機械学習入門

経済学部 BX584

正則化 regularization

機械学習手法の評価の仕方

- データ集合を3つに分ける！
  - 訓練データ
    - 他よりも多めにとる。全体の６～８割。
    - 正解が分かっているとしてよい。
      - 正解をターゲット（望ましい出力値）として使って、機械学習の手法を動かす。
  - 検証データ
    - 全体の１～２割。
    - 予測結果をこれで評価して、ハイパーパラメータをチューニングする。
      - 予測結果と照らし合わせるときに、正解を使う。
  - テストデータ
    - 訓練データ、検証データ以外の残り。
    - 手法自体の最終的な性能を評価するのに使う。（模擬的な本番運用）

予測モデルの良し悪しをどう決める？

- 訓練データ上で分類or回帰による予測が良くても不十分
- 未知データ(unseen data)上でも良い予測ができないとダメ
- これを「汎化性能 generalization performance」と呼ぶ
  - 汎化性能が良い＝未知データでも良い予測ができる

cf. 予測が良い＝予測誤差(error)が小さい

過学習(overfitting)

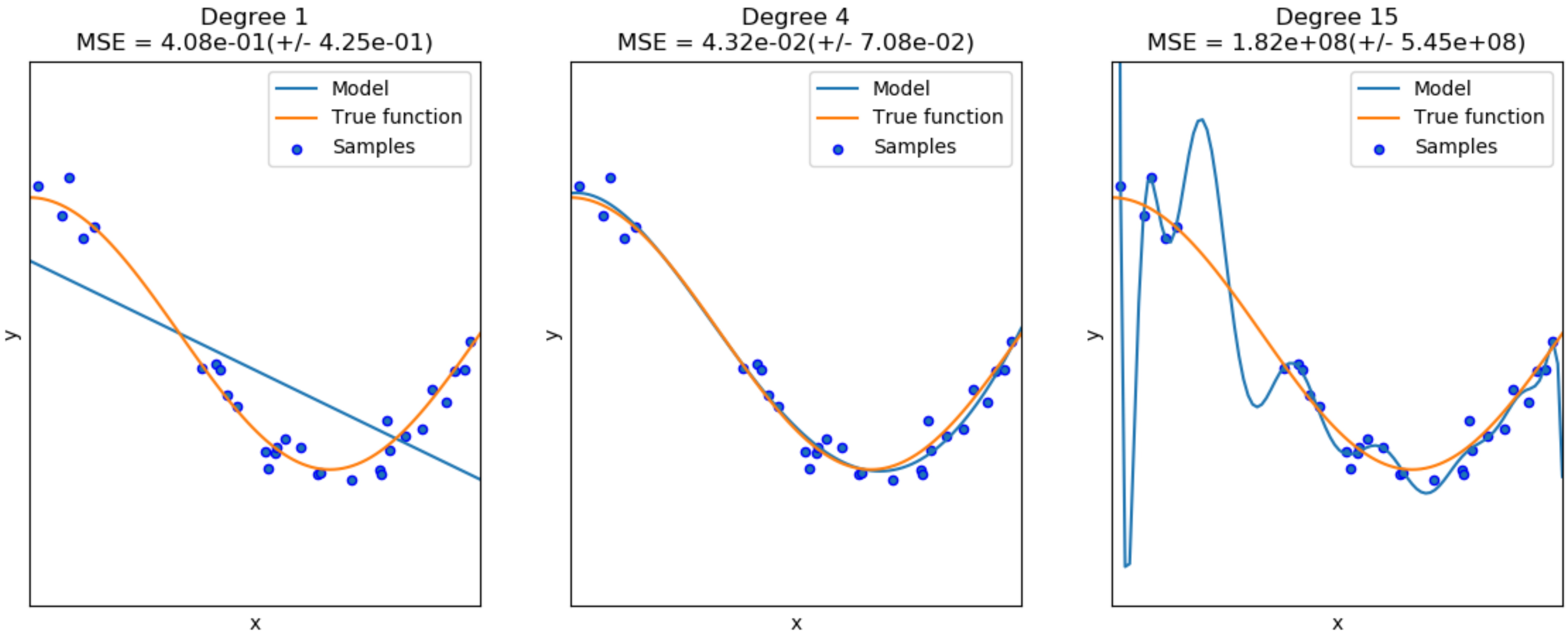
- 訓練データ上でのエラーと未知データ上でのエラーにギャップ
  - 未知データ上でのエラーのほうが大きい、という意味。
    - 訓練データでは予測がうまくいっているのに、ということ。
- 関数を選ぶ範囲が広すぎるときに過学習を起こしやすい
  - 関数を選ぶ範囲が広すぎる ＝ 予測モデルが複雑すぎる
  - 関数を選ぶ範囲を狭めると、過学習を防げることがある

回帰におけるモデルの複雑さ

- ここまで説明した回帰は線型モデル＝1次式で書けるモデル
  - 説明変数がD個の場合の式:

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = b + w_1 x_1 + \cdots + w_D x_D$$

- 2次以上の項を含ませることもできるが・・・
  - 式の形がもっと複雑になる（パラメータの個数も増える）
- 複雑な予測モデルほど良いか？というと…実際はそうでもない
  - なぜか？ → 過学習を起こしやすくなるため



http://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html

複雑なモデルの問題点

- 複雑なモデルは訓練データにぴったりフィットしやすい
- 訓練データとそれ以外のデータは異なるデータ
- 訓練データに<u>フィットしすぎた</u>モデルが未知データで予測を間違う
    - 訓練データだけに特殊な情報も学習してしまうため

      例）データ収集時のノイズ
- 予測モデルは訓練データと未知データとに共通する「何か」を学習しなければならない

正則化 (regularization)

- 正則化とは
  - 予測モデルがあまり複雑にならないようにする方法
- 正則化がない場合
  - 予測と目標値とのズレを最小化するだけ
- 正則化がある場合
  - ズレ（損失関数）だけでなく、<u>モデルの複雑さも同時に最小化</u>する
  - モデルの複雑さはどうやって表す？

線形回帰の場合の正則化

- 予測モデルに線形関数を使う
  - 線形なので式の次数は1次のまま
- 線形関数の「複雑さ」とは？
  - パラメータの絶対値が大きくならないようにする→関数を選ぶ範囲を狭める
  - パラメータの二乗和や絶対値の和があまり大きくならないようにする
    - 二乗和を小さくする＝リッジ回帰
    - 絶対値の和を小さくする＝Lasso

リッジ回帰(ridge regression)

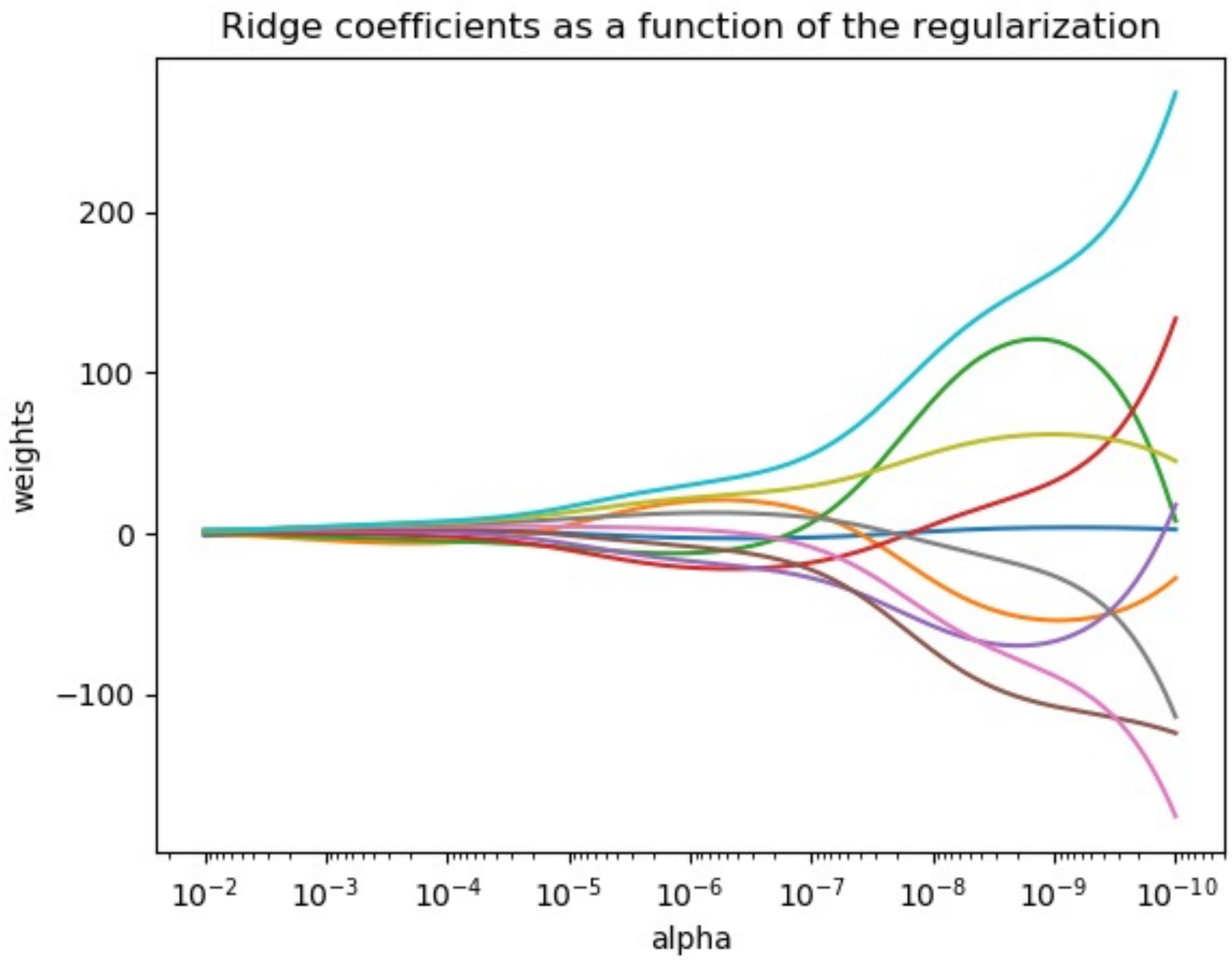
- リッジ回帰で最小化する関数
  - （誤差の二乗和）＋α×（パラメータの二乗和）
- α＝0にすると・・・
  - 通常の最小二乗法と同じ
- α＝無限大にすると・・・
  - すべてのパラメータがゼロになる
- αが小さいほどパラメータが自由に動ける（複雑になる）

リッジ回帰における正則化

- L2ノルムを使って係数が大きくならないようにする
  - 正則化項と損失関数の和を最小化する

$$\min_{w}\left[\underbrace{\alpha\|\boldsymbol{w}\|_2^2}_{\boxed{\text{正則化項}}}+\sum_{i=1}^{N}\{y_i-f(\boldsymbol{x}_i)\}^2\right]$$

https://scikit-learn.org/stable/modules/linear_model.html#ridge-regression



http://scikit-learn.org/stable/auto_examples/linear_model/plot_ridge_path.html

Lasso (least absolute shrinkage and selection operator)

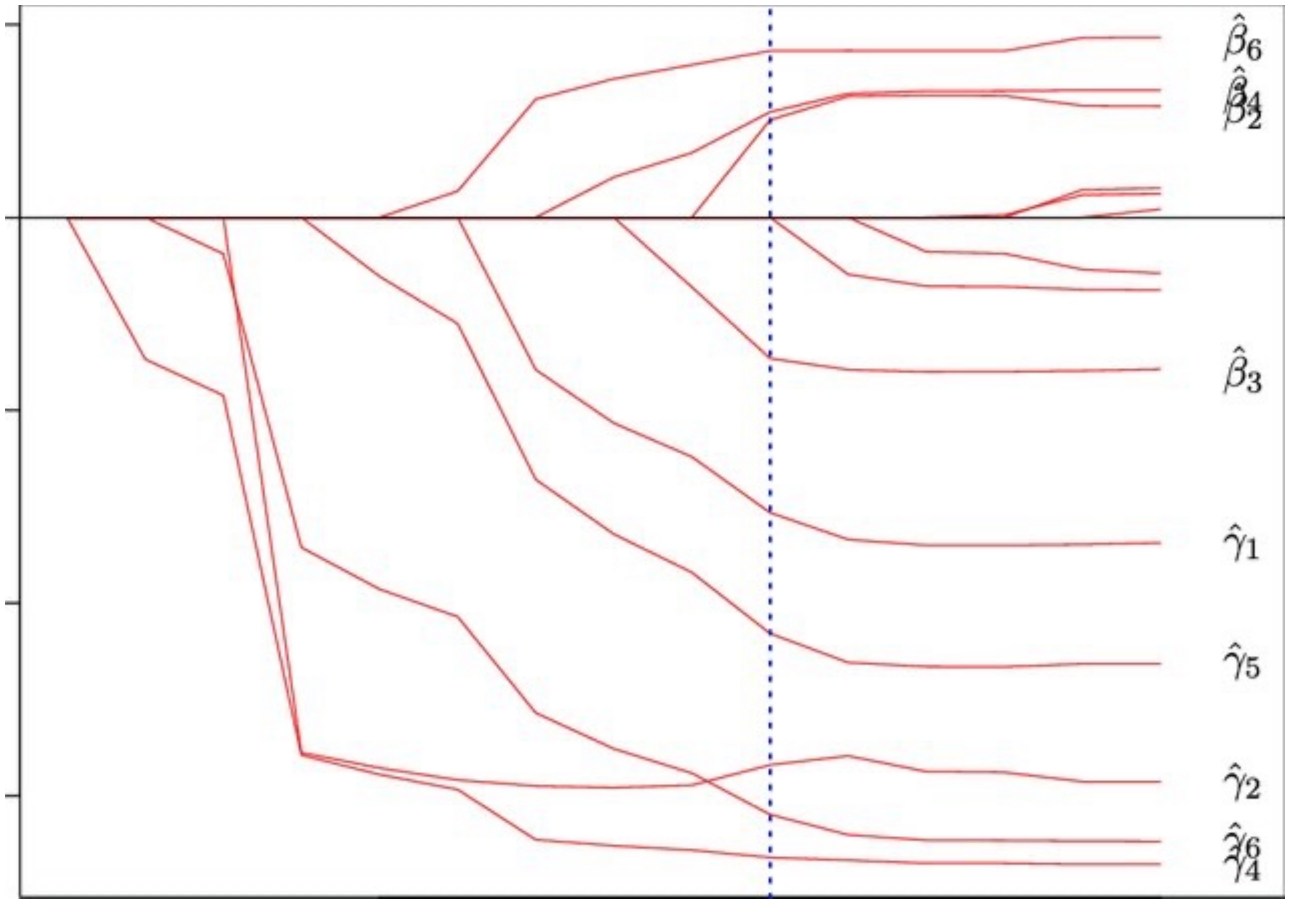
- Lassoで最小化する関数
  - （誤差の二乗和）＋α×（パラメータの絶対値の和）
- α＝0にすると･･･
  - 通常の最小二乗法と同じ
- α＝無限大にすると･･･
  - すべてのパラメータがゼロになる
- αが小さいほどパラメータが自由に動ける（複雑になる）

Lassoにおける正則化

- L1ノルムを使って係数が大きくならないようにする
  - 正則化項と損失関数の和を最小にする

$$\min_{w}\left[\underbrace{\alpha\|\boldsymbol{w}\|_1}_{\text{正則化項}}+\frac{1}{2N}\sum_{i=1}^{N}\{y_i-f(\boldsymbol{x}_i)\}^2\right]$$

https://scikit-learn.org/stable/modules/linear_model.html#lasso



https://www.researchgate.net/publication/258277692_Multiple-Inflation_Poisson_Model_with_L1_Regularization/figures?lo=1

リッジ回帰とLassoはどう違う？

- リッジ回帰
  - パラメータが全体的にゼロに近づいていく

- Lasso
  - ゼロになるパラメータが増えていく