

# 統計的モデリング基礎⑦ ～モデルの選択～

鹿島久嗣  
(情報学科 計算機科学コース)

# モデルの選択と評価： 評価指標と性能検証の枠組み

---

- モデルの予測精度を測る指標
- 精度計測の枠組み：交差検証
- 交差検証の応用：モデルスタッキング

## モデルの予測精度の検証：

### 判別（質的従属変数予測）の予測精度をどう測るか？

- 回帰（量的従属変数）の予測精度は二乗誤差で測る
  - あるいは絶対誤差、あるいはアプリケーション依存
- 判別（質的従属変数）の予測精度はどのように測るか
  - 予測の誤り回数でよさそうだが...
  - ロジスティック回帰モデルは $Y = 1$ となる確率：

$$P(Y = 1|\mathbf{x}, \mathbf{w}) = \frac{1}{1 + \exp(-\mathbf{w}^\top \mathbf{x})} = \sigma(\mathbf{w}^\top \mathbf{x})$$

- 閾値を0.5として $P(Y = 1|\mathbf{x}, \mathbf{w}) \geq 0.5$ かどうかで決める？
- 殆どのデータが $Y = 0$ だとしたら（稀な疾患の診断など）

# 混同行列： 予測の正解・不正解をまとめた表

- 推定後のモデル（例えばロジスティック回帰）は  $Y = 1$  となりそうな程度  $f(\mathbf{x})$  を与える
- 予測時には  $f(\mathbf{x})$  がある閾値  $\tau$  より大きければ  $Y = 1$  と予測する
- 予測が決まると混同行列が決まる：

		予測	
		$Y = 1$	$Y = -1$
真の値	$Y = 1$	真陽性予測数☺	偽陰性予測数
	$Y = -1$	偽陽性予測数	真陰性予測数☺

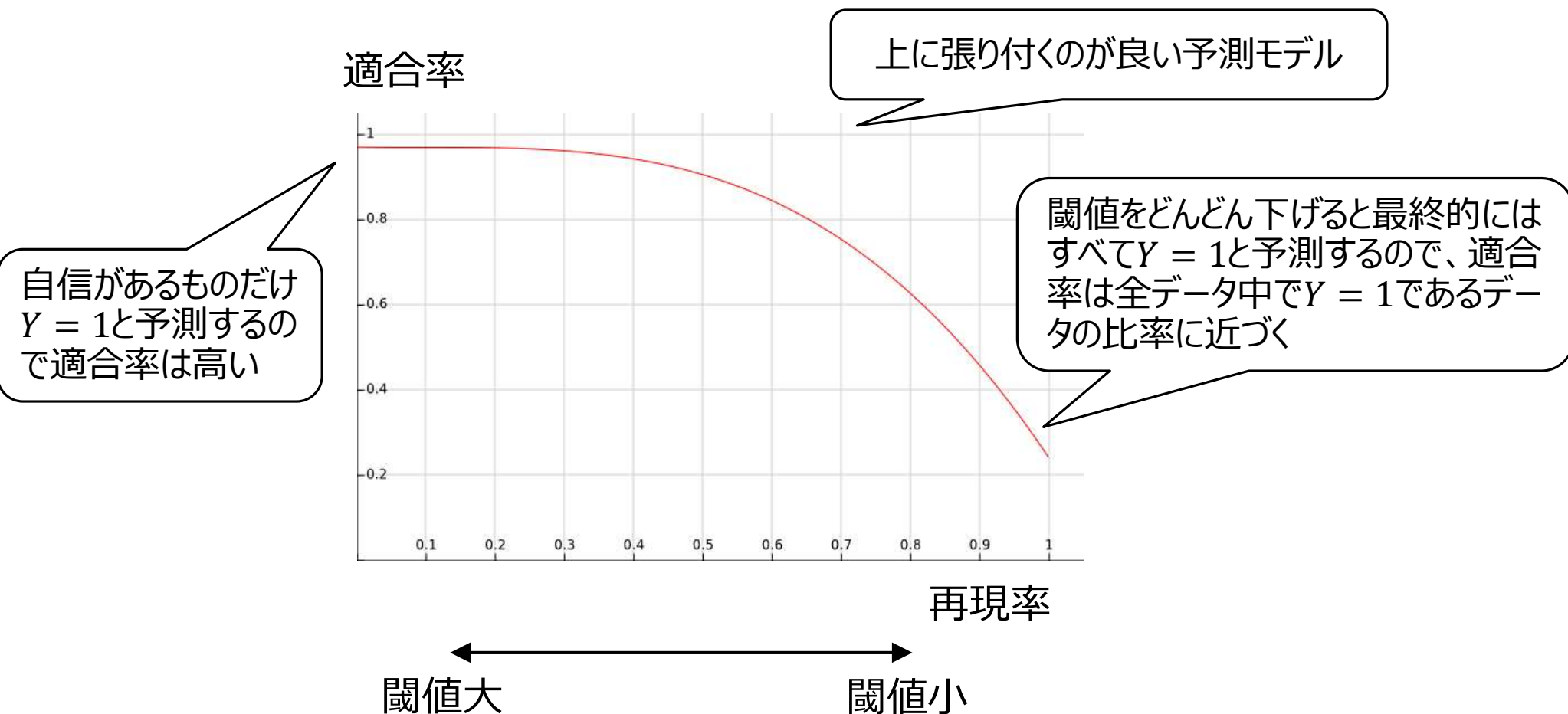
☺： 予測が正しい

# 基本的な予測精度の指標： 正解率、適合率、再現率、F値

- 正解率：
$$\frac{\text{真陽性予測数} + \text{真陰性予測数}}{\text{全予測数}}$$
- 適合率、再現率、F値：
  - 適合率 = 
$$\frac{\text{真陽性予測数}}{\text{陽性予測数}}$$
  - 再現率 = 
$$\frac{\text{真陽性予測数}}{\text{真陽性予測数} + \text{偽陰性予測数}}$$
  - F値 = 
$$\frac{\text{適合率} \cdot \text{再現率}}{\text{適合率} + \text{再現率}}$$
 : 適合率と再現率の調和平均
- 注意：これらは閾値を変えると変わる

# 閾値を変えながら見る： 適合率-再現率（PR） 曲線

- 予測の閾値を変えながら適合率-再現率 をプロットしたもの



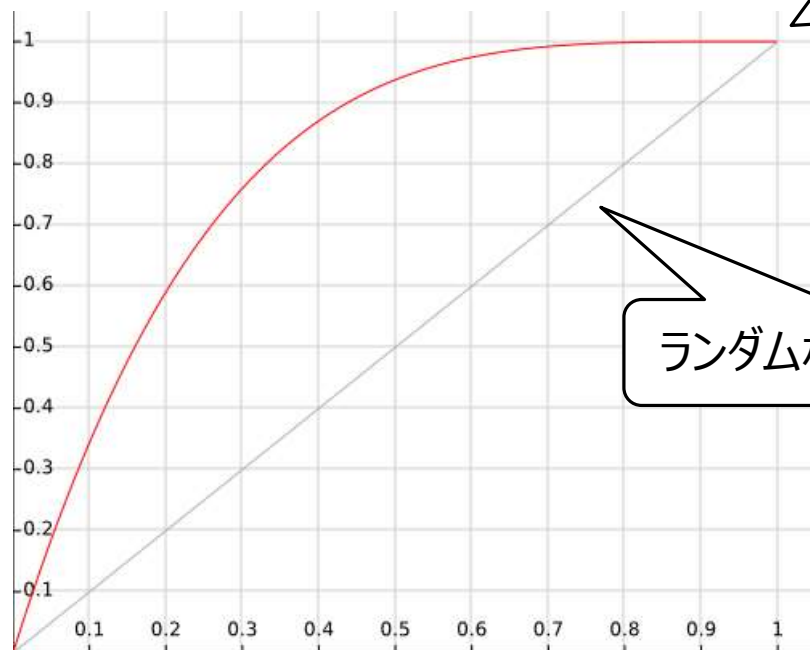
# 閾値を変えながら見る： ROC曲線

- 受信者操作特性（ROC）曲線：閾値を変えながら真陽性予測数（＝再現率）と偽陽性予測数をプロットしたもの

真陽性予測数（＝再現率）

閾値を下げれば真陽性予測の数を増やせるが、偽陽性予測も増える

左上に張り付くのが  
良い予測モデル



ランダムな予測の場合

偽陽性予測数

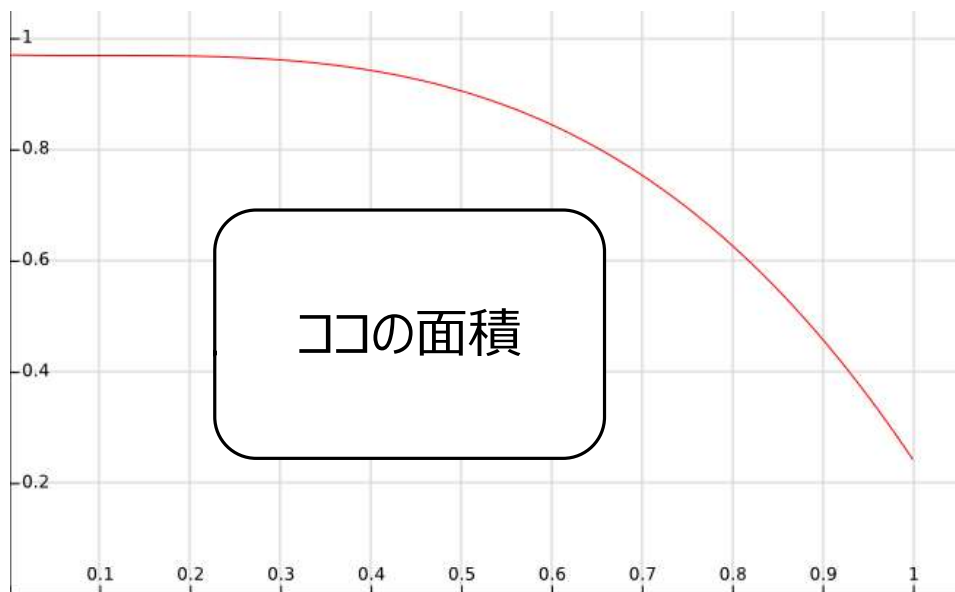
← 閾値大      閾値小 →

# 閾値によらない指標： 曲線の下面積

- PR曲線の下面積（PR-AUC）
- ROC曲線の下面積（ROC-AUC）

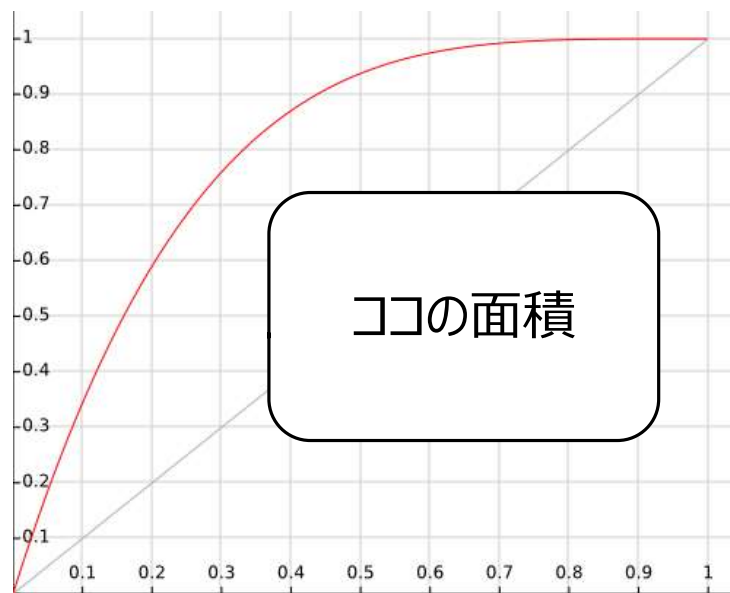
単にAUCといったら  
通常はこちら

適合率



再現率

真陽性予測数



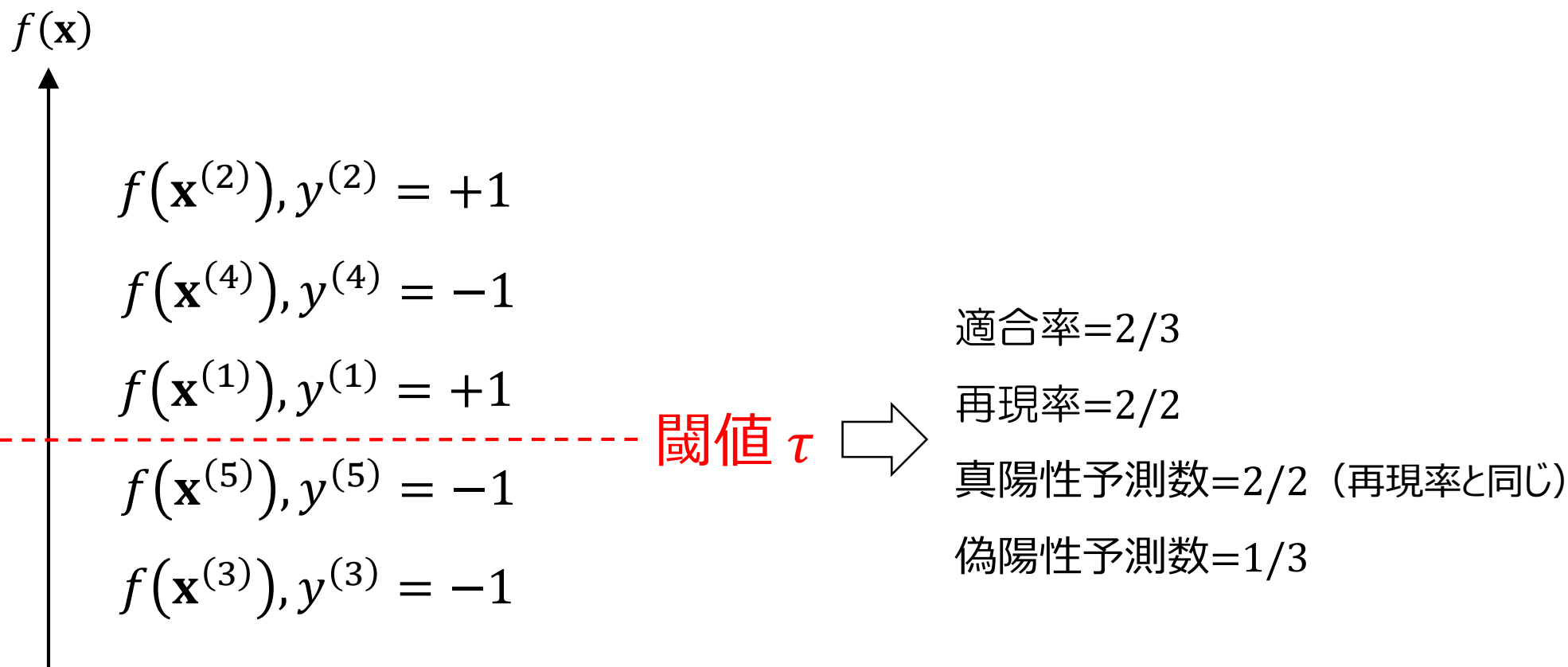
偽陽性予測数



# AUC等の計算量：

PR・ROC曲線、AUCを求める計算量＝データ整列の計算量

- PR曲線、ROC曲線、これらのAUCを求める計算量は $f(\mathbf{x})$ で整列するコスト ( $O(n \log n)$ )



# ROC-AUCの意味： 順序付けの精度を表す

- ROC-AUC :  $y^{(i)} = +1, y^{(j)} = -1$  であるすべての  $(i, j)$  の組のうち、 $f(\mathbf{x}^{(i)}) > f(\mathbf{x}^{(j)})$  となっているものの割合
  - 正しい順序で並べられているかをチェックしている（ $f$  は  $Y = 1$  である信念度合い）
- AUC=1 : 完璧な予測、AUC=0.5 : 完全にランダムな予測（AUC=0は予測を反転すれば完璧な予測）
- 先の例では  $2 \times 3 = 6$  ペアのうち5ペアの順序が保たれているので、AUC=5/6

$f(\mathbf{x})$  ↑

$f(\mathbf{x}^{(2)}), y^{(2)} = +1$
$f(\mathbf{x}^{(4)}), y^{(4)} = -1$
$f(\mathbf{x}^{(1)}), y^{(1)} = +1$
$f(\mathbf{x}^{(5)}), y^{(5)} = -1$
$f(\mathbf{x}^{(3)}), y^{(3)} = -1$

# 評価の枠組み： モデル選択と評価

- 予測モデリングにおいて実際に興味があるのは、推定した予測モデルを運用する際の、将来のデータに対する精度
  - モデル推定に用いたデータと将来のデータは異なる  
(同じメカニズムで発生しているという仮定はあるが)
- ハイパーパラメータを調整して予測精度を向上したい：
  - リッジ回帰：  $\text{minimize}_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda \|\mathbf{w}\|_2^2$
  - ハイパーパラメータはモデル推定の過程では推定されない

# 情報量基準： モデルの真の性能を見積もる基準

---

- 情報量基準：真の性能を見積もる
  - AIC： $-2(\text{対数尤度}) + 2(\text{パラメータ数})$
  - BIC： $-2(\text{対数尤度}) + 2(\text{パラメータ数}) \cdot \ln n$
- ただし、いくつかの仮定のもとで
- 以下ではより実験的な性能評価の枠組み（交差検証）を説明する

# モデル評価の大原則：

## モデル推定に使ったデータを評価に使ってはいけない

- モデルの予測精度を検証するために、モデルに推定に使用したデータを用いてはいけない
  - モデル推定に使ったデータに対するモデルの精度は、そのモデルの真の精度の推定値ではない
- データを推定用データと検証用データに分割して用いる：
  1. 推定用データを用いてモデルを推定する
  2. 推定したモデルの性能を検証用データで評価する
  - 分割はアプリケーションの文脈に合わせて行う必要がある
    - ◆ ランダムに分割、時系列順に分割、...

# モデル評価の統計的枠組み： 交差検証

- ( $K$ -分割) 交差検証：将来のモデル運用時の性能を推定するための枠組み
- 全データを、重複しない  $K$  個の集合に等分割する：
  - うち  $K - 1$  個の集合をモデル推定に用いる
  - 残りひとつの集合で評価を行う
- 検証用のデータ集合を変えると、 $K$  通りの評価が行われる（ $K$  個の評価値が得られる）
  - これらの平均をとって性能の推定値とする

# ハイパーパラメータの推定：


## 交差検証によるハイパーパラメータ推定

---

- 正則化（MAP推定）の際のハイパーパラメータ
  - ハイパーパラメータはモデル推定（の最適化問題）においては自動的に決まらない（0になってしまう）
- ( $K$ -分割) 交差検証によるハイパーパラメータ調整：
  - $K$ 個に分割されたデータのうち  $K - 1$  個を用いて、それぞれのハイパーパラメータ設定においてモデル推定を行う
  - 残りひとつの集合を用いてそれぞれのモデルの精度を測る
  - $K$ 個の評価値の平均がもっともよいハイパーパラメータを採用
    - ◆ この評価値は、モデル運用時の性能とは異なることに注意

## 二重交差検証：

### ハイパーパラメータ推定と性能評価を同時に行う

- しばしば、ハイパーパラメータ推定と、最終的に選ばれたモデルの性能の推定の両方を
- ひとつの  $K$ -分割交差検定で行ってはいけない 
  - ハイパーパラメータ推定を行った際にみたデータを評価に使ってはいけない
- 二重交差検定：
  - 外側のループでは性能評価を行う
  - 内側のループではハイパーパラメータ調整を行う
  - 計算コストが高い



# 二重交差検証の（軽量な）代用：

## “開発用データ”方式

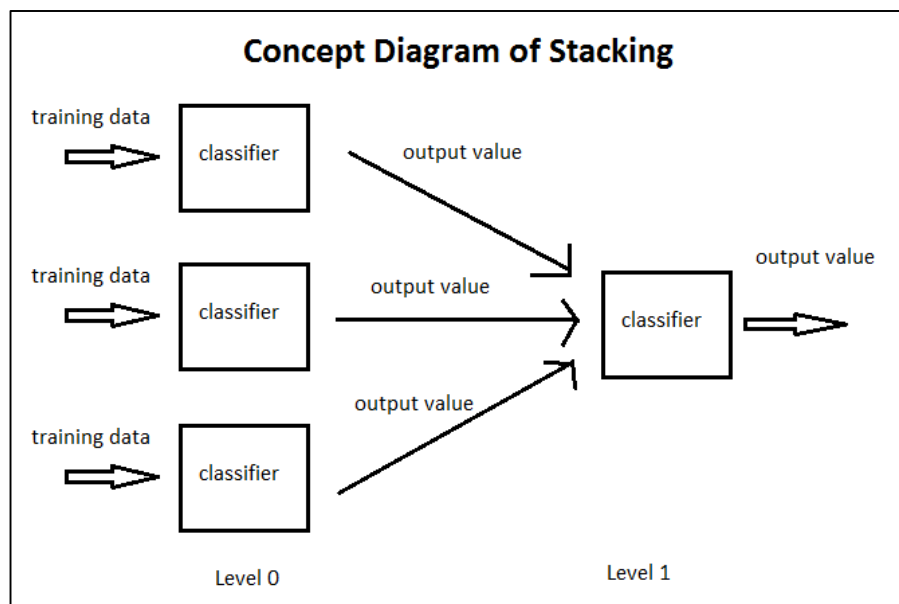
---

- 二重交差検証は計算コストが高いため、もう少し簡単な方法がほしい
- “開発用データ”方式
  - $K$ 分割したデータのうち  $K - 2$  個を推定に用いる
  - 残りのうちひとつをハイパーパラメータ調整に用いる
  - 最後のひとつを性能評価に用いる

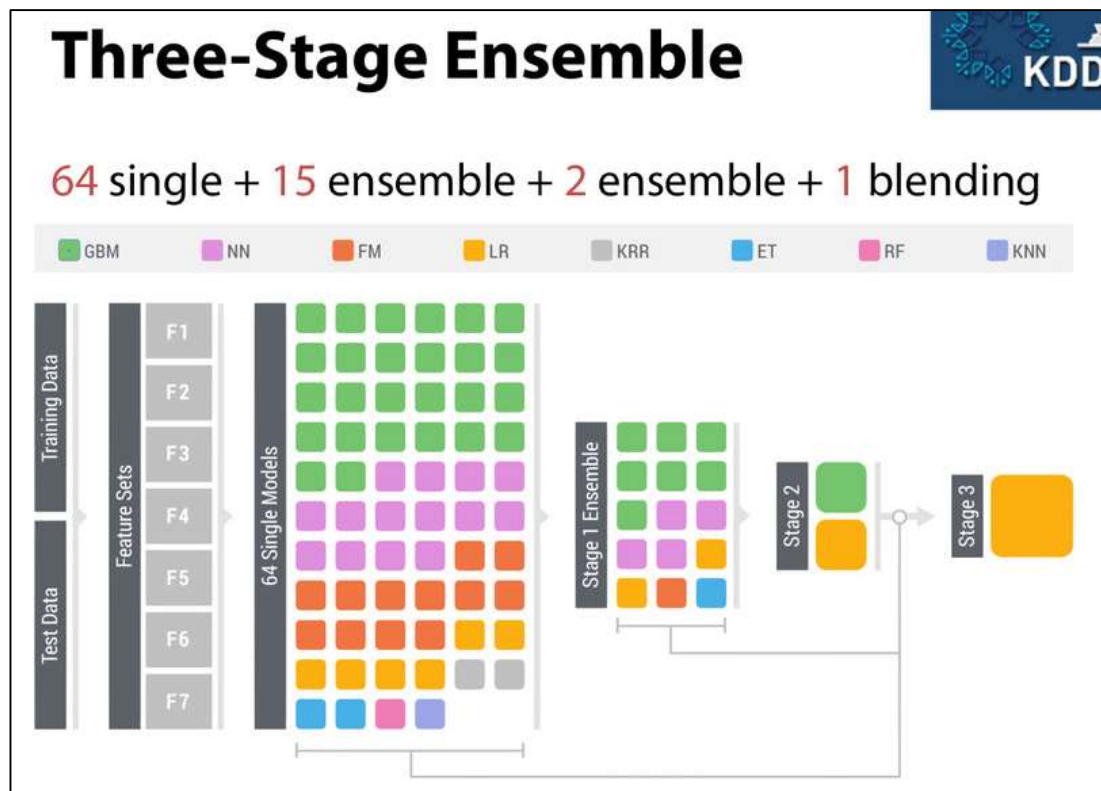
# スタッキング :

## 複数のモデルを並列・直列に積み上げる方法

- 予測モデルの出力を、次の予測モデルの独立変数として用いる
- モデルを2段・3段と積み上げることで複雑なモデルを実現
  - Kaggle等でも多用される
  - コスト大



<http://www.chioka.in/wp-content/uploads/2013/09/stacking.png>



# スタッキングのモデル： ある層の出力は次の層の入力

---

- スタッキング：複数のモデルを並列・直列に結合する
  - ディープニューラルネットワークの構造に類似
  - 別種のモデルでも可能
- $\ell$ 段目の出力が  $\ell + 1$ 段目の入力になる
  - 0段目の出力  $\mathbf{y}_0 =$  元々の独立変数ベクトル  $\mathbf{x}$
  - $\ell$ 段目の出力  $\mathbf{y}_\ell$
  - $\ell + 1$ 段目の入力  $\mathbf{x}_{\ell+1} = \begin{pmatrix} \mathbf{x}_\ell \\ \mathbf{y}_\ell \end{pmatrix}$

# スタッキングにおける難点： 単純に積んだだけではダメ

---

- 単純な方法で実現してみる：

1. データ  $D$  から予測モデル  $f$  を推定
2.  $D$  に対する  $f$  の出力を次のモデルの入力にする

.... これでうまくいきそう？ ... が実際にはダメ

- 「大原則」を思い出す：モデル推定に用いたデータに対する予測は信用してはいけない

- モデルは推定に用いるデータを再現するように推定されるので、データに偏っている

# スタッキングの正しい実施法： 交差検証の方式を用いる

- 推定用データを  $K$  個に分割して：
  1.  $K - 1$  個をモデル推定に用いる
  2. 作ったモデルを残り1個に適用して、次段に渡す
    - 上記のステップ 1&2 を  $K$  通り繰り返せばデータセット全体に対して、推定に用いていないモデルによる予測が得られる
- 上記によって拡張されたデータで次の層（2 層目）のモデル推定を行う
- 以降、同様の手続きを繰り返して積みただけ積む
- 各層の各モデルが  $K$  個できてしまうので、最後にもう一度全データでモデルを推定しなおす