The Elements of Statistical Learning
Chap.18: High-Dimensional Problems: $p \gg N$

Kosuke Kito

August 30, 2020

Section 1

Introduction
p.653-

本日のお題 - $p \gg N$ 問題

特徴量の数がサンプル数よりもずっと大きいとき ($p \gg N$) に困っちゃう話.

- 困っちゃうポイントは, high variance と overfitting
- simple, highly regularized な手法が使われる.
- 主な話題は以下の２つ.
  - prediction
  - feature selection, assesment

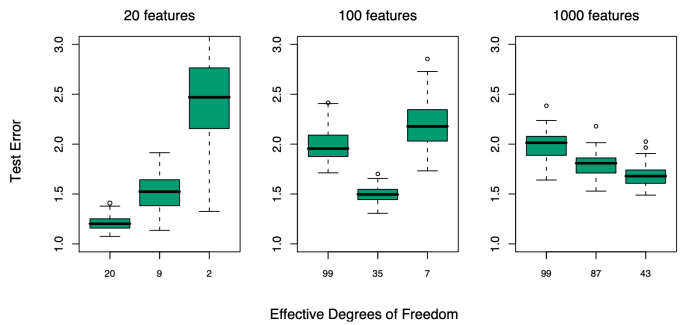


FIGURE 18.1. *Test-error results for simulation experiments. Shown are box-plots of the relative test errors over* 100 *simulations, for three different values of* p*, the number of features. The relative error is the test error divided by the Bayes error,* σ^2*. From left to right, results are shown for ridge regression with three different values of the regularization parameter* λ*:* 0.001, 100 *and* 1000*. The (average) effective degrees of freedom in the fit is indicated below each plot.*

流れ

- 流れを書く.

Section 2

LDA の正則化 - Diagonal LDA と NSC
p.651-

LDA の復習 1 - コンセプト

$p \gg N$ 問題の最初の回避策は, “Diagonal LDA” という線型判別法の強烈な正則化バージョン.
とりあえず, LDA の復習. (不要であれば飛ばします.)

- 分類のための手法.
- 各入力 x に対して, 事後確率 $\Pr[k \mid X = x]$ が最大になるクラス k をクラスの推定値とする.
- 各クラス内で, 入力変数は多変量ガウス分布に従うと仮定.
- 各クラスのクラス内分散が等しいと仮定.
  →クラス間の境界が線形になる.
- 数式で書くと次ページの流れ.

LDA の復習 2 - 定式化と計算

- 各クラス内の分布はガウス分布と仮定.

$$\Pr[X = x|G = k] = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_k)^{\mathrm{T}}\Sigma(x-\mu_k)}$$

- ベイズの定理より事後確率は以下.

$$\Pr[k \mid X = x] = \frac{\Pr[X = x|G = k]\Pr[G = k]}{\sum_l \Pr[X = x|G = l]\Pr[G = l]}$$

- 事後確率の大小比較のため対数比 (log-ratio) を見る.

$$\begin{aligned}
\log \frac{\Pr[k \mid X = x]}{\Pr[l \mid X = x]} &= \log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\mu_k + \mu_l)^{\mathrm{T}}\Sigma^{-1}(\mu_k - \mu_l) + x^{\mathrm{T}}\Sigma^{-1}(\mu_k - \mu_l) \\
&= (\log \pi_k - \frac{1}{2}\mu_k^{\mathrm{T}}\Sigma^{-1}\mu_k + x^{\mathrm{T}}\Sigma^{-1}\mu_k) \\
&\quad -(\log \pi_l - \frac{1}{2}\mu_l^{\mathrm{T}}\Sigma^{-1}\mu_l + x^{\mathrm{T}}\Sigma^{-1}\mu_l)
\end{aligned}$$

- 結局, 点 x がクラス k である度合い (discriminant score) は以下を評価すれば良い.

$$\delta_k(x) = x^{\mathrm{T}}\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^{\mathrm{T}}\Sigma^{-1}\mu_k + \log \pi_k$$

Diagonal LDA - 線型判別法の強烈な正則化バージョン

- 基本的なコンセプトは, 前述の LDA と同じ. 以下の条件を追加する.

$$\Sigma = \mathrm{diag}(s_1, s_2, \ldots, s_p) \quad \text{(対角行列)}$$

- すると, discriminant score は, (クラスに依らない定数 $-x^{\mathrm{T}}\Sigma^{-1}x$ を足して 2 倍することで,) 以下になる.

$$\delta_k(x) = -\sum_{j=1}^{p} \frac{(x_j - \bar{x}_{kj})^2}{s_j^2} + 2\log \pi_k$$

- この discriminant score を使って, 以下のルールで分類する.

$$C(x) = \arg\max_l \delta_l(x)$$

Diagonal LDA - 線型判別法の強烈な正則化バージョン

Diagonal LDA について何点か補足.

- discriminant score は距離に見える.
  → Diagonal LDA は, 適当な標準化をしたデータにおける nearest centroid 法のようにも見える.
- 変数間の共分散が 0 という仮定を, 独立律 (independent rule) ともいう.
- 高次元の時には, effective なことが多いらしい.
- この方法の欠点の一つは, 特徴量選択 (feature selection) ができないこと. 高次元の入力の時には, 一部の変数を選び出せる方法を使いたい.
  →もっと正則化の条件を強めるとパフォーマンスがさらに上がるらしい.
- 次は, 特徴量選択が行われるような正則化の条件をかけたバージョンを考えます.

Nearest Shrunken Centroids

前出の Diagonal LDA = Nearest Centroid 法の centroid を縮小 (shrinkage) させることで, 特徴量選択を行えるようにする.

- 基本的な計算は, Diagonal LDA と一緒.
- discriminant score の計算に使う centroid を単純な平均 $\bar{x}_{kj}$ から変える.
- まず, あるパラメータ X_j のクラス k 内での平均 $\bar{x}_{kj}$ と全体での平均 $\bar{x}_j$ の差を標準化する.

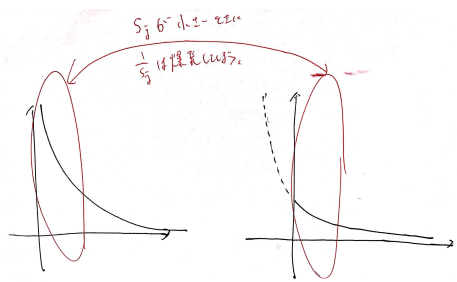
$$d_{kj} = \frac{\bar{x}_{kj} - \bar{x}_j}{m_k(s_j + s_0 b)}$$

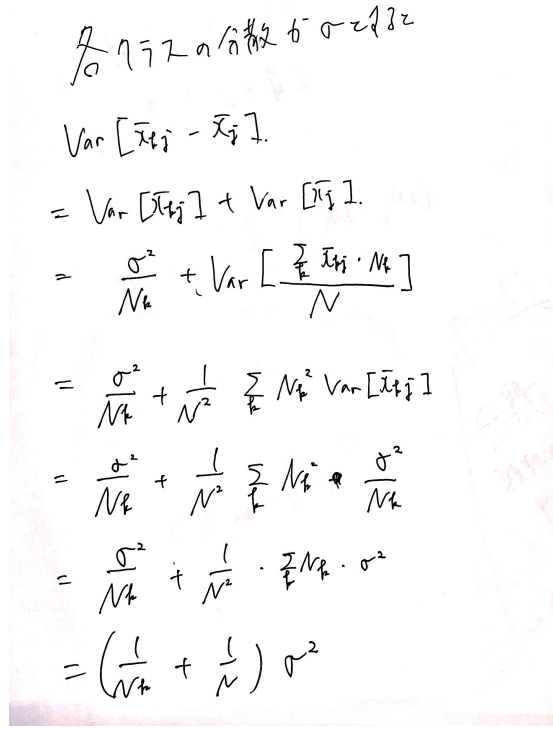
ただし, 各項は以下.

$m_k = \frac{1}{N_k} - \frac{1}{N}$: 疑問点

s_0 = 小さな定数

s_jが小さい時に d_{kj}が大きくなりすぎないように





Nearest Shrunken Centroids

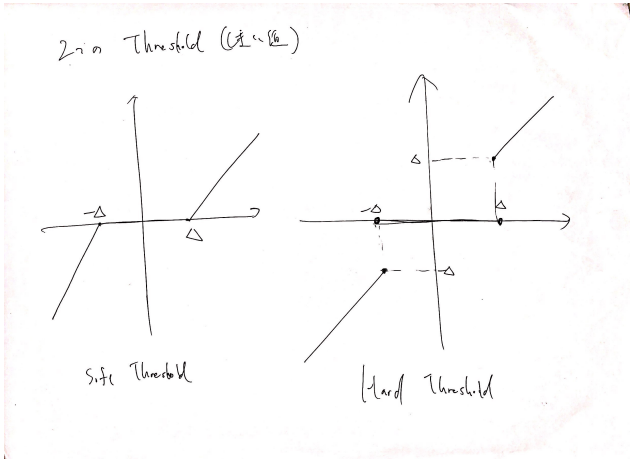
- この標準化された距離を soft-threshold

$$d'_{kj} = \mathrm{sign}(d_{kj})(|d_{kj}| - \Delta)$$

もしくは hard-threshold

$$d'_{kj} = d_{kj} I(|d_{kj}| \geq \Delta)$$

で縮小させる.



- すなわち, 以下.

$$\bar{x}'_{kj} = \bar{x}_j + m_k(s_j + s_0)d'_{jk}$$

寄与の小さい特徴量を無視するようになっている.

- Diagonal LDA の discriminant score の $\bar{x}_{kj}$ の代わりに, $\bar{x}'_{kj}$ を使えば, NSC の完成.

Section 3

2 次で正則化した線型分類

p.654-

Regularized Discriminant Analysis

- 判別分析の正則化を考える.
- 以前見た正則化は, LDA と QDA でバランスを取るために, 分散行列を以下で計算した.

$$\hat{\Sigma}_k(\alpha) = \alpha\hat{\Sigma}_k + (1-\alpha)\hat{\Sigma}$$

- 今回は違うバージョン. 対角行列に近づけようとする.

$$\hat{\Sigma}(\gamma) = \gamma\hat{\Sigma} + (1-\gamma)\mathrm{diag}(\hat{\Sigma})$$

- 上記の正則化で, $\gamma = 0$ のときは, Diagonal LDA, すなわち, 縮小のない NSC と同じ.
- ridge 回帰が分散共分散行列を対角行列に近づけようとするのと, 似ている.

$$\hat{\beta}^{\mathrm{ridge}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$$

- 線型判別を, 各クラスに数値を割り当てた線型回帰と思うと, ridge 回帰との関連をより正確に記述できるそう.

ロジスティック回帰

- 以下の式を使ってきた.

$$\Pr[G = k \mid X = x] = \frac{\exp(\beta_{k0} + x^{\mathrm{T}}\beta_k)}{\sum_l \exp(\beta_{l0} + x^{\mathrm{T}}\beta_l)}$$

- 対数尤度に正則化項を加えた以下を最大化する.

$$\sum_{i=1}^{N} \log \Pr[g_i \mid x_i] - \frac{\lambda}{2} \sum_{k=1}^{K} ||\beta_k||^2$$

- 1 個目の定式化だと, over-parametrized だけど, 正則化項のおかげでいい感じ.
- 定数項のみいい感じじゃないので, 適当に条件加えよう.

$$\frac{e^3}{e^1+e^2+e^3} = \frac{e^3 \times e^{-2}}{(e^1+e^2+e^3)\times e^{-2}}$$

$$= \frac{e^1}{e^{-1}+e^0+e^1}$$

Logistic 回帰の redundancy

ロジスティック回帰

- 前述の最大化問題は凸なので, Newton 法とかで解ける.
- separable なデータに対して, $\lambda \to 0$ とすると, マージン最大化と同じ結果になる.(?)
  → SVM もうまく関連付けられそう.

Support Vector Classifier

- 前に出てきた話. マージン最大化.
- $p \gg N$ のとき, ほぼ確で線型分離可能なので, attractive.
- 正則化しなくても有効. 頑張って正則化しても, 正則化なしと同程度のパフォーマンスのことが多い.
- 多数のクラスへの分類への応用方法を 2 つ紹介.
  - one versus one (ovo)
    全ての 2 つのクラスの組み合わせ ($K(K-1)/2$ 通り) 全てについて, SVM で分類する.
    点 x について, 上記の分類全ての結果, 最も多く分類されるクラスを推定値とする.
  - one versus all (ova)
    各クラスとそのクラス以外に分けて SVM で分類する.
    教会からの符号付き距離 (confidence) が最も大きいクラスを推定値とする.

  ovo と ova は SVC に限らず使える手法.
- 正則化ロジスティック回帰と近しい結果を返す.

特徴量選択

- p が大きい時, 特徴量選択は重要. 解釈可能性のため.
- DLDA, LR, SVC は, アルゴリズム内に特徴量選択の機能を含まない. (二次正則化のため)
  →外付けの特徴量選択手法が提案されている.
- 例. Recursive Feature Elimination.
  重みの小さい特徴量から無視していく.
  →あまり上手くいかないらしい.
  (“ we do not have an explanation for this behavior.”)
- Kernel 法使って外れ値に強くさせることも可能.

計算の工夫

$p \gg N$ で二次正則化の線形モデルを考えた時に使える計算の工夫について.
まずは最も簡単な ridge 回帰について.

- 以下の最小化を考える.

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

- これは, 解けて解は以下.

$$\hat{\beta} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda \mathbf{I})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$$

- ここで, 行列 $\mathbf{X}$ の特異値分解を考える.

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathrm{T}} = \mathbf{R}\mathbf{V}^{\mathrm{T}}$$

- すると, 上記の解は以下になる.

$$\hat{\beta} = \mathbf{V}(\mathbf{R}^{\mathrm{T}}\mathbf{R} + \lambda \mathbf{I})^{-1}\mathbf{R}^{\mathrm{T}}\mathbf{y}$$

- $\mathbf{R}$ を入力, $\mathbf{y}$ を出力としたときの ridge 回帰の結果を $\hat{\theta}$ とすると, $\hat{\beta} = \mathbf{V}\hat{\theta}$ となっている.
- 結論, p 次元データの回帰の計算を N 次元データの計算に還元できる.

計算の工夫

ridge 回帰に関する考察は，より一般的な話につながる．

- 線形モデルを考える．
$$Y = f(X) = \beta_0 + X^{\mathrm{T}}\beta$$
- 任意の損失関数を取る．
$$\sum_{i=1}^{N} L(y_i, f(x_i))$$
- 入力変数のデータの行列 $\mathbf{X}$ を特異値分解して，$N \times N$ 行列 $\mathbf{R}$ を作る．
$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathrm{T}} = \mathbf{R}\mathbf{V}^{\mathrm{T}}$$
- 以下の最小化を考える．
$$(\hat{\beta}_0, \hat{\beta}) = \arg\min_{\beta_0, \beta} \sum_{i=1}^{N} L(y_i, \beta_0 + x_i^{\mathrm{T}}\beta) + \lambda\beta^{\mathrm{T}}\beta$$
$$(\hat{\theta}_0, \hat{\theta}) = \arg\min_{\theta_0, \theta} \sum_{i=1}^{N} L(y_i, \theta_0 + r_i^{\mathrm{T}}\theta) + \lambda\theta^{\mathrm{T}}\theta$$
- すると，以下が成り立つ．
$$\hat{\beta}_0 = \hat{\theta}_0, \hat{\beta} = \mathbf{V}\hat{\theta}$$

計算の工夫

- LR, LDA, SVM など広く応用可能.
- L_1 正則化の時には使えないので注意.
- λ は交差検証とかで.

Section 4

L_1 正則化線形分類

p. 661-

L_1 正則化線形分類

ここまで L_2 正則化．ここから L_1 正則化．嬉しいポイントは勝手に特徴量選択してくれること．

- 例えば，lasso 回帰．以下を最小化．

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

- L_1 正規化は勝手に特徴量選択をしてくれる．

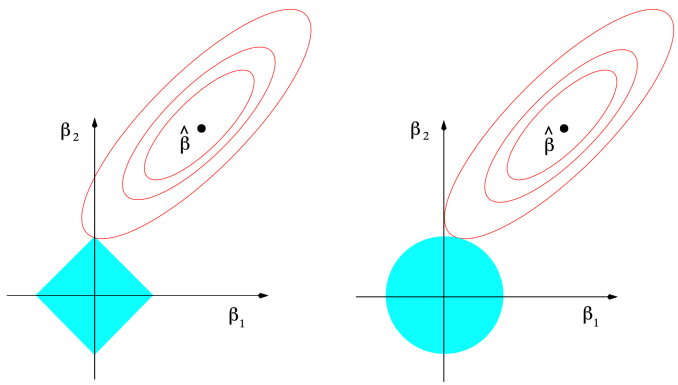


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \le t$ *and* $\beta_1^2 + \beta_2^2 \le t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

- 選ばれる特徴量は高々 N 個になる．(convex duality より．凸共役?)

Elastic Net

- L_1 正規化の困るポイント．相関の強い変数に対応できない．
  $Y = aX_1 + b, X_2 = X_1, a_1 + a_2 = a$ とすると，

$$Y = a_1 X_1 + a_2 X_2 + b$$

  となり，lasso では a_2, a_2 を特定できない．
- 回避策として，ridge と lasso の合体を考えるのが Elastic Net．正則化項は以下．

$$\sum_{j=1}^{p} (\alpha |\beta_j| + (1 - \alpha)\beta_j^2)$$

- *alpha* は大抵 pre-chosen．もちろん，交差検証でも良い．

Fused Lasso

特徴量たちが順序を持つ場合. 時系列データとか.

- 隣り合う特徴量について, 回帰係数を近づけたい.
  →以下を最小化する.

$$\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_i j \beta_j)^2 + \lambda_1 \sum_{j=1}^{p} |\beta| + \lambda_2 \sum_{j=1}^{p-1} |\beta_{j+1} - \beta_j|$$

- 隣り合う特徴量の間隔が一定でないなら, それを考慮して以下を使う.

$$\lambda_2 \sum_{j=1}^{p-1} \frac{|\beta_{j+1} - \beta_j|}{|t_{j+1} - t_j|}$$

特徴量が明確でない場合 - String Kernel

分類したいけど, 特徴量がわからない場合. (タンパク質の構造)
分類したいけど, あまりにも次元が大きすぎる場合. (文書の分類)
→ (非) 類似度が測れればいろいろできる.

タンパク質の構造による分類の例

- タンパク質は 20 種類のアミノ酸からなる列. 文字列で表せる.

IPTSALVKETLALLSTHRTLLIANETLRIPVPVHKNHQLCTEEIFQGIGTLESQTVQGGTV
ERLFKNLSLIKKYIDGQKKKCGEERRRVNQFLDY**LQE**FLGVMNTEWI

PHRRDLCSRSIWLARKIRSDLTALTESYVKHQGLWSELTEAER**LQE**NLQAYRTFHVLLA
RLLEDQQVHFTPTEGDFHQAIHTLLLQVAAFAYQIEELMILLEYKIPRNEADGMLFEKK
LWGLKV**LQE**LSQWTVRSIHDLRFISSHQTGIP

- 部分文字列は良い特徴量になる. タンパク質 x と m 文字のアミノ酸列 a に対して, $\varphi_a(x)$ を x に含まれる a の数とする.
- 特徴量を

$$\Phi_m(x) = \{\varphi_a(x)\}_{a \in \mathcal{A}_m}$$

で定める.

- 類似度を内積として,

$$K_m(x_1, x_2) = \langle \Phi_m(x_1), \Phi_m(x_2) \rangle$$

で定める. (ちょっと RKHS 感出てきた.)

- あとは, SVM とかすれば良い.

Kernelization

- 様々な手法を “カーネル化” することができる. 簡単にいうと, データそのものの値ではなく, 各データの組の内積 (類似度) や距離 (非類似度) に注目しようという話.
  RKHS 感が出てきている.
- 内積や距離の情報から計算できるモデルの例.
  SVM, 最近傍法, ロジスティック回帰, 主成分.

k-NN の Kernelization

内積の情報をもとに K 近傍を見つけるために, 以下を用いれば良い.

$$||x_i - x_{i'}||^2 = \langle x_i, x_i \rangle + \langle x_{i'}, x_{i'} \rangle - 2\langle x_i, x_{i'} \rangle$$

Nearest Centroid の Kernelization

K-NN と似たイメージで, Nearest Centroid 法もカーネル化できる.

$$||x - \bar{x}_k||^2 = \langle x, x \rangle - \frac{2}{N_k} \sum_{g_i = k} \langle x, x_i \rangle + \frac{1}{N_k^2} \sum_{g_i = k} \sum_{g_j = k} \langle x_i, x_j \rangle$$

ロジスティック回帰の Kernelization

https://www.ism.ac.jp/~fukumizu/svm-ism.pdf

- ロジスティック回帰の Lagrange Dual Form は以下.

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \langle x_i, x_{i'} \rangle$$

PCA の Kernelization

- データ行列 $\mathbf{X}$ の特異値分解を $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathrm{T}}$ とすると, 主成分は以下.

$$\mathbf{Z} = \mathbf{U}\mathbf{D}$$

- ここから, 簡単に, 主成分をデータの分散共分散行列の対角化として得られることがわかる. (略)
- 分散共分散行列

$$\hat{\mathbf{K}} = \left(\langle x_i - \bar{x}, x_{i'} - \bar{x} \rangle\right)_{i,i'}$$

なので, Kernelization できている.

Kernelization でできないこと

最適化を内積のみを用いて表すのは便利．じゃあできないことは?

- 変数の標準化
- 各変数の寄与の推定
- 効果のない変数と効果のある変数の見分け

総じて，一つ一つの入力変数に注目は出来ない，というのが弱み．

Section 5

余談 - 再生核ヒルベルト空間
(Reproducing Kernel Hilbert Space, RKHS)

今こそ RKHS - カーネル法 (Kernel Method)

- まずはカーネル法について, やりたいこととざっくり全体像の整理.
  - 線形に分類できないもの (同心円が典型例) をうまく分類したい.
  - 変数変換してあげると, 線形の手法を使える.

$$(x, y) \mapsto (x^2, y^2)$$

  - 高次元に写せば, ほぼ確実に線形分類可能にできる.

$$(x, y) \mapsto (x, y, x^2, xy, y^2)$$

  - 一方, 高次元に写すと計算量が増える. 困る.
    →高次元でも計算が楽なものはないのか...
  - 数学の話. 再生核ヒルベルト空間 (RKHS) という数学的対象とカーネル関数というある条件を満たす関数が 1 対 1 対応する.
    - 再生核ヒルベルト空間は, 高次元に写す先の空間に相当.
    - カーネル関数は, 高次元空間における内積を直接計算する関数に相当.
  - RKHS は内積計算が楽!内積で計算できるモデルならいい感じに使える!
    (SVM, CPA, etc... 先述のものたちとか)
  - 例えば, 以下を考える.

$$\Phi\colon \mathbb{R}^2 \to \mathbb{R}^3, \Phi(x_1, x_2) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

    対応する RHKS は $\mathbb{R}^3$. カーネル関数は,

$$\begin{aligned} K(x, y) &= \langle (x_1^2, x_2^2, \sqrt{2}x_1x_2), (y_1^2, y_2^2, \sqrt{2}y_1y_2) \rangle \\ &= x_1^2y_1^2 + x_2^2y_2^2 + 2x_1x_2y_1y_2 \\ &= (x_1y_1 + x_2y_2)^2 \\ &= \langle x, y \rangle^2 \end{aligned}$$

Section 6

Supervised Principal Component
p. 674-

Supervised Principal Component

目的変数をよく説明できて, 分散が大きいような線形結合成分を取りたい.

Algorithm 18.1 *Supervised Principal Components.*

1. Compute the standardized univariate regression coefficients for the outcome as a function of each feature separately.
2. For each value of the threshold θ from the list $0 \leq \theta_1 < \theta_2 < \cdots < \theta_K$:
   (a) Form a reduced data matrix consisting of only those features whose univariate coefficient exceeds θ in absolute value, and compute the first m principal components of this matrix.
   (b) Use these principal components in a regression model to predict the outcome.
3. Pick θ (and m) by cross-validation.

Step. 1　普通に 1 変数線形回帰. 入力変数と目的変数の関係の強さを調べる.

Step. 2(a)　変数選択からの主成分分析.

Step. 2(b)　主変数での多変数回帰.

Step. 3　交差検証.

Supervised PCA と Partial Least Squares

3.5.2 で見た Partial Least Squares も, 説明力のある主成分を見つけようとする手法.

Algorithm 3.3 *Partial Least Squares.*

1. Standardize each $\mathbf{x}_j$ to have mean zero and variance one. Set $\hat{\mathbf{y}}^{(0)} = \bar{y}\mathbf{1}$, and $\mathbf{x}_j^{(0)} = \mathbf{x}_j,\ j = 1, \ldots, p$.
2. For $m = 1, 2, \ldots, p$
   (a) $\mathbf{z}_m = \sum_{j=1}^{p} \hat{\varphi}_{mj}\mathbf{x}_j^{(m-1)}$, where $\hat{\varphi}_{mj} = \langle \mathbf{x}_j^{(m-1)}, \mathbf{y} \rangle$.
   (b) $\hat{\theta}_m = \langle \mathbf{z}_m, \mathbf{y} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle$.
   (c) $\hat{\mathbf{y}}^{(m)} = \hat{\mathbf{y}}^{(m-1)} + \hat{\theta}_m \mathbf{z}_m$.
   (d) Orthogonalize each $\mathbf{x}_j^{(m-1)}$ with respect to $\mathbf{z}_m$: $\mathbf{x}_j^{(m)} = \mathbf{x}_j^{(m-1)} - [\langle \mathbf{z}_m, \mathbf{x}_j^{(m-1)} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle]\mathbf{z}_m,\ j = 1, 2, \ldots, p$.
3. Output the sequence of fitted vectors $\{\hat{\mathbf{y}}^{(m)}\}_1^p$. Since the $\{\mathbf{z}_\ell\}_1^m$ are linear in the original $\mathbf{x}_j$, so is $\hat{\mathbf{y}}^{(m)} = \mathbf{X}\hat{\beta}^{\mathrm{pls}}(m)$. These linear coefficients can be recovered from the sequence of PLS transformations.

(アルゴリズム見てもイメージ湧かないかも)
両者の違いは, SPCA は要らない特徴量を切り捨てるが, PLS は downweight するだけという点.

Pre-Conditioning for Feature Selection

SPCA の問題.

- 必ずしも sparce なモデルにならない.
- 選ばれなかった特徴量の中に主成分と相関の強いものがあるかも.
- 相関の強い特徴量たちがみんな選ばれる. (冗長な変数選択)

→ lasso で 1 個目は解決できるが, 単なる lasso では性能が低い.
→ predictioned lasso

Predictioned Lasso

1. 通常の SPCA で出力変数 Y の推定値 $\hat{Y}$ を計算する.
2. X を入力変数, $\hat{Y}$ を出力変数として lasso 回帰する.

アルゴリズムの各ステップの意図は以下。

Step. 1 Denoising the Outcome.
出力変数の生データに含まれるノイズをなくす。

Step. 2 Lasso で sparce な回帰をする。

パフォーマンス良い。