正　誤　情　報

このたびは森北出版株式会社発行の書籍をお買い求めいただき，誠にありがとうございました．下記の書籍につきまして誤りのある箇所がございましたので，お詫びし訂正させていただきます．

2020年8月21日　　森北出版株式会社　生産マネジメント部

タイトル

フリーソフトではじめる機械学習入門(第2版)

正誤対象

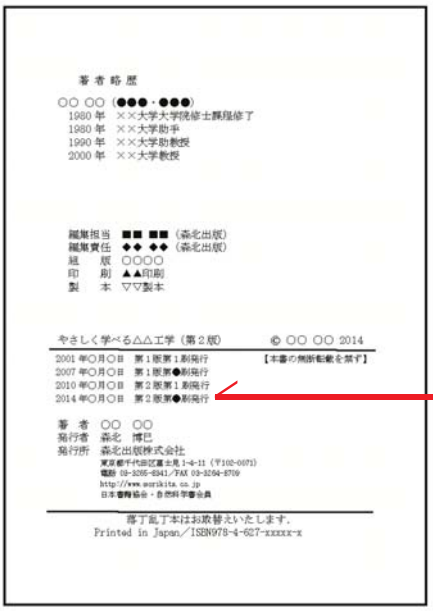
お手持ちの書籍の刷数をお調べのうえ，下の表をご覧下さい．正誤表内の一番左に「対応刷数」という列がございます．該当する刷数の訂正情報をご参照下さい．

なお，刷数につきましては下記「刷数の調べ方」をご参照ください．

お持ちの本の刷数						
1	対応刷数	1	より	2	までをご参照ください	
2	対応刷数	2	をご参照ください			
それ以降	現在把握している訂正情報はございません					

刷数の調べ方

本の一番後ろのページ(広告等除く)に下図のようなページがございます．ご参照いただき，お持ちの本の刷数をお調べください．


著者略歴

○○ ○○（●●●・●●●）
1980年　××大学大学院修士課程修了
1980年　××大学助手
1990年　××大学助教授
2000年　××大学教授

編集担当　■■ ■■（森北出版）
編集責任　◆◆ ◆◆（森北出版）
組　版　○○○○
印　刷　▲▲印刷
製　本　▽▽製本

やさしく学べる△△工学（第2版）　© ○○ ○○ 2014
2001年○月○日　第1版第1刷発行　【本書の無断転載を禁ず】
2007年○月○日　第1版第●刷発行
2010年○月○日　第2版第1刷発行
2014年○月○日　第2版第●刷発行

著　者　○○　○○
発行者　森北　博巳
発行所　森北出版株式会社
東京都千代田区富士見1-4-11（〒102-0071）
電話 03-3265-8341/FAX 03-3264-8709
http://www.morikita.co.jp
日本書籍協会・自然科学書協会

落丁・乱丁本はお取替えいたします．
Printed in Japan/ISBN978-4-627-xxxxx-x

日付の最も新しい行に記載された数字がお持ちの本の刷数となります

対応刷数	頁	行数，図・表・式番号	誤	正
2	51	下から2，1行目	最頻正解クラス	最頻クラス　　（2か所）
2	92	下から7行目	d 次元列ベクトルの…	d+1 次元列ベクトルの…
2	94	式（5.11）	（右辺の分母） $1+\exp\{-(w_0+\boldsymbol{w}\cdot\boldsymbol{x})\}$	$1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x})$
2	95	4行目	尤度の最大値 $\mathcal{L}(D)$ を求めるときは，計算をしやすいように対数尤度に…	尤度の最大値を求めるときは，計算をしやすいように対数尤度 $\mathcal{L}(D)$ に…
2	97	下から13行目	$g(\boldsymbol{x})=w_0+\boldsymbol{w}\cdot\boldsymbol{x}$	$g(\boldsymbol{x})=\boldsymbol{w}\cdot\boldsymbol{x}$
2	103	式(6.6)	$R^2=1-\dfrac{\sum_{i=1}^{N}\{y_i-\hat{c}(x_i)\}}{\sum_{i=1}^{N}(y_i-\tilde{y})}$	$R^2=1-\dfrac{\sum_{i=1}^{N}\{y_i-\hat{c}(x_i)\}^2}{\sum_{i=1}^{N}(y_i-\tilde{y})^2}$
2	117	6.8節7行目	…，線形識別を正規化項を入れながら…	…，線形回帰を正規化項を入れながら…
2	121	式（7.8）	$\dfrac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}=-\lambda\dfrac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}}$	$\dfrac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}=\lambda\dfrac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}}$
2	121	図7.2の右の式	$\dfrac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}=-\lambda\dfrac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}$	$\dfrac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}=-\lambda\dfrac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}}$
2	122	式（7.12）	$L(\alpha)=\dfrac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j-\sum_{i=1}^{N}\alpha_i$	$L(\alpha)=-\dfrac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j+\sum_{i=1}^{N}\alpha_i$
2	122	6行目	次はこれを最小化するわけですが，これは…	得られた式は，もとの問題の双対問題で，最小化問題が最大化問題に入れ替わります．これは…
2	123	7行目	式（7.7）の SVM の…	式（7.6）の SVM の…
2	124	下から9行目	…，線形識別面が存在する可能性が高くなります．	…，線形識別面が偶然に存在する可能性が高くなります．

2	129	例題 7.4 1 行目	…，Weka に付属の ReutersGrain.arff に対して，…	…，Weka に付属の ReutersGrain データに対して，…
2	139	下から 1 行目	最下行の式を 1 行削除する．	
2	140	式(8.6)	$w_j \leftarrow w_j - \eta \cdots$	$w_j \leftarrow w_j + \eta \cdots$
2	141	アルゴリズム 8.1 10〜12 行目	$\delta_h \leftarrow o_k(1-o_k)\sum_k w_{kh}\delta_k$ /*重みの更新*/ $\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta\delta_j \boldsymbol{x}$	$\delta_h \leftarrow o_h(1-o_h)\sum_k w_{kh}\delta_k$ /*重みの更新（g_i は一つ手前の層のユニット i の出力）*/ $w_{ij} \leftarrow w_{ij} + \eta\delta_j g_i$
2	148	式（9.2）	$E(\boldsymbol{w}) \equiv -\sum_{\boldsymbol{x}\in D} y_i \log o_i$	$E(\boldsymbol{w}) \equiv -\sum_{\boldsymbol{x}_i\in D} y_i \log o_i$
2	152	式（9.3）	$\min \sum_{i=1}^{N} \mathrm{Dist}(f_i, y_i)$	$\min \sum_{i=1}^{N} \mathrm{Dist}(\boldsymbol{f}_i, \boldsymbol{y}_i)$　（太字にする）
2	156	上から 3 つめの網掛け 1 行目	＃　画像の次元数を入力	＃　入力画像の次元数
2	169	下から 6 行目	…，この確率は 0.349 となり，N= 100 で，0.366 です．	…，この確率は約 0.349 となり，N= 100 で約 0.366 です．
2	179	下から 8 行目	h_m	$h(\boldsymbol{x};\gamma_m)$
2	179	下から 7，6，4， 3 行目	h_m	h
2	188	アルゴリズム 11.2 10 行目	$\boldsymbol{\mu}_j \leftarrow \frac{1}{N_j} \sum_{\boldsymbol{x}_k\in\text{クラスタ}j} \boldsymbol{x}_k$	$\boldsymbol{\mu}_j \leftarrow \frac{1}{N_j} \sum_{\boldsymbol{x}_i\in\text{クラスタ}j} \boldsymbol{x}_i$
2	195	例題 11.5 見出し	**例題 11.5** Weka	**例題 11.5** Python
2	201	6 行目	…と分散 $\boldsymbol{\Sigma}_m$ を推定する問題になります．	…と共分散行列 $\boldsymbol{\Sigma}_m$ を推定する問題になります．
2	202	アルゴリズム 11.3 3 行目	入力空間上に k 個の分布 ϕ_j をランダムに設定	入力空間上に k 個のクラスタ c_j 分布 ϕ_j をランダムに設定

2	202	アルゴリズム 11.3 6 行目	**for all** 学習データ $\boldsymbol{x}^{(i)}$ **do**	**for all** $\boldsymbol{x}_i \in D$ **do**
2	202	アルゴリズム 11.3 7 行目	$p\left(\boldsymbol{x}^{(i)}\middle\|c_j\right)=\phi_j\left(\boldsymbol{x}^{(i)}\right)\left(j=1,\ldots,k\right)$ を計算	ϕ_j を用いて確率 $p\left(c_j\middle\|\boldsymbol{x}_i\right)\ \left(j=1,\ldots,k\right)$ を計算
2	202	アルゴリズム 11.3 10 行目	E ステップの確率 $p\left(\boldsymbol{x}^{(i)}\middle\|c_j\right)$ を使って…	E ステップで求めた $p\left(c_j\middle\|\boldsymbol{x}_i\right)$ を使って…
2	202	アルゴリズム 11.3 最下行	最下行の次に右の 1 行を挿入	**return** $\phi_j\ \left(j=1,\ldots,k\right)$
2	202	下から 4 行目	E ステップのガウス混合モデルによる尤度計算は，…	E ステップのガウス混合モデルによる確率計算は，…
2	202	式（11.7）	$p\left(c_m\middle\|\boldsymbol{x}^{(i)}\right)=\dfrac{p\left(c_m\right)p\left(\boldsymbol{x}^{(i)}\middle\|c_m\right)}{p\left(\boldsymbol{x}^{(i)}\right)}$	$p\left(c_m\middle\|\boldsymbol{x}_i\right)=\dfrac{p\left(c_m\right)p\left(\boldsymbol{x}_i\middle\|c_m\right)}{p\left(\boldsymbol{x}_i\right)}$
2	202	式（11.8）	$=\dfrac{p\left(c_m\right)p\left(\boldsymbol{x}^{(i)}\middle\|c_m\right)}{\sum_{j=1}^{k}p\left(c_j\right)p\left(\boldsymbol{x}^{(i)}\middle\|c_j\right)}$	$=\dfrac{p\left(c_m\right)p\left(\boldsymbol{x}_i\middle\|c_m\right)}{\sum_{j=1}^{k}p\left(c_j\right)p\left(\boldsymbol{x}_i\middle\|c_j\right)}$
2	202	式（11.9）	$=\dfrac{p\left(c_m\right)\phi\left(\boldsymbol{x}^{(i)};\boldsymbol{\mu}_m,\boldsymbol{\Sigma}_m\right)}{\sum_{j=1}^{k}p\left(c_j\right)\phi\left(\boldsymbol{x}^{(i)};\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j\right)}$	$=\dfrac{p\left(c_m\right)\phi\left(\boldsymbol{x}_i;\boldsymbol{\mu}_m,\boldsymbol{\Sigma}_m\right)}{\sum_{j=1}^{k}p\left(c_j\right)\phi\left(\boldsymbol{x}_i;\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j\right)}$
2	203	1 行目	この各データの尤度を用いた…	この各データの確率を用いた…
2	203	式(11.10)	$\boldsymbol{\mu}_m=\dfrac{1}{\left\|D\right\|}\sum_{\boldsymbol{x}^{(i)}\in D}\boldsymbol{x}^{(i)}$	$\boldsymbol{\mu}_m=\dfrac{1}{\left\|D\right\|}\sum_{\boldsymbol{x}_i\in D}p\left(c_m\middle\|\boldsymbol{x}_i\right)\boldsymbol{x}_i$
2	203	式(11.11)	式全体を削除	
2	203	式(11.12)	$\boldsymbol{\Sigma}_m=\dfrac{1}{\left\|D\right\|}\sum_{\boldsymbol{x}^{(i)}\in D}\left\{\boldsymbol{x}^{(i)}-\boldsymbol{\mu}_m\right\}\left\{\boldsymbol{x}^{(i)}-\boldsymbol{\mu}_m\right\}^T$	$\boldsymbol{\Sigma}_m=\dfrac{1}{\left\|D\right\|}\sum_{\boldsymbol{x}_i\in D}p\left(c_m\middle\|\boldsymbol{x}_i\right)\left(\boldsymbol{x}_i-\boldsymbol{\mu}_m\right)\left(\boldsymbol{x}_i-\boldsymbol{\mu}_m\right)^T$
2	213	アルゴリズム 12.1 7 行目	**for all** $\boldsymbol{x}\ \in\ D$ **do**	**for all** $\boldsymbol{x}_i\ \in\ D$ **do**

2	213	アルゴリズム 12.1 9 行目	**if** $c \subset x$ **then**	**if** $c \subset x_i$ **then**
2	213	アルゴリズム 12.1 13～14 行目	$F_k \leftarrow \{c \in C_k \mid c.count > \text{閾値}\}$ **end for**	**end for** $F_k \leftarrow \{c \in C_k \mid c.count > \text{閾値}\}$　（行の入れ替え）
2	221	下から 5~4 行目	そして，この新しい x：1 をルートとする FP–木に対して残りの要素を挿入する作業を再帰的に繰り返します．	このように，それまでにできた木の構造にデータを当てはめ，当てはまらないところは新たに枝を伸ばすという手順で，すべてのデータを木に追加してゆきます．
2	228	2 行目	これが系列認識問題です．	これが系列識別問題です．
2	233	下から 6 行目	・状態の集合：$\{S_i\}(1 \geq i \geq n)$	・状態の集合：$\{S_i\}(1 \leq i \leq n)$
1	249	14.3 1 行目	右のように差し替え	14.3 Python scikit-learn の LabelPropagation アルゴリズムにおいて，データの類似度計算法を変更し，性能の変化を確認せよ．
1	275	14.3	欄外①に差し替え	
2	277	B.1 6 行目	KnowledgeFlow，Simple CLI の中から選びます．	KnowledgeFlow，Workbench，Simple CLI の中から選びます．
2	277	表 B.1	（KnowledgeFlow の行の下に右の 1 行を挿入）	Workbench　上記三つを組み合わせ，ある程度カスタマイズ可能なインタフェース
2	278	B.2 4 行目	…，保存しておくことできるというメリットがあります．	…，保存しておくことができるというメリットがあります．
2	293	13 行目	値の分布を可視化が可能な…	値の分布の可視化が可能な…

欄外①

14.3　ここでは例題 14.3 の手順を参考に，iris データのうち 10% を正解付きとして半教師あり学習を行います．以下のコードでは，類似度計算法として k-NN 法（$k = 3$ と $k = 5$）とガウシアンカーネル ($\gamma = 1, 10, 100$) で性能評価を行っています．iris データのような性質のよいデータの場合は，ガウシアンカーネルを用いて γ（分散の逆数）の値を大きくすると，近くのデータの値の影響が大きくなるので，比較的よい結果となるようです．

```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.semi_supervised import LabelPropagation

iris = load_iris()
X = iris.data
y = iris.target

# 10%が正解付きデータ
unlabeled_points = np.random.choice(np.arange(y.size),
int((y.size)*.9), replace=False)
labels = np.copy(y)
labels[unlabeled_points] = -1

labeled = 0.1
num = y.size
lps = [LabelPropagation(kernel='knn', n_neighbors=3),
       LabelPropagation(kernel='knn', n_neighbors=5),
       LabelPropagation(kernel='rbf', gamma=1),
       LabelPropagation(kernel='rbf', gamma=10),
       LabelPropagation(kernel='rbf', gamma=100)]
for lp in lps :
    score = 0
    for i in range(100):
        unlabeled_points = np.random.choice(np.arange(num),
int(num-num*labeled), replace=False)
        labels = np.copy(y)
        labels[unlabeled_points] = -1
        lp.fit(X, labels)
        score += lp.score(X[unlabeled_points], y[unlabeled_points])
    print("{0}{1:4.1f}{2}{3:6.3f}".format("labeled:", labeled*100,
"%,score=", score/100))
```