

データサイエンス

第11回

～クラス分類(2)～

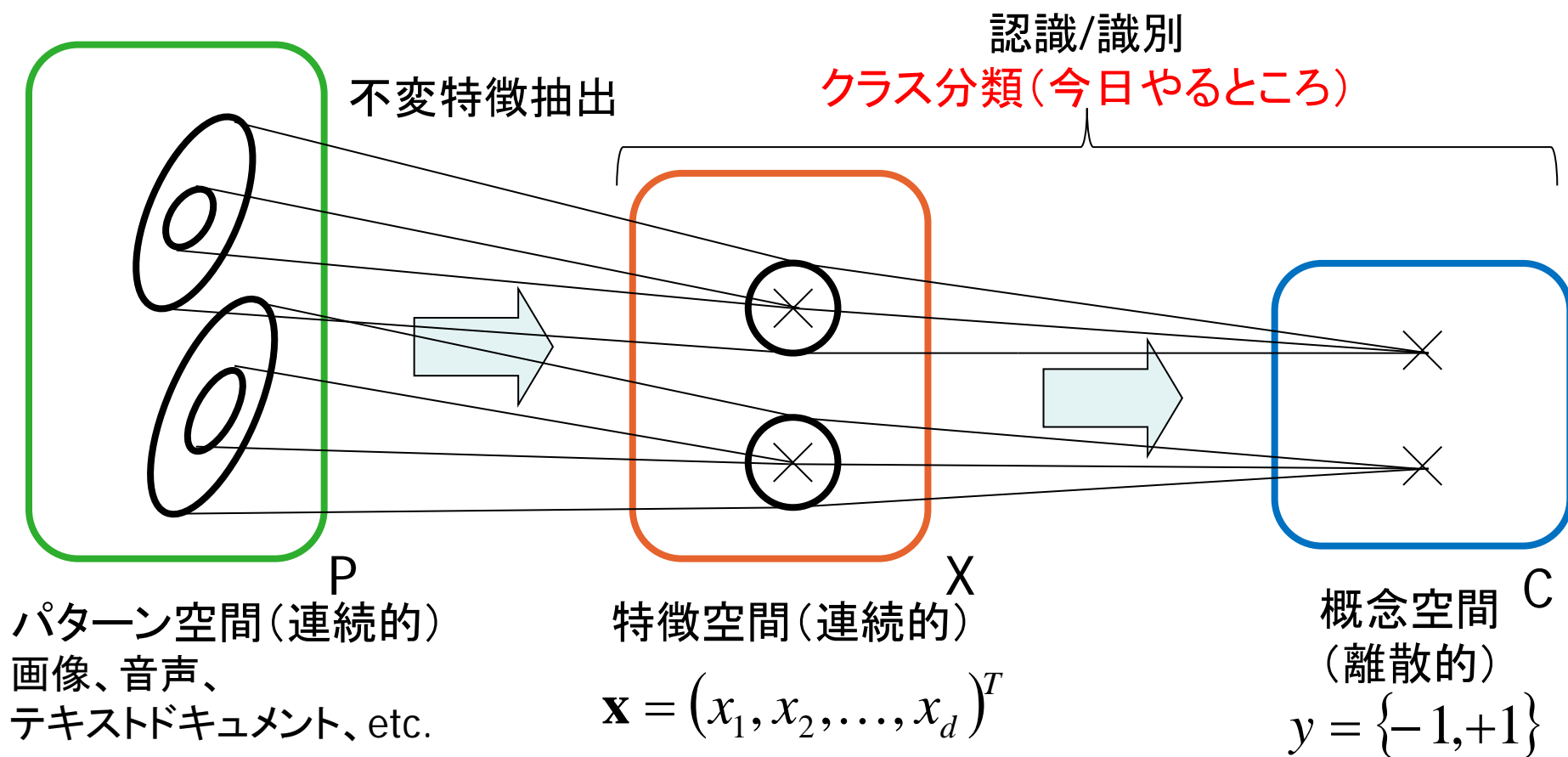
情報理工学系研究科
創造情報学専攻
中山 英樹

本日の内容

- クラス分類
 - 前回の復習（と残り）
- 第二回課題説明
- アンサンブル学習
 - バギング、ブースティング

パターン認識のパイプライン

- よい特徴量（説明変数）を抽出・選択することは極めて重要
 - Garbage in garbage out
 - （ドメイン依存なので今日は扱いませんが…）



方法論

- パターンの特徴ベクトル \mathbf{x} が与えられた時のクラス C の事後確率が重要

$$P(C | \mathbf{x}) = \frac{P(\mathbf{x} | C)P(C)}{P(\mathbf{x})} \quad \leftarrow \text{ベイズの定理}$$

- 事後確率を最大とするクラスへ識別
 - 誤識別率を最小にする
 - 誤識別のリスク（ペナルティ）がクラスによらず一定の時、最適な識別境界を与える

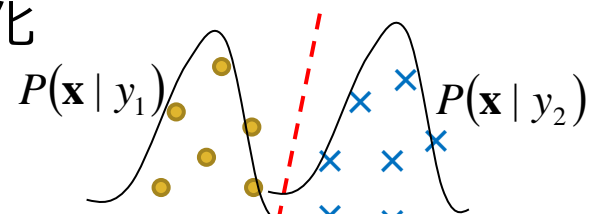
$$\hat{C} = \arg \max_c P(C | \mathbf{x})$$

分類のアプローチ（事後確率の推定方法）

より一般的

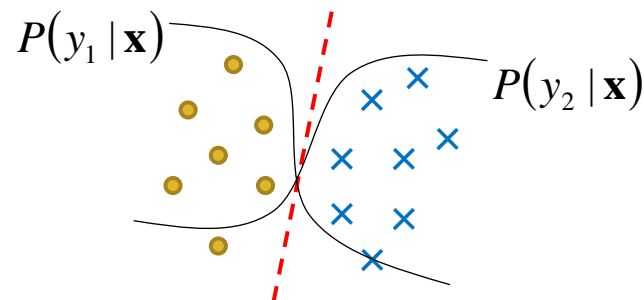
- 1. 生成モデル
 - クラスごと条件付き確率と事前確率をモデル化
 - ナイーブベイズ
 - k-最近傍法

$$P(y | \mathbf{x}) = \frac{P(\mathbf{x} | y)P(y)}{P(\mathbf{x})}$$

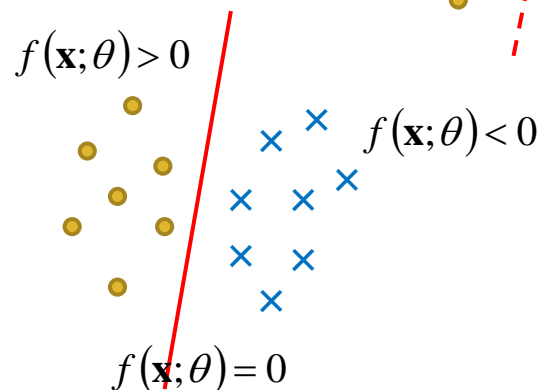


- 2. 識別モデル
 - 事後確率を $P(y | \mathbf{x})$ 直接的にモデル化（元の分布はどうでもよい）
 - ロジスティック回帰

$$P(y_1 | \mathbf{x}) > P(y_2 | \mathbf{x}) \quad P(y_2 | \mathbf{x}) > P(y_1 | \mathbf{x})$$



- 3. 識別関数
 - 識別の境界面だけモデル化
 - SVM



より識別に特化

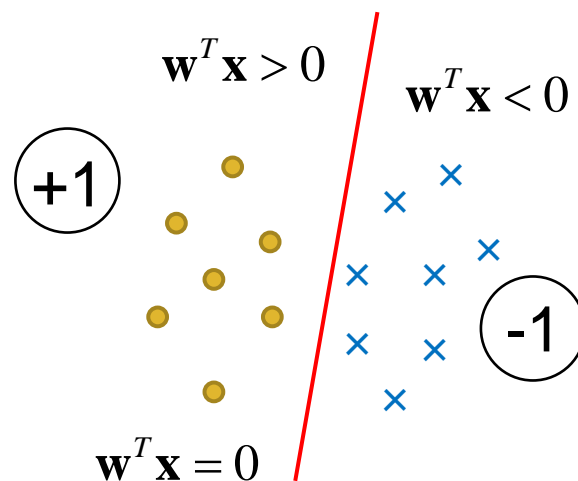
線型識別関数&識別モデル

- 以下、2カテゴリの分類問題を扱う
 - 出力 (目的変数)は $y = \pm 1$ の二値とする

$$\hat{y} = \begin{cases} 1 & \mathbf{w}^T \mathbf{x} > 0 \\ -1 & \mathbf{w}^T \mathbf{x} < 0 \end{cases}$$

最終的には符号しか使わない

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x})$$



多クラス分類

- One-versus-all

- あるクラスと、それ以外のクラス全てを識別する2クラス識別器 C 個のうち、最も高い出力を得たクラスを採用

$$\left. \begin{array}{l} \text{クラス 1 とクラス 1 以外の識別器 : } f_1(\mathbf{x}) \\ \text{クラス 2 とクラス 2 以外の識別器 : } f_2(\mathbf{x}) \\ \dots \\ \text{クラス } C \text{ とクラス } C \text{ 以外の識別器 : } f_C(\mathbf{x}) \end{array} \right\} \hat{y} = \arg \max_k f_k(\mathbf{x})$$

- 一般的にはこの方法がよく用いられる（と思う）
- 各識別器に与える学習サンプル数が多く、正例・負例が不均衡になりやすい

多クラス分類

- One-versus-one

- Cクラス中の全ての2クラス識別器による多数決を行う
 - 全部で $C(C-1)/2$ 個の識別器

$$\text{sign}(f_{k,k'}(\mathbf{x})) = \begin{cases} 1 & \Rightarrow \text{クラス}k\text{に一票} \\ -1 & \Rightarrow \text{クラス}k'\text{に一票} \end{cases}$$

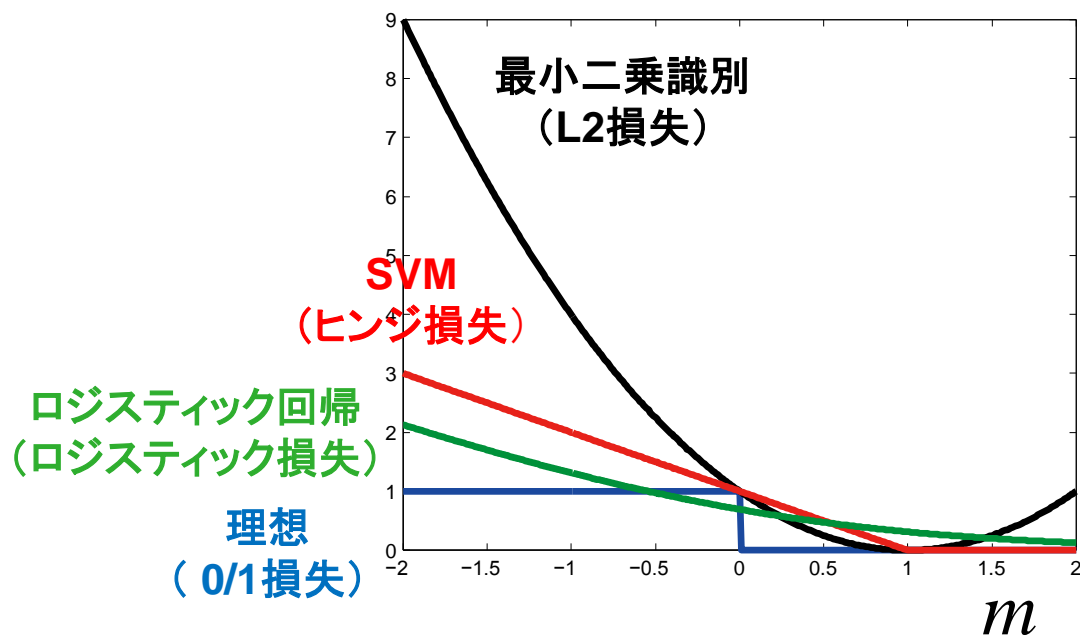
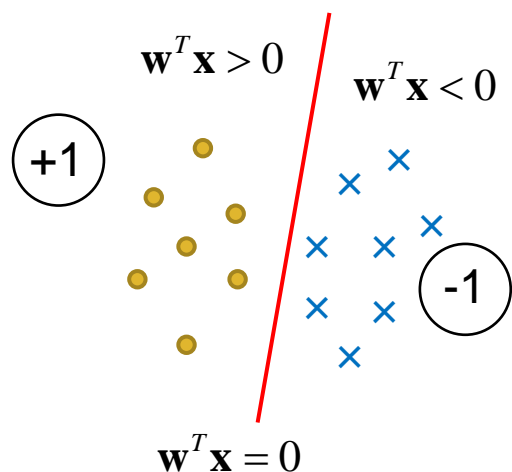
- 例) 3クラスの場合は $f_{1,2}(\mathbf{x}), f_{2,3}(\mathbf{x}), f_{3,1}(\mathbf{x})$ の3つの識別器を構築
- クラス数が増えると識別器の数が増えて大変
- 識別器あたりのサンプル数は（相対的に）少ない

識別的アプローチ（線形モデル）

- 大枠としては、どれもパーセプトロン
- 損失の測り方が違う

サンプル i の教師 $\{-1, +1\}$

マージン $m_i = (\mathbf{w}^T \mathbf{x}_i) y_i$



Support Vector Machine (SVM)

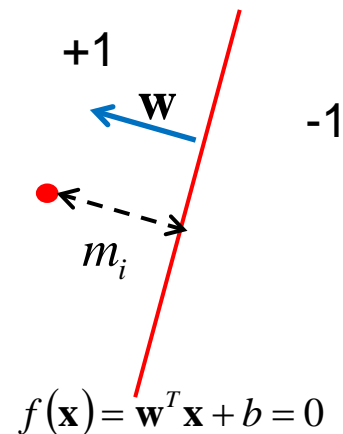
- 現在、最も標準的に用いられる識別器の一つ
- ヒンジ損失 + L2正則化

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w}} \left[\sum_{i=1}^n \max(0, 1 - (\mathbf{w}^T \mathbf{x}_i + b)y_i) + \lambda \|\mathbf{w}\|^2 \right]$$

- (最小)マージンの最大化を行う手法
 - あるサンプル i の正規化マージン

$$m_i = \frac{y_i f(\mathbf{x}_i)}{\|\mathbf{w}\|} \quad f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

= サンプルと分離超平面との距離



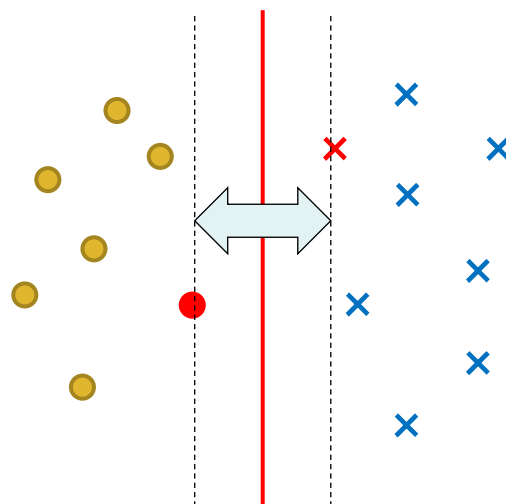
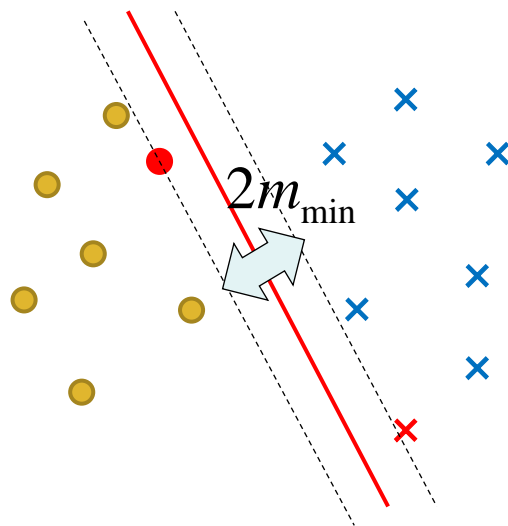
最小マージン

- 最小マージン：サンプルごとの正規化マージンの最小値
 - その識別平面にとって、もっとも「難しい」サンプル（=サポートベクター）のマージン

$$m_{\min} = \min m_i$$

- 最小マージンを最大とする識別平面は一般に汎化性が高い

どっちがよい
識別平面？



ハードマージンSVM

- 最小マージンを最大化する識別超平面を求める

$$\max_{\mathbf{w}, b} \left(\min_i y_i f(\mathbf{x}_i) / \|\mathbf{w}\| \right) \quad f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

- 線型分離可能（全学習サンプルを正しく分離する平面が引ける）ことが前提

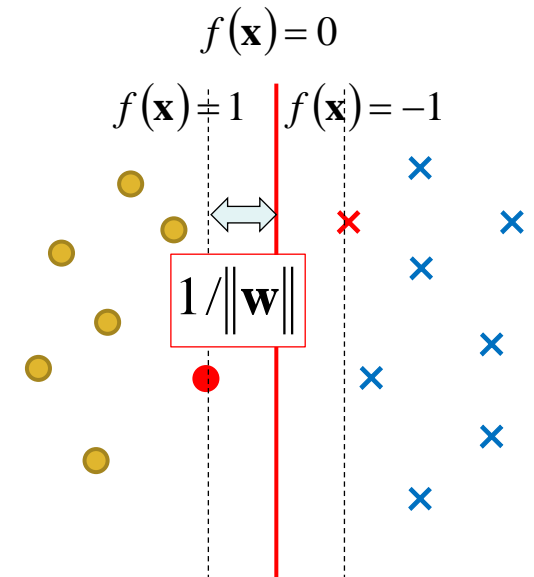
- 二次計画問題（等価な表現）

$$\min_{\mathbf{w}, b} \|\mathbf{w}\|^2 \quad \longleftrightarrow \quad \max 1 / \|\mathbf{w}\|$$

subject to $y_i f(\mathbf{x}_i) \geq 1$ for $\forall i$

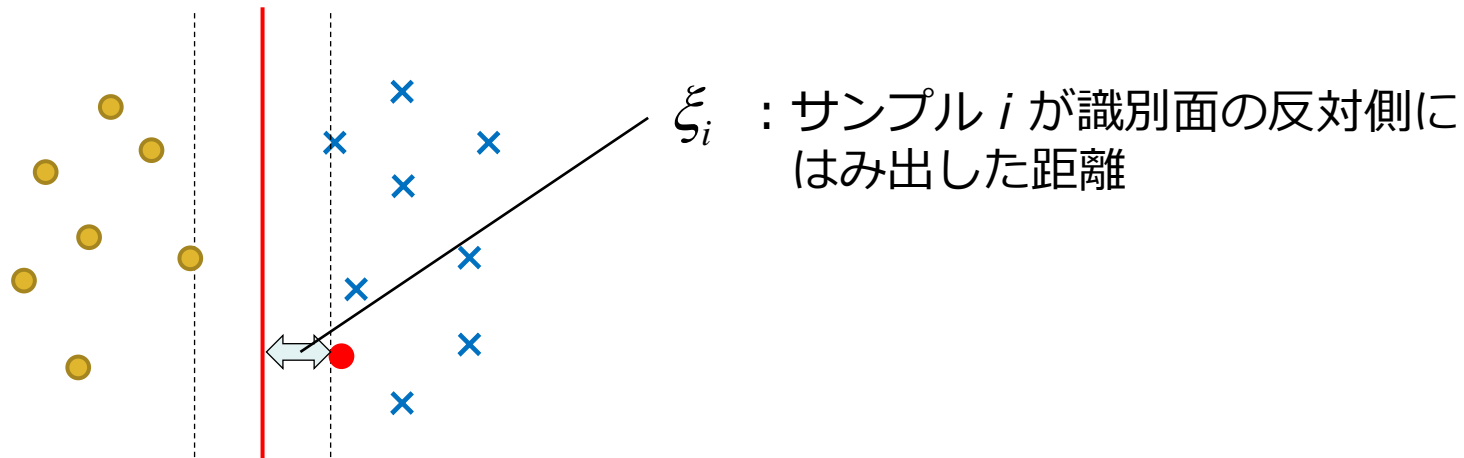
$y_i f(\mathbf{x}_i) > 0$

w, b に関してスケールの自由度があるので置き換え可能（サポートベクターが $f(\mathbf{x}) = \pm 1$ 上にあるように）



ソフトマージンSVM

- 実際には線形分離可能でないことがほとんどなので、少しの誤差 ξ_i を各サンプルに許容する



$$\min_{\mathbf{w}, b, \xi} \left(\|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \right)$$

$C > 0$ は許容誤差の程度を決めるパラメータ
(とても重要!)

$$\text{subject to } y_i f(\mathbf{x}_i) \geq 1 - \xi_i, \xi_i \geq 0 \text{ for } \forall i$$

実際には、ラグランジュ双対問題と呼ばれる等価表現による最適化問題を解くのがスタンダード

生成的アプローチ

- 事後確率分布だけでなく、同時確率分布までモデル化

$$\underline{P(C | \mathbf{x})} \propto \underline{P(\mathbf{x} | C)} \underline{P(C)}$$

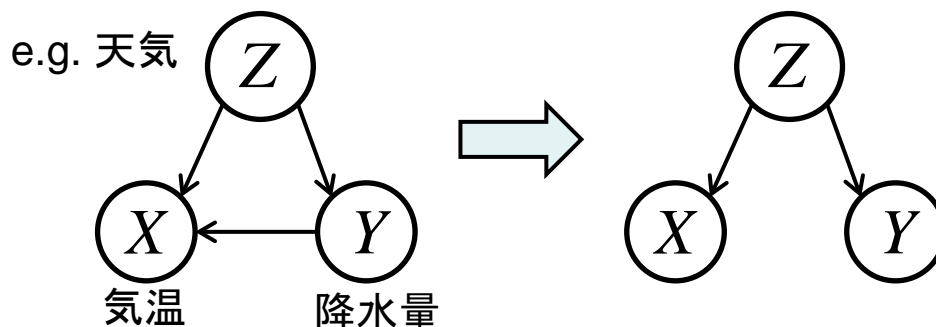
事後確率 条件付き確率 事前確率

- 条件付き確率（生成モデル）の推定が重要

ナイーブベイズ (単純ベイズ)

- もともと識別手法の名前ではない（非常に一般的かつ重要な概念）
 - 条件付きの同時確率をこの条件付き確率の積へばらす近似方法

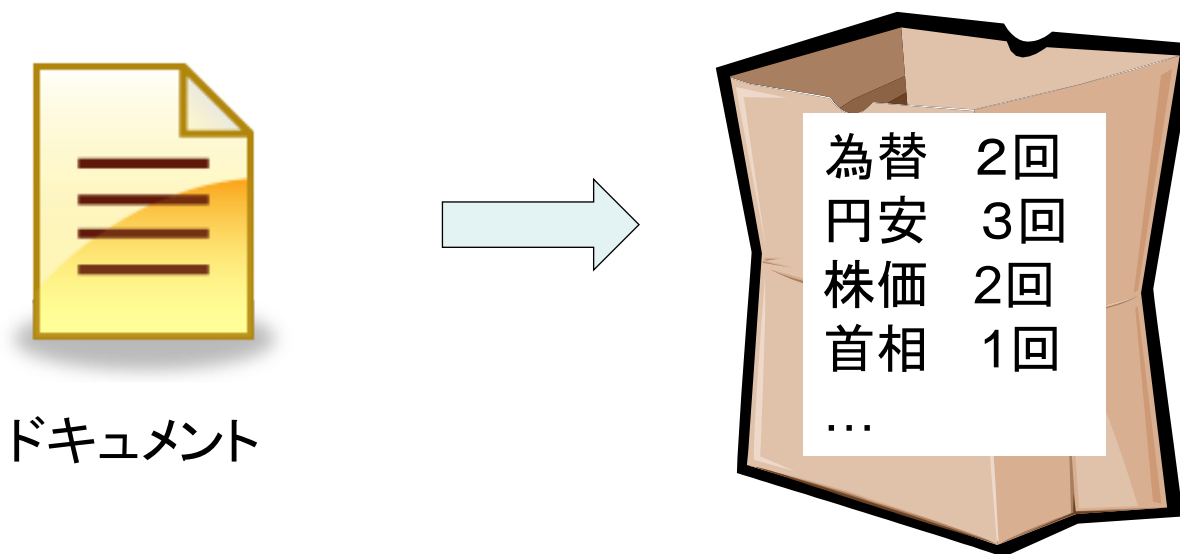
$$P(X, Y | Z) \cong P(X | Z)P(Y | Z)$$



- Z が潜在的な構造をよく捉えていれば、比較的妥当な近似になると期待できる

ナイーブベイズ識別

- テキスト分類の基本的な手法
- ドキュメントを、出現する単語の集合として表現
=bag-of-words
 - 出現回数だけ利用
 - 各単語の位置や出現順などのコンテキストは考慮しない



ナイーブベイズ識別

ドキュメント D の事後確率

$$P(C | D) \propto \frac{P(D | C)P(C)}{P(D)}$$

訓練サンプル中の比率で近似(あるいは単に一定)

ナイーブベイズ

$$\hat{P}(D | C) = P(W_1, W_2, \dots, W_n | C) \propto P(W_1 | C)P(W_2 | C) \cdots P(W_n | C)$$

$$P(W_i | C) = \frac{\text{カテゴリ } C \text{ に属する訓練データ中の単語 } W_i \text{ の数}}{\text{カテゴリ } C \text{ に属する訓練データの全単語数}}$$

- あらかじめ、訓練データ中の単語を数え上げておくだけで識別ができる！

例) スパムメール識別

非スパム

Hi Peter,

With Jose out of town, do you want to meet once in a while to keep things going and do some interesting stuff?

Let me know
Eugene

スパム

--- Codeine 15mg -- 30 for \$203.70 -
- VISA Only!!! --

-- Codeine (Methylmorphine) is a
narcotic (opioid) pain reliever
-- We have 15mg & 30mg pills --
30/15mg for \$203.70 - 60/15mg for
\$385.80 - 90/15mg for \$562.50 --
VISA Only!!! ---

```
>>> import bayes
```

```
>>> bayes.spamTest()
```

```
classification error ['home', 'based', 'business', 'opportunity', 'knocking', 'your', 'door', 'don',  
'rude', 'and', 'let', 'this', 'chance', 'you', 'can', 'earn', 'great', 'income', 'and', 'find', 'your',  
'financial', 'life', 'transformed', 'learn', 'more', 'here', 'your', 'success', 'work', 'from', 'home',  
'finder', 'experts']
```

```
the error rate is: 0.1
```

(ランダムにサンプルを選ぶので毎回違った結果になる)

スパム識別：コード

```
def spamTest():
    (略：データの読み込みなど)
    vocabList = createVocabList(docList)    #create vocabulary
    trainingSet = range(50); testSet=[]    #create test set
    for i in range(10):
        randIndex = int(random.uniform(0,len(trainingSet))) #ランダムに10サンプルをテスト用を選ぶ
        testSet.append(trainingSet[randIndex])
        del(trainingSet[randIndex])
    trainMat=[]; trainClasses = []
    for docIndex in trainingSet: #train the classifier (get probs) trainNB0
        trainMat.append(bagOfWords2VecMN(vocabList, docList[docIndex]))
        trainClasses.append(classList[docIndex])
    p0V,p1V,pSpam = trainNB0(array(trainMat),array(trainClasses))
    errorCount = 0
    for docIndex in testSet:    #classify the remaining items
        wordVector = bagOfWords2VecMN(vocabList, docList[docIndex])
        if classifyNB(array(wordVector),p0V,p1V,pSpam) != classList[docIndex]:
            errorCount += 1
        print "classification error",docList[docIndex]
    print 'the error rate is: ',float(errorCount)/len(testSet)
    #return vocabList,fullText
```

スパム識別：コード

```
def trainNB0(trainMatrix,trainCategory):    # 2クラス識別が前提
    numTrainDocs = len(trainMatrix)
    numWords = len(trainMatrix[0])
    pAbusive = sum(trainCategory)/float(numTrainDocs)    # スпамメールの事前確率
    p0Num = ones(numWords); p1Num = ones(numWords)    #change to ones()
    p0Denom = 2.0; p1Denom = 2.0    #change to 2.0
    for i in range(numTrainDocs):    #クラスごとに単語の数え上げ&条件付き確率計算
        if trainCategory[i] == 1:
            p1Num += trainMatrix[i]
            p1Denom += sum(trainMatrix[i])
        else:
            p0Num += trainMatrix[i]
            p0Denom += sum(trainMatrix[i])
    p1Vect = log(p1Num/p1Denom)    #change to log()
    p0Vect = log(p0Num/p0Denom)    #change to log()
    return p0Vect,p1Vect,pAbusive
```

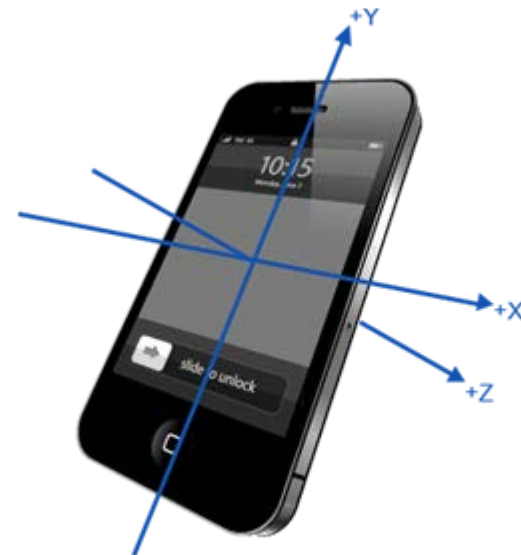
```
def classifyNB(vec2Classify, p0Vec, p1Vec, pClass1):    # 2クラス識別が前提
    p1 = sum(vec2Classify * p1Vec) + log(pClass1)    #element-wise mult
    p0 = sum(vec2Classify * p0Vec) + log(1.0 - pClass1)
    if p1 > p0:
        return 1
    else:
        return 0
```

第二回レポート課題（クラス分類）

- Human Activity Recognition Using Smartphones Sensor Data

Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz. Human Activity Recognition on Smartphones using a Multiclass Hardware-Friendly Support Vector Machine. International Workshop of Ambient Assisted Living (IWAAL 2012). Vitoria-Gasteiz, Spain. Dec 2012.

- スマートフォンの慣性センサデータ等からユーザの行動を推定
- 30人の被験者データ
- 特徴量（説明変数）：561種類
- 行動カテゴリ（目的変数）：6種類
 - 1. WALKING
 - 2. WALKING UPSTAIRS
 - 3. WALKING DOWNSTAIRS
 - 4. SITTING
 - 5. STANDING
 - 6. LAYING



<http://classmethod.s3.amazonaws.com/wp-content/uploads/2012/06/gyrocompass.png>

配布物

- 学習データ
 - X_train.csv : 特徴ファイル
 - y_train.csv : カテゴリID
 - subject_train.csv : 被験者ID
- テストデータ
 - X_test.csv : 特徴ファイル
- サンプルプログラム
 - sample1_knn.py k最近傍法
 - sample2_svm.py サポートベクターマシン
- その他
 - activity_labels.txt カテゴリ名
 - features_info.txt, features.txt 特徴量の説明
 - authors.pdf 元論文

コンペティション形式

- 被験者IDでデータを学習用・テスト用にランダムに分割
 - 学習データ：22人分 （全7600サンプル）
 - テストデータ：8人分 （全2699サンプル）※元の論文とは分け方が違うので注意
- 評価指標
 - 識別正解率（Accuracy） = $\text{正解したサンプル数} / \text{全サンプル数}$
- ランキング
 - スコアリングサーバ: <http://www.nlab.ci.i.u-tokyo.ac.jp/~nakayama/ds13/report2/index.php>
 - テストサンプルの識別結果を一行ずつ記載したテキストファイルを提出
 - ユーザ名を忘れずに（区別できればなんでもよい。）
 - 提出した結果は上書きされる。最後のものだけ保存されるので注意。
 - 現在は、提出されたテストデータのうち、所定の被験者4人分のサンプルでスコアリングしている
 - 最終的なスコアは、締め切り後に残りの4人分のサンプルで算出

課題詳細

- レポート内容
 - Human Activity Recognition の問題に取り組み、以下の点を中心にA4用紙1~2枚程度にまとめよ。講義で扱っていない技術を用いても構わない。
 - 1. 識別性能を向上させるための自分なりの工夫点と結果、考察（必須）
 - コードを載せる必要はない（もちろん、実装がポイントの場合は載せてかまわない）
 - 2. その他、自由にデータを分析した結果（+α）
 - 学習データ数と性能、正則化パラメータの関係
 - いろいろな手法の比較
 - カテゴリごとの識別率 …など
 - 3. 講義の感想、要望など

課題詳細

- 実装が難しい場合の課題
 - 配布している元論文あるいは、データマイニング・機械学習に関する論文を一つ自由に選び、内容をまとめよ。また、それが自分の研究にどのように応用できそうか考察せよ

課題詳細

- 評価の方針
 - アイデアや試行錯誤の過程を重視
 - スコアが悪くても、しっかり考察してくれればOK
(もちろん良くなればプラスに評価します)
 - 面白い分析を期待します
- 提出先
 - 以下のアドレスへメールで提出すること（質問もこちらへ）
 - ds2013@nlab.ci.i.u-tokyo.ac.jp
 - 件名は「データサイエンスレポート課題2」
 - 氏名、学籍番号、所属を忘れずに

締め切り

- レポート提出締め切り **2月15日（土）**
- コンペについて
 - 2月4日（火） 18:00に一度テストサンプルで中間評価
 - ちょっと変則的ですが、最終日（2月5日）に一度みんなで結果を見たいので
 - 2月11日（火） 18:00にもう一度ランキング（最終）
 - いいレポートのアイデア等は、本人に了解を取りつつ、ホームページに掲載したいと思います。

工夫できそうなところ

- バリデーションの方法
 - 識別したいのは、学習データに含まれてないユーザの行動
 - クロスバリデーションはどのように行うべきか？
- 手法・特徴
 - SVM, Random forest, ...
 - パラメータ

アンサンブル学習

- 複数の(性能が低い)識別器を組み合わせることにより強力な識別を行うアプローチの総称
- 「三人よれば文殊の知恵」

$$\text{混合モデル} \quad p(y | \mathbf{x}) = \sum_{k=1}^K \pi_k(\mathbf{x}) p_k(y | \mathbf{x})$$

$$\text{コミッティ (多数決)} \quad y_{COM}(\mathbf{x}) = \frac{1}{K} \sum_{k=1}^K y_k(\mathbf{x})$$

- 分類に限らず、回帰等さまざまな問題へ適用可能な概念
 - c.f. コンピュータ将棋

Bagging (**B**ootstrap **A**ggregating)

- ブートストラップ法により学習データのサブセットを繰り返し抽出し、それぞれ弱識別器を構築
- 決定木分類器(後述)とよく合わせて用いられる

1. For $k = 1, \dots, K$

(a) n 個の学習サンプル $\{\mathbf{x}_i, y_i\}_{i=1}^n$ から、重複を許してランダムに n 個を抽出

(b) 抽出したサンプルを用い、弱識別器 f_k を学習

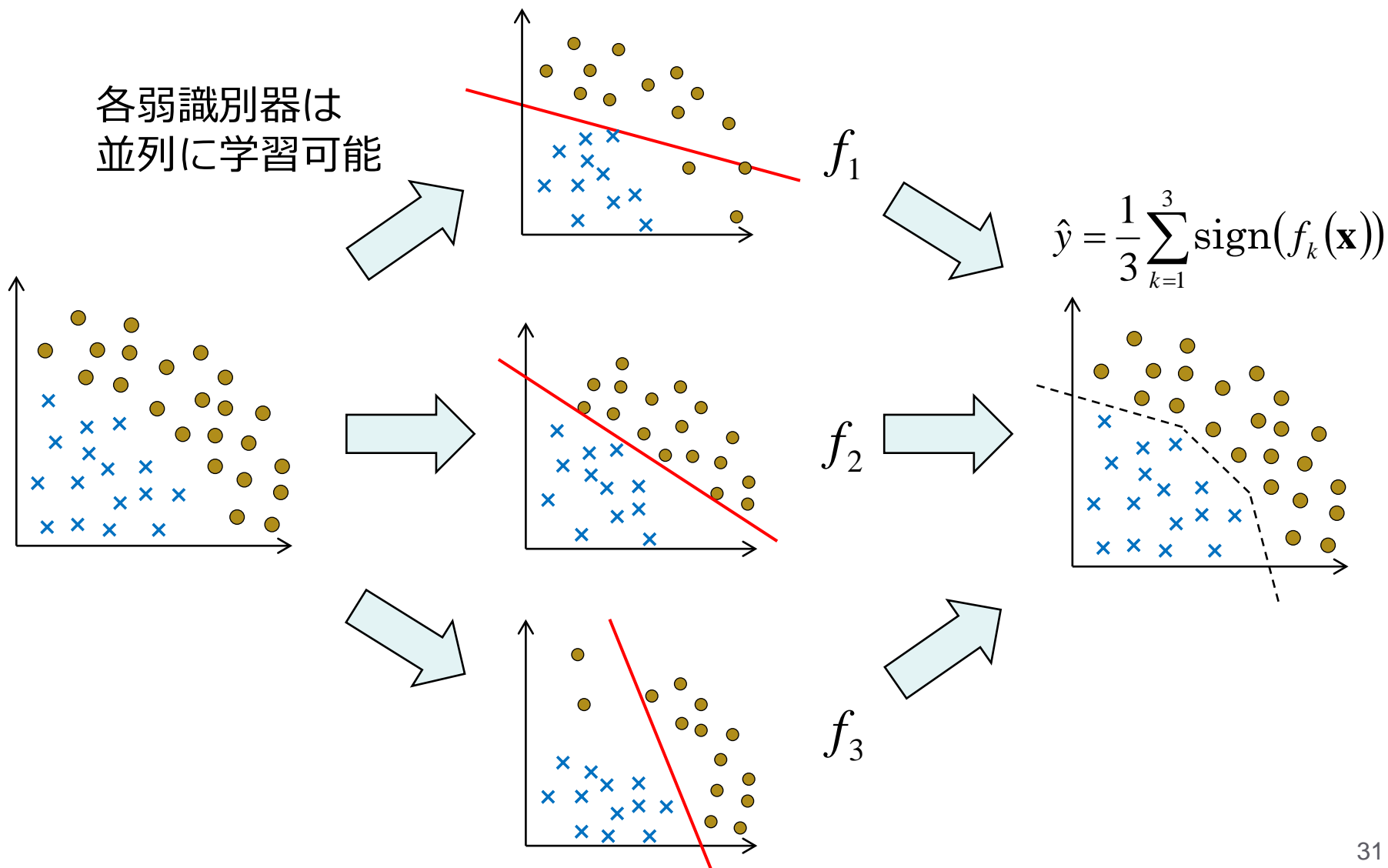
2. 全ての弱識別器 $\{f_k\}_{k=1}^K$ の平均を強学習器として出力

$$f(\mathbf{x}) = \frac{1}{K} \sum_{k=1}^K f_k(\mathbf{x})$$

- 識別関数の場合、要は多数決

Bagging

各弱識別器は
並列に学習可能

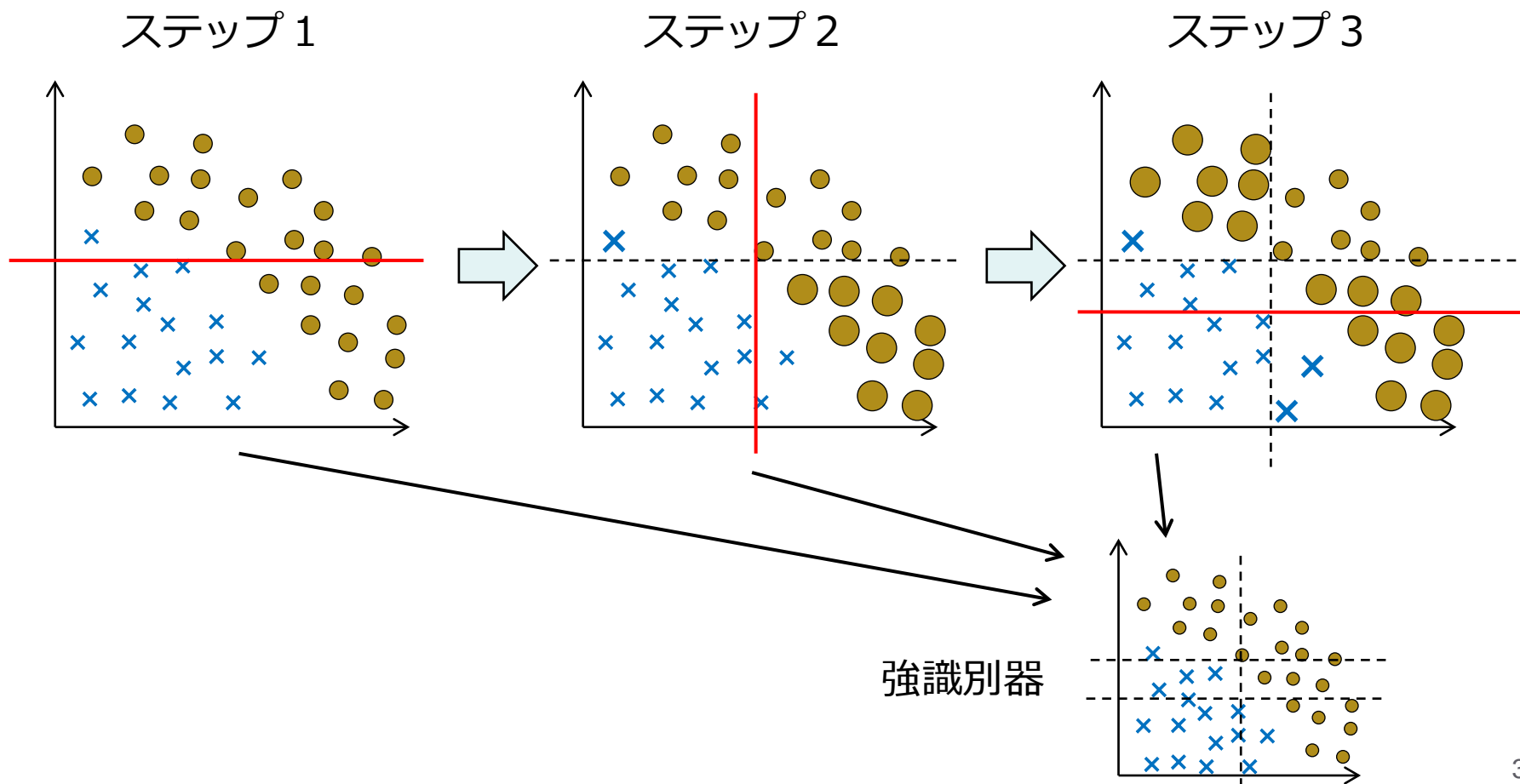


Boosting

- 複数の弱識別器を順々に学習
 - それまでに学習した弱識別器がうまく分類できない、難しい学習サンプルに強い重みをつけ次の弱識別器を学習
 - 最終的に、何らかの信頼度で重み付平均した結果を出力
- Baggingよりもよい識別性能を示すことが多い
- AdaBoost(Adaptive boosting)と呼ばれる手法が最も一般的
 - Y. Freund and R. E. Schapire. "A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting", 1995.

AdaBoost

- 弱識別器を順々に学習



AdaBoostのアルゴリズム

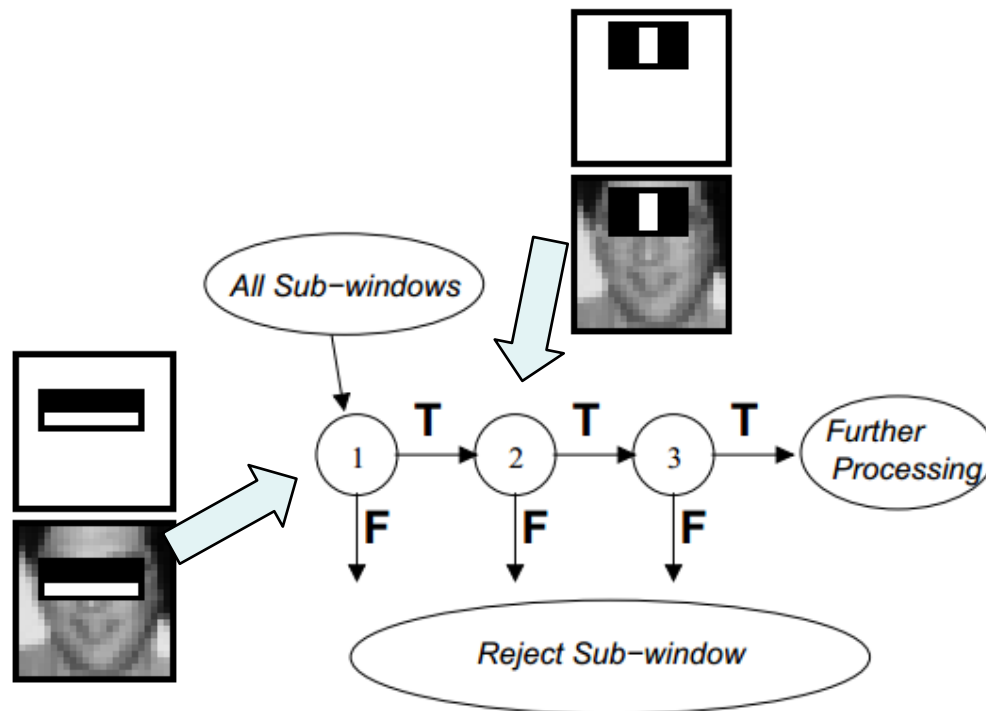
$\{\mathbf{x}_i, y_i\}_{i=1}^n, y_i \in \{-1, +1\}$: 学習データセット

$W_k(i)$: k番目の弱識別器に与えるサンプルの重み

- 重みの初期化 $W_1(i) = 1/n$
- For $k = 1, \dots, K$
 - W_k に基づいて弱識別器 f_k を学習
 - 識別器の信頼度 $\alpha_k = \frac{1}{2} \log \left(\frac{1 - \varepsilon_k}{\varepsilon_k} \right)$ 重み付経験誤差
 - サンプル重みの更新 $W_{k+1}(i) = \frac{1}{Z_k} W_k(i) \exp(-\alpha_k y_k f_k(\mathbf{x}_i))$
正規化定数 ←
- Output: $f(\mathbf{x}) = \text{sign} \left\{ \sum_{k=1}^K \alpha_k f_k(\mathbf{x}) \right\}$ ← 信頼度で重み付けた多数決

例) Viola-Jones の顔識別アルゴリズム

- P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", 2001.
 - Haar-like特徴とAdaBoostを利用
 - カスケードによる高速な識別（明らかに顔でないものは早々に却下）



ちなみに...

- AdaBoostは、指数損失を最小化する逐次的な学習アルゴリズム

$$J(\bar{f}) = \frac{1}{n} \sum_{i=1}^n \exp(-\underbrace{y_i \bar{f}(\mathbf{x}_i)}_{= m_i}) \quad \bar{f}(\mathbf{x}) = \sum_{k=1}^K \alpha_k f_k(\mathbf{x})$$

- 損失関数が異なる他のboostingアルゴリズム

- LogitBoost: Logistic損失
確率的な解釈が可能
- MadaBoost:
異常値に対してロバスト

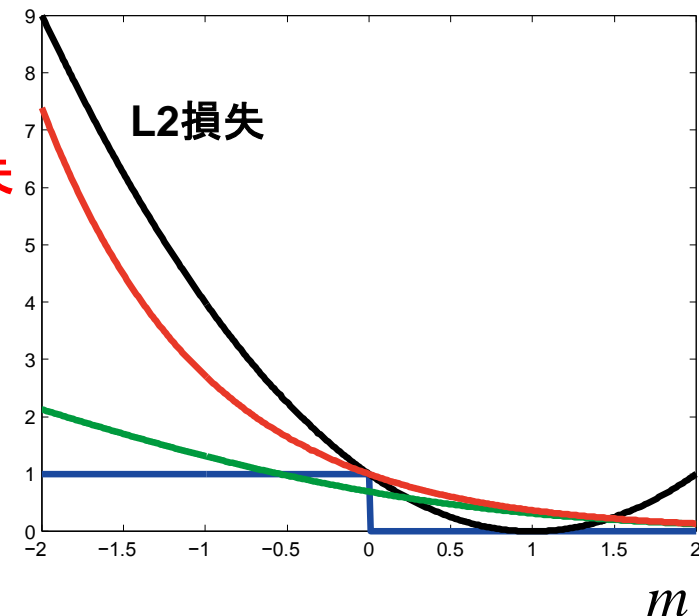
$$\begin{cases} -m + 1/2 & (m \leq 0) \\ \exp(-2m)/2 & (m > 0) \end{cases}$$

ロジスティック損失

指数損失

0/1損失

Loss



サンプルコード

- Decision Stump (決定株分類器) を弱識別器とするAdaBoost

```
>>> import adaboost
>>> datArr,labelArr = adaboost.loadDataSet('horseColicTraining2.txt')
>>> classifierArray,aggClassEst
>>> adaboost.adaBoostTrainDS(datArr,labelArr,10) #弱識別器の数
>>> testArr,testLabelArr = adaboost.loadDataSet('horseColicTest2.txt')
>>> prediction10 = adaboost.adaClassify(testArr,classifierArray)
```

# weak classifiers	Training error	Testing error
1	0.28	0.27
10	0.23	0.24
50	0.19	0.21
100	0.19	0.22
500	0.16	0.25
1000	0.14	0.31

BaggingとBoosting：まとめ

	Bagging	Boosting
弱識別器の学習	並列	逐次的
サンプルへの重みづけ	ブートストラップによるサブセット抽出	誤識別に基づき重みを更新
識別	単純な多数決	信頼度による重み付け多数決

議論

- どんな弱識別器を使うべき？
- そもそも何故集団学習はうまくいく？

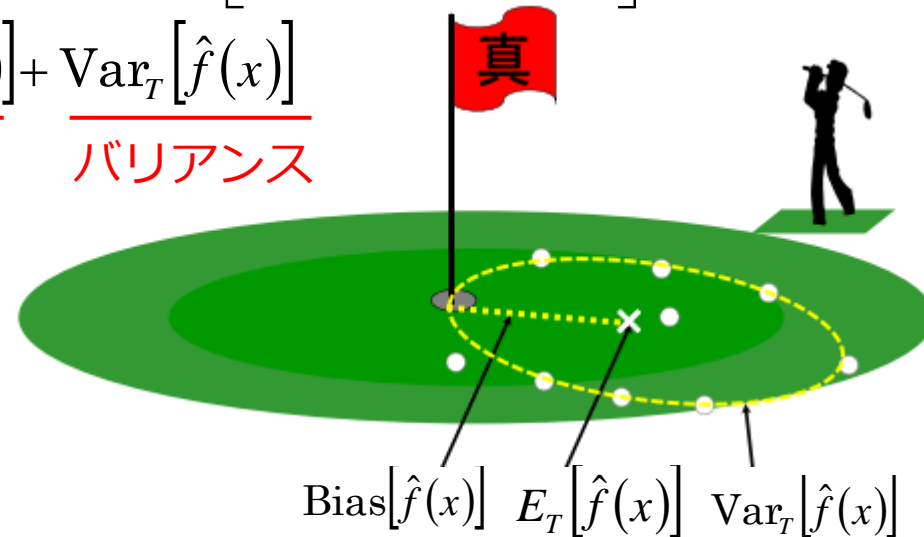
バイアス-バリエンス分解

$f(x)$: 真の分布 $\hat{f}(x)$: あるモデルのもとで、学習データ T から推定した分布

$$\underline{E\left[\left(y - \hat{f}(x)\right)^2\right]} = \sigma^2 + \left(E_T\left[\hat{f}(x)\right] - f(x)\right)^2 + E_T\left[\left(\hat{f}(x) - E_T\left[\hat{f}(x)\right]\right)^2\right]$$

$$\text{二乗汎化誤差} = \sigma^2 + \underbrace{\text{Bias}^2\left[\hat{f}(x)\right]}_{\text{バイアス}} + \underbrace{\text{Var}_T\left[\hat{f}(x)\right]}_{\text{バリエンス}}$$

http://www.nii.ac.jp/userdata/karuizawa/h23/111104_3rdlecueda.pdf より引用



- バイアス：仮定したモデルの、真の構造との本質的なずれ
- バリエンス：仮定したモデルの下での、訓練サンプルの違いに起因する推定結果のばらつき

バイアス-バリエーション分解

- バイアス：仮定したモデルの、真の構造との本質的なずれ
- バリエーション：仮定したモデルの下での、訓練サンプルの違いに起因する推定結果のばらつき
- バイアス・バリエーションはトレードオフの関係
 - 線形など単純なモデルは、バイアス大、バリエーション小
 - 高次の複雑なモデルは、バイアス小、バリエーション大
- 適切な複雑さのモデルを選ぶ必要がある

アンサンブル学習の効果

- 識別器のバリエーションを減らす
- Baggingの場合

$$E_T \left[E_k \left\{ (y - f_k(x))^2 \right\} \right] = E_T \left[(y - \bar{f}(x))^2 \right] + \underbrace{E_T \left[E_k \left\{ (\bar{f}(x) - f_k(x))^2 \right\} \right]}_{\geq 0}$$

$$\therefore \underbrace{E_T \left[(y - \bar{f}(x))^2 \right]}_{\text{平均識別器のバリエーション}} \leq \underbrace{E_T \left[E_k \left\{ (y - f_k(x))^2 \right\} \right]}_{\text{もとの識別器のバリエーション}}$$

- Boostingの場合も同様
 - 誤差の大きい点での重みを増やし、より積極的にバリエーションを減らす

弱識別器のモデル

- 低バイアス・高バリエーションが理想
 - SVM等は低バリエーションのモデルなので、あまり集団学習の効果を得られない
- 直感的には
 - 一つ一つの識別器はデータに対して過学習しても、平均をとっていけばいいところに落ち着く

ランダムフォレスト（次回）

- 決定木を弱識別器としたバギング
- 識別以外にも回帰、クラスタリングなど広く応用されている
- SVMと並んでファンの多い識別手法

まとめ

- SVM
 - ヒンジロス最小化、最小マージン最大化
- アンサンブル学習
 - 3人よれば文殊の知恵
 - バギング、ブースティング
- 次回
 - 決定木、Random forest、カーネル法、大規模化