小化する区分けを考えうる全てのくみあわせを試すことは計算量的に難しいため、とりあえず今ある空間を全ての予測変数と分割の基準値の組み合わせを考え、 学習データの正解ラベルとモデルによって予測された値の差分の二乗和が最小になるよっっっっそく変数と分割の基準値を見つける方法があります。

[*16] 単語を特徴量として用いる場合は基本的にはある文章にある単語が含まれているかどうかに基づいて特徴量をつけます。 ただここで他の質的変数のように 0/1 の二値で値をつけると、文章の中である単語が他と比較して多く出現しているなどの情報を生かしきれなくなります。実際には tf-idf という手法を用いて文章を n 種類の単語に対応したベクトルに変換することが 2000 年代までは主流でした。ちなみに深層学習モデルにおいては単語の離散表現（それぞれの単語を 数百次元のベクトルで表現する）を用いることが一般的です。

[*17] ある変数間で相関がある時、それが解析に支障が出るほど X 同士の相関が強い時に、「多重共線性（たじゅうきょうせんせい）があると言います。 http://xica.net/vno4ul5p/

[*18] https://vincentarelbundock.github.io/Rdatasets/doc/vcd/Hitters.html

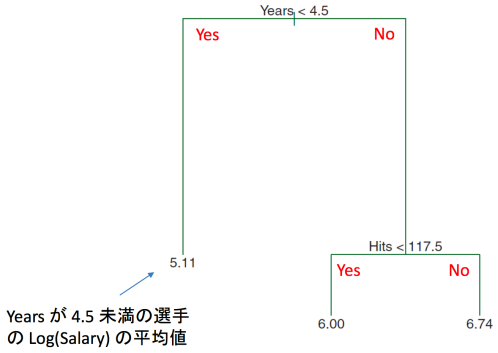

図 11　Hitters データに対する回帰木予測

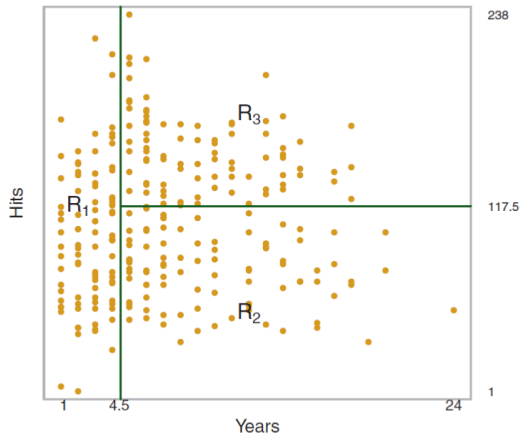

図 12　Hitters データに対する領域分割

貪欲法による領域の分割

全ての予測変数 $X_1, X_2, \ldots, X_p$ と分割の基準値 s を組み合わせ、以下が最小になる分割方法を選択する。

$$\sum_{i:x_i \in R_1(j,s)} (y_i - \hat{y}_{R_i})^2 + \sum_{i:x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

$$R_1(j,s) = \{X | X_j < s\}$$

$$R_2(j,s) = \{X | X_j \geq s\}$$

このような方法により分割された領域を示しているのが図 10 であり、R_1, R_2, R_3 が終端ノード (つまり分類木ならこれが予測カテゴリに、回帰木ならこのリージョンに含まれたデータの平均値が予測結果となります) です。 仮に Years < 4.5 以下の選手の事例が与えられれば彼は自動的に R_1 に分類され、Years > 4.5 かつ $Hits \leq 117.5$ ならばこの事例は R_2 に含まれることが分割図からもわかります。 ただ、分割によって作られた二つの領域をさらに同様に分割し、領域に含まれる事例の数が閾値以下になるまで繰り返し再帰的に領域を構築していく こういった分割方法だと、領域同士が入り組んだ分割を行うことはでき図、複雑な分割には対応できない恐れがあります。また、回帰木はデータに対して容易にオーバーフィットしてしまうため、枝刈りによって木のサイズを小さくして過学習を抑制する必要があります。

枝刈り

一旦木を構築してしまってから、Cost Complexity Pruning によって多すぎる枝分かれ（領域）を減らしていくこと。

$$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{Rm})^2 + \alpha |T|$$

ここで T は木のノード数です。

Cost Complexity Pruning は、木のノードの数（すなわち枝分かれの数）に対してペナルティをつけて行き、ただ RSS でフィットさせるのではなく、どこの枝同士をマージすべきかを計算して求めます。ここで α は任

意の値であり、 α を増やすと木は縮みます。このチューニングも縮小推定の λ のチューニングと同様、交差検証によって行います。

8.2 分類木

分類木とは質的変数（カテゴリなど）を応答変数（予測したい値、Y）とし、末端ノードの出力は回帰木のような平均ではなく、多数決でラベルを決定する。 分類木の構築は回帰木と同様、再帰的な分割を繰り返すことにより行います。しかしここで値を求める際の指標では分類誤り率（正しくカテゴリが予測されているか否か）ではなく、 Gini index や entropy 等が用いられることが多いです。単なる正解率で考えないのは多数決により分類木のラベルを決定するため、単純な正解率によるモデル評価はこの多数決で少数派となったラベルについての情報が考慮されなくなってしまうためです。

Gini inde と entropy

$\hat{p}_{mk}$ を領域 m に含まれるクラス k の事例の割合とすると、

Gini index

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

entropy

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

枝刈りの結果、NO と NO、YES と YES に分岐するノードが存在するが、これは予測するクラスは同じではあるが、「どのくらいの確率で YES/NO か」が異なってくる。

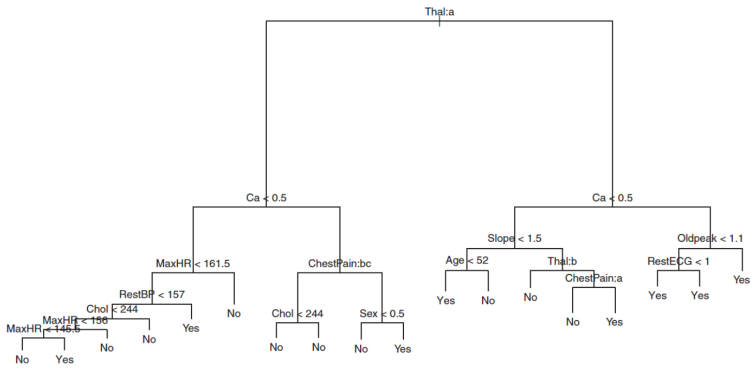

図 13　分類木による Heart データセット予測（枝刈り後）

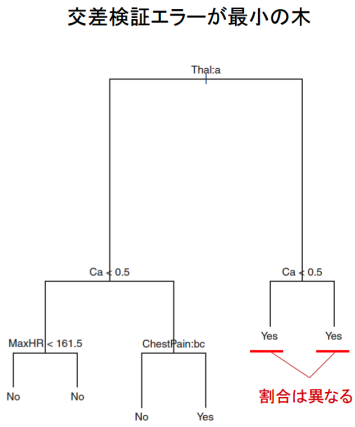

図 14　分類木による Heart データセット予測（枝刈り後）

8.3 決定木の長所短所

決定木の長所としては、以下の二点を上げることができます。

- モデルが実際に利用した条件分岐などを確認でき、またそれも極めて人間の行う意思決定の方法に近いため学習結果の解釈が容易。
- 質的変数を扱う際にダミー変数を作らなくてよい

逆に短所としては

- 回帰や分類の精度はあまり高くない（特殊なデータを除き、一般的な回帰モデル、分類モデルの方が高い精度を示すことが多い)
- 学習データにオーバーフィットしやすいため、学習データの小さな違いで構築される木が大きく変わることがある

例えば映画レビューをネガティブなものかポジティブなものか判断するとき、決定木モデルを用いるとある単語の有無で条件分岐を行うため、 とてもネガティブな内容ではあるものの、学習データのポジティブなメールに多く含まれていた単語が多用されているレビューに対しては正しく予測できない可能性がより高くなる。ただ決定木はアンサンブル学習の「ランファムフォレスト」、ブースティング等により精度の向上が可能で、以下では以下に決定機による予測精度を向上するかについて説明します。

8.4 バギング

バギングはブートルトラップ（一つ抜き法によるサンプリング）によってバリアンス（異なる学習データ間での結果のばらつき。学習データによってフィットしすぎる結果、異なる学習データに対応できなくなる）を減らすことを目指す方法です。 この出力は回帰木の出力の平均になります。Out-of-bag (OOB) エラー推定が便利です。複数の決定木の平均を取ることにより、個々の決定木よりも精度が高くなります。

バギング (bagging)

手順は以下の通りになる。

1. 学習データから n 個の事例をランダムに復元抽出し、抽出された n 個の事例を学習データとして決定木を構成する。
2. これを B 回繰り返す。
3. 得られた B 個の回帰木の出力の平均値が出力となる。分類木ならば平均を取る代わりに多数決で出力ラベルを決定する。

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x)$$

また、バギングで用いられるエラー推定法を Out-of-bag エラー推定法といます。これは LOO 交差検証とほぼ同等の推定結果を交差検証を行うことなく得られるので、 効果的な推定法であると言えます。

Out-of-bag(OOB) エラー推定

OOB(Out-of-bag) とはブートストラップサンプルによって選択されなかったデータであり、 OOB エラー推定とは B 個の回帰木に対して学習をさせた後、それぞれの OOB を用いて、OOB の中の i 番目のデータと実際の結果 yi を比較しテストを行い、 この誤り率 (OOB 推定値) をみる。

8.5 ランダムフォレスト

バギング [*19] はシンプルに B 個の決定木を構成し、その平均を取っているシンプルなモデルでした。これだけでも単一の決定木よりは精度が出るのですが、これをさらに改善した手法に「ランダムフォレスト」というモデルがあります。 ランダムフォレストはそのシンプルかつ高い精度を実現するモデルであり SVM 等と共に現実社会でも（割と）使われている機械学習モデルです。

ランダムフォレスト (random forest))

基本的な実行手順はバギングと同じ。ランダムフォレストでは、これに加えて決定木を構築するための再帰的な分割の際に考慮する予測変数を制限する。ランダムフォレストでは、各分割で考慮する予測変数をランダムに選択された m 個の予測変数に限定する。$m = \sqrt{p}$ がよく用いられる。

つまり、使用する入力変数を $m = p$ としたランダムフォレストモデルはバギングと考えることもできます。あえて入力変数を減らした方が複数の決定木のアンサンブルにおいて精度が向上する理由としては、 このように入力変数をランダムに選ぶことで、単純なバギングよりも構築される決定木のバリエーションが増え、構成された決定木の相関が減るためです。ちなみに分類の場合は $m = \sqrt{p}$ が一般的ですが、回帰の場合は $m = p/3$ が用いられることが多いらしいです。

バギングやランダムフォレストはシンプルな方法で精度を向上することが可能である一方、モデルの解釈が難しくなってしまう、またランダムフォレストの場合は分割によって RSS や Gini Index が減少してしまうという問題があります。

8.6 ブースティング

ブースティングとは、広義ではそこまで真の分類を精度高く予測できるわけではないモデル (弱学習器) を多数構築し、組み合わせることによりより高い精度をもつモデル（強学習器）を作ること意味します。ここで組み合わされるモデルは決定木に限らず、様々なモデルに適用可能です。

ブースティング (boosting)

一連の弱学習器を多数構築し、まとめることで強学習器を構成し、性能を向上させること。

ブースティング自体は決定木に限定した内容ではないのですが、回帰木にブースティングを適用することにより、より性能の高いモデルを構成することができます。 (ブースティングの手順のスライドがちょっとわかりにくいので多分後で補足します)

回帰木におけるブースティングでは、小さな回帰木を多数作り、構築されたモデルによる予測結果と実際の

[*19] この論文のランダムフォレストやブースティング、OOB、SVM の説明が結構わかりやすいです。時間のある方は読んでみてください。「ランダムフォレスト法による文章の書き手の同定」http://www.ism.ac.jp/editsec/toukei/pdf/55-2-255.pdf

データとの残差を応答変数として回帰木を作ります。最終的なモデルの出力は構築された多数の回帰木の出力の和になります。

9 サポートベクターマシン (第 12 講)

9.1 サポートベクターマシンとは？

サポートベクターマシンは元々二値分類 [*20] のための識別モデルで、1990 年代に開発されその高い精度から現在でも様々な場面で活用されています。 一般的に「サポートベクターマシン」と総称することが多いのですが、

- 最大マージン分類器： マージンを最大化することによる分類。線形分離可能な場合以外は適応が難しい。
- サポートベクター分類器：線形分離可能でない場合でも適用可能だが、分離超平面は線形な関係に基づいている。
- サポートベクターマシン：カーネルにより非線形分類が可能であり線形分離の極めて難しい入り組んだデータでも対応は可能。

と実は細かく定義が分かれています [*21]。

9.2 最大マージン分類器

ある入力 X がそのクラスに所属するかを超平面 (hyperplane) により分割するものです [*22]。 超平面とは言ってはいますが、入力が二次元であれば直線、三次元であれば平面になります。

最大マージン分類器

p 次元空間中の超平面とは、p-1 次元の平坦な部分空間であり、以下の式で表される。

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

空間は超平面により超平面より大きいか小さいかにより、二つに区分される。

$$\begin{cases} \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > 0 & (7) \\ \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < 0 & (8) \end{cases}$$

では、このような超平面はどのように求められるのでしょうか。まず、以下のような $n \times p$ のデータ行列 X、$n \times 1$ のクラスラベル y から構成される学習データと、$1 \times p$ で構成されるテストデータ x^* を考えます。 つまり、p 個の入力変数をもつ n 個のデータとそれに対応した正解ラベルのペアを使って、新しい 1 個のテスト

[*20] クラスが二つの場合の分類問題

[*21] 実際に使用する場面ではほとんど区別していないと先生談

[*22] ちなみに SVM は若干概念的にこれまでのモデルよりややこしく感じるかもしれません。この資料とかがわかりやすいかも。 http://www.sakurai.comp.ae.keio.ac.jp/classes/infosem-class/2005/07SVM.pdf

事例（同様に p 個の入力変数を持つ）を正しく分類する分類器をこれから構築していくことを考えます。

$$X = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix}$$

$$Y = (y_1\ y_2\ \dots\ y_n)^T \in \{-1, 1\}$$

$$\boldsymbol{x^*} = \begin{pmatrix} x_1^* \\ x_2^* \\ \vdots \\ x_p^* \end{pmatrix}$$

入力データ X, Y に基づいて分離超平面を作り、この超平面によってテスト事例 x^* の分類を行います。

分離超平面

学習事例をすべて正しく分類する超平面を最大マージン分類器における分離超平面という。

$$\begin{cases} \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > 0 \text{ if } y_i = 1 & (9) \\ \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < 0 \text{ if } y_i = -1 & (10) \end{cases}$$

より、全ての i に対して $y_i(\beta_0 + \beta_1 x_{i2} + \ldots \beta_p x_{ip}) > 0$ が成立し、テスト事例の分類は x^* をこの分離超平面に代入した時の正負により行えれる。

分離超平面の決め方ですが、全ての学習事例を正しく分類できてれば良いため、分離超平面は無数に存在しうることになります。ただ学習事例を分離可能な複数な超平面の中にも、 少しノイズが入っただけで誤った分類をしてしまいそうな危うい超平面よりは、どちらのクラスよりもある程度離れている（つまり超平面が入力データのギリギリに惹かれているわけではない）超平面を選んだ方が、 新たな入力に対しても頑健なモデルになりそうです。これが最大マージン分類器の基本的な考え方になっています。

最も近い学習事例までの距離をマージン、超平面に最も近い事例をサポートベクターと言います。最大マージン分類器は、このマージンを最大にすることにより、様々な入力に対して適切に分類を行うことのできるモデルの構築を目指しています。

最大マージン分類器の構築

学習事例を $x_1, \ldots x_n$、それに対応したクラスラベル $y_1 \ldots y_n$ の最大マージン超平面は以下の最適化問題をとくことにより求められる。

$$\underset{\beta_0, \beta_1, \ldots, \beta_p}{\text{Maximize M}}$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1$$

$$y_i(\beta_0 + \beta_1 x_{i2} + \ldots \beta_p x_{ip}) \geq M \forall i = 1, \ldots, n$$

つまり、$\sum_{j=1}^{p} \beta_j^2 = 1$ の時、$y_i(\beta_0 + \beta_1 x_{i2} + \ldots \beta_p x_{ip})$ は超平面への距離を表しているため、全ての事例のうちこれが最も小さくなる事例（サポートベクター）におけるマージン M を最大にすることにより、超平面を求めています。(なぜこれで超平面までの距離を表すことができるかはスライドを参照してください。)

最大マージン分類器は以下のような問題点があります。

- そもそもデータが入り組んでおり、分離超平面を作ることができない
- 分離超平面があった場合でも、学習データの変化に対して敏感に反応してしまうため、与えるデータによって大きく異なるモデルとなってしまう

9.3 サポートベクター分類器の構築

こういった最大マージン分類器の問題に対応することを目指しているのがサポートベクターマシンであり、マージンが小さく、設定されたマージンの内側に入ってしまうような事例や正しく判別されない事例を許容することで、データに大きく左右されないより頑健な [*23] モデルを構築することを目指しています。

サポートベクター分類器

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\varepsilon_1,\ldots,\varepsilon_n}{\text{Maximize M}}$$

$$\text{subject to } \sum_{j=1}^{p}\beta_j^2 = 1$$

$$y_i(\beta_0 + \beta_1 x_{i2} + \ldots \beta_p x_{ip}) \geq M(1-\varepsilon_i) \forall i = 1, \ldots, n$$

$$\varepsilon_i \geq 0,\ \sum_{i=1}^{n}\varepsilon_i \leq C$$

ε をスラック変数といい、$\varepsilon = 0$ ならば違反を許さず、この分類器は上述の最大マージン分類器と同値になりますが、$\varepsilon > 0$ ならば、ある事例がマージン M より ε 分だけ分離超平面に 接近していることを許すため、マージンの制約に違反していることがわかります。また、$\varepsilon > 1$、例えば ε を 2 に設定すると、$1 - \varepsilon = 1 - 2 = -1$ となり、クラス A に分類されるはずの事例が超平面の向こう側 (つまりクラス B) に分類されることを許容していることになります。 つまり ε を大きくすることでより許容度を上げていくことができるのですが、全て許容していてては逆に全く正しくない超平面となってしまうためある定数 C に制限し、違反を合計でどれだけ許すかを定めます。C の値は交差検証等で適切な値を探すことが多いのですが、この大小はモデルに対して以下のような影響を与えます。

C の値が大きい時

- C を大きくすると、超平面の反対側やマージンの内側に事例が存在することが多くなるため、サポートベクターの数が多くなる。
- 入力事例の変化には頑健になるためバリアンスは小さくできるが、バイアスは大きくなってしまう。

C の値が小さい時

- C が小さければモデルはより最大マージン分類器に近くなりサポートベクターの数は少なくなる。
- C が小さいと入力される事例の変化によって超平面がどんどん動くので、バリアンスは大きいが、今あるデータにはよくフィットするためバイアスは小さくなる。

[*23] 余談ですがこういった入力データにいちいち左右されない頑健なモデルのことをロバスト (robust) なモデルと言います

スラック変数により、サポートベクター分類器である程度入り組んでいるようなデータにも対応できるようになりました。しかし、サポートベクター分類器は許容度をあげてはいるものの、線形な境界に基づいた分類であるため完全に入り組んだ（螺旋状にデータが散らばっている等）の事例にはうまく対応ができません。次のサポートベクターマシンはこういったケースに対応するため非線形な境界による分類を可能にしています。

9.4 サポートベクターマシン

サポートベクター分類器はスラック変数によりマージンの内側もしくは反対側に入るような事例をある程度許容することで空間を分類することを目指していましたが、 かなり入り組んだような事例ではそもそも線形の分離面で分類することは難しいこともあります。サポートベクターマシンとは、このような場合に非線形な境界を引くことによって入り組んだ事例を分類します。

サポートベクターマシン

非線形カーネルを用いたサポートベクター分類器である。

$$f(x) = \beta_0 + \sum_{i \in S} \alpha y_i K(x, x_i)$$

S : サポートベクターのインデックスの集合

特徴空間を拡張するのではなく、カーネル関数を用いることで非線形な境界を引くことを可能にする。

これまでの項で触れたように非線形な境界による分類を行いたいときは、入力 $X_1, X_2, \ldots X_p$ だけを用いるのではなく、X_1^2のような項や X_1X_2 のような項を加える、すなわち特徴空間を拡張することができました。 しかしこのような特徴量空間の拡張では複数の入力変数を掛け合わせる項や n 乗項が登場することになり特徴量の数を大きく増加させてしまいます。 一般に、サポートベクター分類器はテスト事例と学習データの事例間の内積の和で表現が可能であり、事例ベクトル間の内積を

$$< x_i, x_i' >= \sum_{j=1}^{p} x_{ij} x_{ij'}$$

とすると、境界面は以下の式で表すことができます。

$$f(x) = \beta_0 + \sum_{i=1}^{n} a_i y_i < x, x_i >$$

ここで x_i がサポートベクターでなければ $\alpha_i = 0$、サポートベクターであれば $\alpha_i > 0$ となり、 サポートベクターでない事例についてはこれまでと同様、境界面には影響を与えないことがわかります。 ただ、このような内積計算は特徴量空間の大きさ p が大きい時などはかなり計算が必要になってしまいます。 そこで計算量を抑えながら非線形境界を引くために内積に置き換えて用いられるのが「カーネル関数」です。

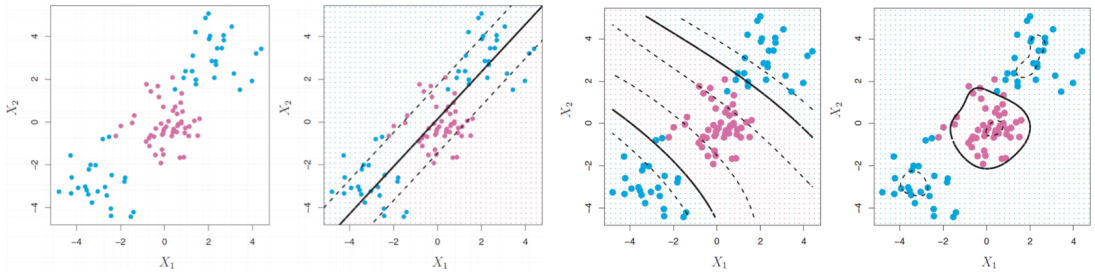

図 15　左から順に、元の特徴量空間、サポートベクター分類器による分類結果、多項式カーネルを用いた SVM による分類結果、RBF カーネルをもちいた SVM による分類結果

カーネル

内積の計算をすべてカーネル関数に置きかえ、 明示的に特徴量を増やすことなく非線形な決定境界を構成する。 以下のカーネルが SVM でよく使用されるものである。

線形カーネル

そのまま内積計算を行う。

$$K(x_i, x_i') = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

多項式カーネル

多項式で計算を行う。

$$K(x_i, x_i') = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

RBF **カーネル** (Gaussian **カーネル**)

点と点の距離が離れると小さくなる。$(x_{ij} - x_{i'j})^2$ が距離を表す。

$$K(x_i, x_i') = \exp\left(-\gamma \sum_{j=1}^{p} (x_{ij} x_{i'j})^2\right)$$

実際にこれらのカーネルを使うと何がいいのでしょうか。まず多項式カーネルの場合を考えると、$p = 2, d = 2$ の時、

$$K(x_i, x_i') = (1 + \sum_{j=1}^{2}) x_{ij} x_{i'j})^2 = (1 + x_{i1} x_{i'1} + x_{i2} x_{i'2})^2$$

これを展開した結果は拡張された特徴空間での内積結果と一致するため、非線形な分類を行うための特徴量が自然に結果に入ってくることになり、非線形な境界が作れるようになります。 このようにカーネル関数を活用することにより、事例同士の内積計算の増加を招いてしまう、特徴量空間の拡張を行うことなく同様の結果を得ることができます。 図 15 が実際にサポートベクター分類器の線形境界で分割できない領域についても多項式カーネル、RBF カーネルがうまく分離を行なっていることが確認できます。

10　教師なし学習

これまでは教師あり学習、つまり入力変数 X から出力変数 Y をいかに精度高く予測するかという問題でした。一方、この項で紹介する教師なし学習では、 入力データに対して正解データが与えられず、いかに上手く入力データを分析するかという問題になります。

教師なし学習

入力変数 $X_1, X_2 \dots X_p$ のみを使い、主成分分析やクラスタリングを行ってデータの構造を分析すること。 教師あり学習のように客観的、定量的にモデルを評価することが難しいが、人間が高次元の複雑なデータを理解するために重要なアプローチである。

主な手法には主成分分析、クラスタリング (Kmeans、階層的) があります。

10.1　主成分分析 (Principal Component Analysis, PCA)

特にデータのクラスタリングで 10 次元等の高次元のデータではデータが実際にどう分布しているのか可視化することは難しく、またデータにか学習しやすくなってしまいます。主成分分析は、多次元データのもつ情報をできるだけ損わずに低次元空間に情報を縮約し、視覚化等を容易にすることを目指しています。

主成分分析 (Principal Component Analysis, PCA)

高次元のデータをなるべく情報を失うことなく、低次元データ（例えば 2 次元等）に変換する方法。可視化や構造抽出、過学習の回避やノイズ除去に有用。

三次元空間を二次元で落とし込むことを考えると主成分分析についてイメージがしやすいかも知れません。三次元データを二次元データに落とし込みたい場合は、この三次元空間上で各事例（各点）との距離の和が最小になるような平面を求め、 データをこの平面に射影することで三次元データを二次元データに落とし込みます。主成分分析は、第一主成分及び第二主成分を計算することで求められます。

第一主成分と第二主成分の求め方

まず前処理として各次元について平均をゼロに揃える。

$$\sum_{i=1}^{n} x_{ij} = 0$$

第 1 主成分

射影したデータの分散が最大となるような軸を探す。分散が最大になるような特徴量の線形結合を行う。

$$Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + \phi_{p1}X_p$$

これは負荷量ベクトルの二乗和が 1 になるという制約下での第一主成分のスコアを最大にする最適化問題として捉えることができます。

第 2 主成分

第 1 主成分と直交する軸の中で、軸上に射影したデータの分散が最大となる軸を探す。 Z1 と相関がなく、分散が最大になる特徴量の線形結合と考えられる。

$$Z_2 = \phi_{12}X_1 + \phi_{22}X_2 \ldots \phi_{p2}X_p$$

第 2 主成分において、負荷量ベクトル $\phi_2 = (\phi_{12}, \phi_{22} \ldots \phi_{p2})$ は Z_1 と無双感であるため、ϕ_1 と直交する。

上で紹介しているのは第 2 主成分までですが、第 3 主成分以降も、 それ以前の主成分と相関がなく、分散が最大になる特徴量の線形結合を行っていけば求めることができます。詳細な求め方は授業でもあまり立ち入っていないので省略しますが、基本的には数値計算で行列の固有ベクトルを求めます。 主成分分析の結果得られた第一主成分、第二主成分がどれくらいデータを上手く捕らえられているかを測る指標を寄与率 (Proportion of variance explained, PVE)[*24] と言います。

[*24] どうでもいいんですけど PVE って直訳すると被説明分散比率とかになると思うんですけどなんで偉い人最初に寄与率って和訳つけたんでしょうか。連想できなくないですか…?

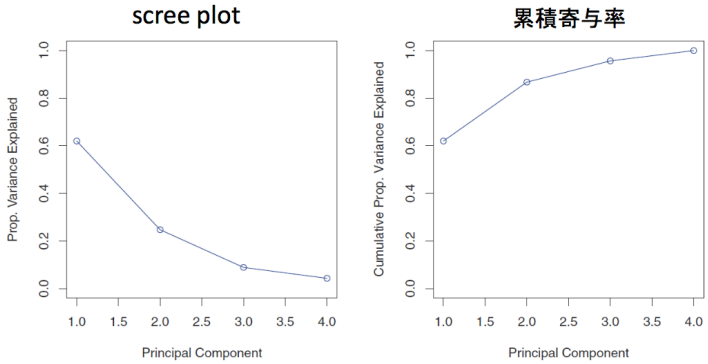

図 16　寄与率グラフ

寄与率 (Proportion of variance explained, PVE)

元のデータの情報 (分散) をどれだけ説明できているかを測る指標。元のデータの分散が以下で表される。

$$\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2$$

また第 m 主成分によって説明される分散を

$$\frac{1}{n} \sum_{i=1}^{n} z_{im}^2 = \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2$$

とした時、第 m 主成分の寄与率は (第 m 主成分により説明される分散)/(元のデータ全体の分散) で求められるため、

$$\text{第 } m \text{ 主成分の } PVE = \frac{\frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2}{\sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2} = \frac{\sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2}$$

図 16 は Advertising データの主成分分析結果 (だったと思います) であり、左のグラフの x 軸が第 m 主成分、y 軸がその第 m 主成分の PVE を表しており、 このグラフでは第 1 主成分が元の分散の 6 割程度、第 2 主成分が元の分散の 2 割程度、第 3 主成分が元の分散の 1 割程度と、第 3 主成分までで元の分散の 9 割程度まで説明できていることがわかります。

10.2　クラスタリング

クラスタリングとはデータをサブグループ (クラスタ) に分け、それぞれのクラスタの性質等の分析に役立てます。 またそれ以外の用途として半教師あり学習があり入力データに対するラベル情報が少ない時、 まずクラスタリングで入力データだけを使ってデータをいくつかのクラスタに分けた後、少ない分類ラベルを用いてそれぞれのクラスタにラベルを振っていくことで、 ラベルのある教師データが少ない時でも正しいモデルを作ることに役立てたりもします。

クラスタリング (clustering)

データをサブグループ (クラスタ) に分けることであり、元のデータを類似の事例 (距離が近い事例) が同一クラスタに属するように分割する。 事例間の距離についてはユークリッド距離など、事前に定義を行う必要がある。非階層的な手法と階層的な手法がある。

ユークリッド距離は簡単に言えば人が測るような二点間の「通常の」距離であり、例えば x_i, x_j という二つの事例について、p 次元空間（つまり p 個の特徴量が与えられている）におけるユークリッド距離は以下の式で求められます。

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{p} (x_{ik} - x_{jk})^2}$$

クラスタリングにおいて代表的なのは以下の二つの手法になります。

- K-Means クラスタリング (非階層的クラスタリング)
- 階層的クラスタリング

10.2.1 K-Means クラスタリング

K-Means クラスタリング (K-平均法) (あんまりちゃんと聞いてなくてスライドの定義式にある「インデックスの集合」がよくわかりませんごめんなさい…多分テスト前に補足します…は各クラス内の広がりが最小になるように分けることを目指しており、以下のような手順で予測を行います。

K-Means クラスタリングの手順

1. 各事例の所属するクラスタをランダムに決める
2. ランダムに決められた各クラスタの重心を計算する。

$$\overline{x_k} = \frac{1}{|C_k|} \sum_{i \in C_k} x_i$$

3. 各事例の所属するクラスタを、重心が最も近いクラスタとする
4. 収束するまで 2 及び 3 を繰り返します。

簡単に言うと、まずランダムに入力事例をいくつかのクラスタに分けた後、このランダムなクラスタの重心を求めます。次に、それぞれの点を元々のランダムなクラスタ分けから、自分に重心が最も近いクラスタに振り分け直します。そうするとさっきとは重心が変わってしまうので、 この新しいクラスタ分けで重心を計算し直します。そしてまたそれぞれの事例のクラスタを最も重心が近いクラスタ C_i にして…をクラスタ内の点同士の距離の総和の平均が最小になるまで何度も繰り返します。

K-Means クラスタリングには注意点があります。Kmeans クラスタリングで得られる結果は局所最適解であり、上の手順からもわかるように最適化アルゴリズムの収束先は初期解 (つまり最初にランダムに行ったクラスタリング) に依存します。そこで初期解があまりよくなかった場合精度が十分に出ない恐れ [*25] があるため、最適解に近い解を得るためには何回か初期値を変えて実行する必要があります。 解は何回か変えて実行した者のうち、最も評価値の高かったクラスタリングを選択します。

[*25] ソフトウェア II でやった巡回セールスマン問題とかも、ランダムに初期値を形成する場合はその初期値によって結構大きく結果が変わってきましたよね

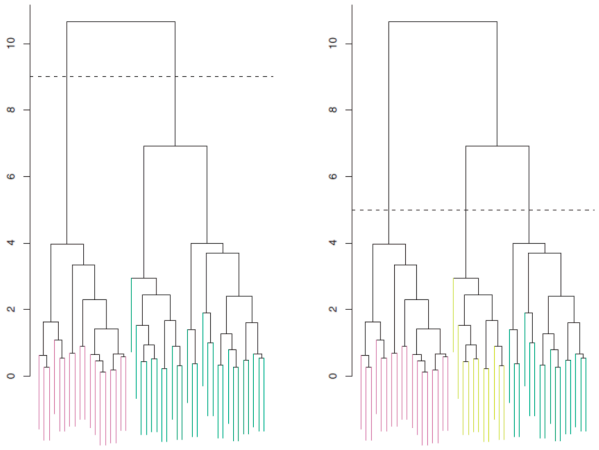

図 17　階層的クラスタリング

10.2.2　**階層的クラスタリング**

クラスタ数をあらかじめ決めることなく、距離的に近い事例をボトムアップにまとめあげていく手法を階層的クラスタリングといいます。

階層的クラスタリング (hierarchical clustering)

N 個の入力データから類似度の高い順に融合して次第に大きなクラスタを作り、最終的には N 個のデータを一つのクラスタに統合するクラスタリング手法。 結果は樹状図 (dendrogram) として得られる。クラスタを構成する手順は以下のようになる。

1. n 個の各事例を単一の事例からなるクラスタとする。
2. 距離が最も近いクラスタを統合する。
3. クラスタ間距離を更新する。
4. これをクラスタの個数が一つに収束するまで繰り返す。

階層的クラスタリングはその定義の通り、まずある事例に最も近い事例をまとめ、その二つから近い別の一つの事例と今度はまとめ、さらにまだ残っている近い事例もまとめ上げ…と一つにまとまるまで繰り返していく方法です。 甲子園の地方大会のトーナメント表とかをイメージしてみると樹状図のイメージはつくんじゃないでしょうか。 実際に出来上がった樹状図から決まった数のクラスタリングを得る方法はとてもシンプルで、距離の閾値を設定してそこに線を引き、 その線と交わる樹状図中の縦線の数だけクラスタリングができます。図 17 はこの階層的クラスタリングの例を示しており、 距離の閾値を 9 で切った左側のグラフでは二つの樹状図の縦線が交わっていることから最大距離が 9 まで離れた二つのグループができ、距離の閾値を 5 で切った右側のグラフでは三つのクラスタができていることがわかります。ちなみに三個以上の事例をクラスタリングしていく際のクラスタ間距離の定義には以下の 4 つの手法があります。

完全リンク (complete-linkage)

あるクラスタ C_i とあるクラスタ C_j 間のクラスタ間距離は、それぞれのクラスタの中で最も遠い事例間の距離とする。つまり最も類似度の低い事例間の距離。

単一リンク (single-linkage)

あるクラスタ C_i とあるクラスタ C_j 間のクラスタ間距離は、それぞれのクラスタの中で最も近い事例間の距離とする。

平均リンク (average-linkage)

あるクラスタ C_i とあるクラスタ C_j 間のクラスタ間距離は、それぞれのクラスタの中の事例間の距離の平均とする。

重心リンク (centroid-linkage)

あるクラスタ C_i とあるクラスタ C_j 間のクラスタ間距離は、重心間の距離とする。

先生曰く実際によく使われるのは完全リンクか平均リンクらしいですが、単一リンクだと計算を早く済ませることができます。

11 参考資料

授業で「参考」とされた内容や数式の計算過程等をまとめてあります。

11.1 予測誤差の大きさ（第 3 講）

予測結果 $\hat{Y}$ と実際のラベル Y の二乗誤差の大きさの期待値から予測誤差を求めると、予測誤差に削減可能な誤差と削減不可能な誤差があることがわかります。統計的機械学習において、$Y = f(X) + \epsilon$ で表されるため、

$$\begin{aligned} E[(Y - \hat{Y})^2] &= E[((f(X) + \varepsilon) - \hat{f}(X))^2] \\ &= E[f(X)^2 + \varepsilon^2 + \hat{f}(X)^2 + 2\varepsilon f(X) - 2\varepsilon\hat{f}(X) + 2f(X)\hat{f}(X)] \\ &= E[(f(X) - \hat{f}(X))^2 + 2(f(X) - \hat{f}(X)\varepsilon) + \varepsilon^2] \\ &= E[(f(X) - \hat{f}(X))^2] + 2E[(f(X) - \hat{f}(X)\varepsilon] + E[\varepsilon^2] \end{aligned}$$

3 行目から 4 行目あたりの変形については $E[X + Y] = E[X] + E[Y], E[aX] = aE[X]$ を用いた。ここで $f(X),\ \hat{f}(X))^2$ はいずれも決定項（確率変数ではない）ので、期待値操作とは無関係になる。そのため $E[(Y - \hat{Y})^2]$ は

$$\begin{aligned} E[(Y - \hat{Y})^2] &= (f(X) - \hat{f}(X))^2 E[1] + 2f(X)E[\varepsilon] - 2\hat{f}(X)\varepsilon E[\epsilon] + Var(\varepsilon) \\ &= (f(X) - \hat{f}(X))^2 + Var(\varepsilon) \end{aligned}$$

となり、第 1 項は学習したモデル $\hat{f}(X)$ による予測結果と実際の関数 $f(X)$ の値のずれを示しており、削減可能な誤差ですが、第 2 項は元々のデータに含まれる統計的な誤差 ε によって生じており、 削減のできない誤差になります。

11.2 線形単回帰のパラメータの推定法 (第 4 講)

線形単回帰の RSS を最小にする $\hat{\beta}_0, \hat{\beta}_1$ は、ぞれぞれで RSS を偏微分し、偏微分した結果を 0 にする $\hat{\beta}_0, \hat{\beta}_1$ を 連立方程式にとけば求めることができます。

$$\mathrm{RSS} = (y_1 - \hat{\beta}_0 - \hat{\beta}_1 x_1)^2 + (y_2 - \hat{\beta}_0 - \hat{\beta}_1 x_2)^2 \ldots (y_n - \hat{\beta}_0 - \hat{\beta}_1 x_n)^2$$

$$(y_1 - \hat{\beta_0} - \hat{\beta_1} x_1)^2 = y_1^2 + \hat{\beta_0}^2 + \hat{\beta_1}^2 x_1^2 - 2y_1\hat{\beta_0} + 2\hat{\beta_0}\hat{\beta_1}x_1 - 2\hat{\beta_1}x_1 y_1 \text{より}$$

$$\begin{cases} \dfrac{\partial \mathrm{RSS}}{\partial \hat{\beta_0}} = 0 \iff \displaystyle\sum_{i=1}^{n} y_i - \beta_1 \sum_{i=1}^{n} x_i - \beta_0 n = 0 & (11) \\ \dfrac{\partial \mathrm{RSS}}{\partial \hat{\beta_1}} = 0 \iff \displaystyle\sum_{i=1}^{n} x_i y_i - \beta_1 \sum_{i=1}^{n} x_i^2 - \beta_0 \sum_{i=1}^{n} x_i = 0 & (12) \end{cases}$$

式 (3) より

$$\beta_0 = \frac{1}{n}\sum_{i=1}^{n} y_i - \beta_1 \sum_{i=1^n} x_i \tag{13}$$

ここで

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{14}$$

式 (5)(6) を式 (4) に代入し、β_1 について解くと、

$$\begin{aligned} \beta_1 &= \frac{\sum_{i=1}^{n} x_i y_i - \frac{1}{n}(\sum_{i=1}^{n} y_i \sum_{i=1}^{n} x_i)}{\sum_{i=1}^{n} x_i^2 - \frac{\sum_{i=1}^{n} x_i^2}{n}} \\ &= \frac{\sum_{i=1}^{n}(x - \overline{x})(y - \overline{y})}{\sum_{i=1}^{n}(x - \overline{x})^2} \end{aligned}$$

また式 (5) より

$$\beta_0 = \overline{y} - \beta_1 \overline{x}$$

11.3 RSS の性質の導出 (第 4 講)

式 (5) について、全て一つのシグマでくくると

$$\sum_{i=1}^{n} y_i - \beta_1 x_i - \beta_0 = \sum_{i=1}^{n} e_i = 0 \tag{15}$$

よって、RSS に関する一つ目の性質が示された。また同様に式 (6) より

$$\sum_{i=1}^{n} x_i y_i - \beta_1 x_i^2 - \beta_0 x_i = \sum_{i=1}^{n} x_i(y_i - \beta_1 x_i - \beta_0) = \sum_{i=1}^{n} e_i x_i = 0 \tag{16}$$

よって二つ目の性質も RSS からパラメータを推定する際に用いた式より導くことができる。

11.4 標準誤差の求め方及び標準偏差 (第 4 講)

母平均の信頼区間 = 標本平均 $\pm\, t \times$ 標本標準偏差 $\div \sqrt{\text{標本数}}$ です。

$$\text{標準誤差 } SE = \frac{\text{標準偏差 } SD}{\sqrt{\text{標本数 } N}}$$

より、パラメータ β_0, β_1 の信頼区間はそれぞれ

$$\hat{\beta}_0 \pm t\mathrm{SE}(\hat{\beta}_0), \hat{\beta}_1 \pm t\mathrm{SE}(\hat{\beta}_1)$$

自由度	信頼係数：95%	信頼係数：99%
1	12.706	63.657
2	4.303	9.925
3	3.182	5.841
4	2.776	4.604
5	2.571	4.032
6	2.447	3.707
7	2.365	3.499
8	2.306	3.355
9	2.262	3.250

図 18　t 分布表の見方

となる。t の値は t 分布表から t の値を求めることができる。信頼区間の当たる確率、すなわち 95% 信頼区間であれば 95% を信頼係数といい、 標本数から 1 を引いた数を自由度と呼び、t 分布表ではこの自由度と信頼係数より t の値を求めることができます。 例えば標本数が 10 で 95% 信頼区間を求めたい時、以下の表より t は自由度が 9、信頼係数が 95% の 2.262 ですことがわかる。 標本数が大きくなるとこの t の値はほぼ正規分布の値と変わらず 2 前後になるため、線形単回帰のパラメータの信頼区間の推定において近似的に $t = 2$ とすることができる。(ザックりな説明ですみません… https://blog.apar.jp/data-analysis/4632/)

11.5　縮小推定の手計算（第 10 講）

$n = p, \beta_0 = 0$

$$X = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

という問題に対して、パラメータ推定及び縮小推定（リッジ推定、Lasso 推定）を行う。

まず、最小二乗法でパラメータを推定する際には、最小二乗法を用いる場合、その目的関数は以下のようになる。目的関数を最小化する β_i は目的関数の微分が 0 になる時となるため、

$$L = \sum_{j=1}^{p} (y_i - \beta_j)^2 \tag{17}$$

$$\frac{\partial L}{\partial \beta_i} = \frac{\partial}{\partial \beta_i} \sum_{j=1}^{p} (y_i - \beta_j)^2 = \frac{\partial}{\partial \beta_i} (y_i^2 - 2y_i\beta_i + \beta_i^2) = -2(y_i - \beta_i)$$

L に対して β_i の微分を取る時、i 番目以外のパラメータ β は無視してよいので、結果的に L を最小にする β_i の値は以下のようになる。

$$\beta_i = y_i \tag{18}$$

リッジ回帰においては目的関数が以下 (5) で表現されるため、同様に β_i で微分をし、L を最小化する β_i を求めると以下の (6) のようになる。

$$L = \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{19}$$

$$\frac{\partial L}{\partial \beta_i} = \frac{\partial}{\partial \beta_i} \left(\sum_{j=1}^{p} (y_i - \beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right) + 2\lambda\beta_i = \frac{\partial}{\partial \beta_i} (y_i^2 - 2y_i\beta_i + \beta_i^2) = 2(1 + \beta_i) - 2y_i$$

$$\hat{\beta_i^R} = \frac{y_i}{1 + \lambda} \tag{20}$$

(6) より、$\lambda = 0$ ならば目的関数を最小化する β_i がパラメータ推定の結果求められた (5) と一致することがわかり、 $\lambda = 0$ の縮小推定の結果はパラメータ推定により求められる値と一致することが確認できる。

Lasso 回帰においては目的関数が以下 (7) で表現される。Lasso 回帰においては正則化項が絶対値符号を含むため、正か負かで場合わけをする必要があります。ただやることはリッジと同じなので、$|\beta_j|$ が正か負かで場合分けして各自計算してみてください。

$$L = \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{21}$$

11.6 リッジ回帰、Lasso 回帰を不等式制約付きの最適化問題として捉える

リッジ回帰、Lasso 回帰は以下のような不等式条件付きの最適化問題として捉えることができ、この見方をするとなぜ Lasso 回帰においてパラメータが 0 に収束しやすいのか 理解がしやすくなります。

不等式条件つき最適化問題としての等式化

リッジ回帰

$$\begin{aligned} \underset{\beta}{\text{minimize}} \quad & \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \\ \text{subject to} \quad & \lambda \sum_{j=1}^{p} \beta_j^2 \leq s \end{aligned}$$

Lasso 回帰

$$\begin{aligned} \underset{\beta}{\text{minimize}} \quad & \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \\ \text{subject to} \quad & \lambda \sum_{j=1}^{p} |\beta_j| \leq s \end{aligned}$$

ちょっと眠いので省略なのですが、上の最適化問題の制約条件からわかるように、Lasso については制約条件を満たす領域が β_1, β_2 平面上の四角形になり、頂点はそれぞれ $(-s, 0),\ (s, 0),\ (0, -s),\ (0, s)$ の四点にきます。 一方、リッジ回帰の場合、制約条件によって許容される β_1, β_2 の領域は、半径が $\sqrt{s}$ 以下の円となります。 結果として、Lasso 回帰においては RSS(最小二乗誤差、$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$) の等高線が制

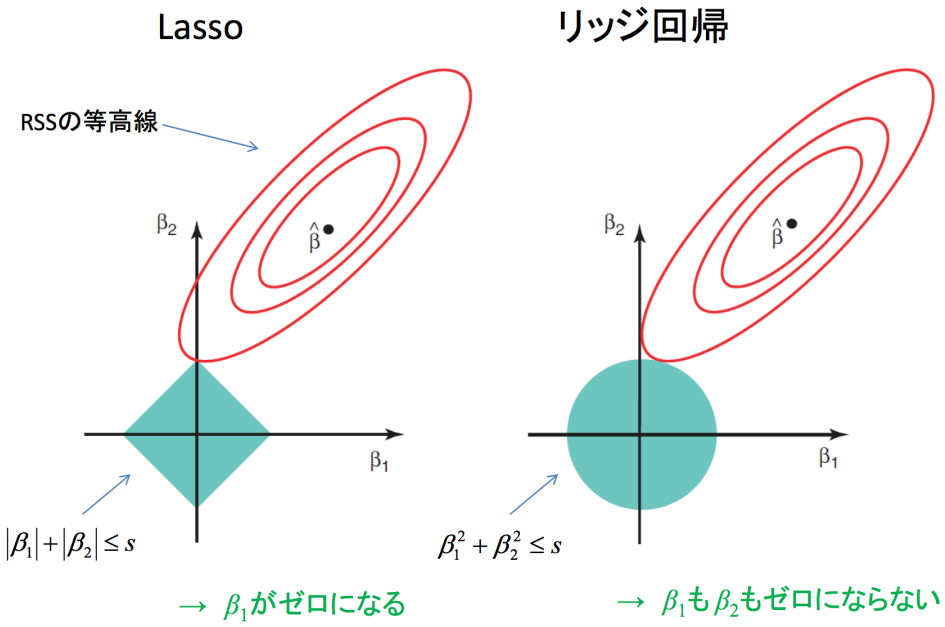

図 19　最適化問題としてリッジと Lasso 回帰をといた場合

約条件を満たす四角形の頂点に接する可能性もあり、 この場合、例えば RSS の等高線が $(0, s)$ に接していた時、パラメータ β_1 については 0 となるためこのパラメータ β_1 が消失することがわかります。一方リッジについては、例え RSS の等高線が制約境界上にあったとしても、どちらかのパラメータが 0 になることは発生しえないことがわかります（ほんとか？）。 これについてはスライドの図がわかりやすかったのでそのまま貼っておきます。