知的情報処理
3. 最近傍法
事例ベース

櫻井彰人
慶應義塾大学理工学部

本日の目的

- 「k-最近傍法」or「最近傍法」とは
- 機械学習の最も基本なアイデア
  - 事例をたくさん記憶する
  - 必要になったら、事例そのものか、近い事例を思い出す。

もう少し具体的には
  - 事例(入力 x_i と出力 y_i の対)を記憶する
  - 問合せ(入力 x に対する出力 y は何?)に対し、
  - 答える(x に近い事例を参照して、最適なyを作る)

そして
- その基本と問題(実に一般的な問題です)

R の基礎勉強もします

目次

- 丸暗記(?)
- 作ってみよう(ぴったり一致するもの)
  - 簡単なプログラム例
- 少し発展させよう(ぴったりでなくとも)
  - 近さとは
    - 次元と単位と距離
  - どの事例を参考に
    - 複数個と多数決
  - プログラム例

丸暗記

丸暗記 --- オウムのように

- 考えてみよう。
- 最近傍法に必要なものは?
- 答え:
- 教えられたことを全部、間違いなく覚えること、すなわち、丸暗記
- 問合せがあったら、記憶の中から、最も似たものを探し出す。

最近傍法のポイント

http://teflworldwiki.com/index.php/Rote_Learning
http://sozaishu.seesaa.net/article/115935835.html

丸暗記学習するプログラム

- 丸暗記すること自体は、コンピュータにとって難しいことではない(そのようなプログラムを書くことは、難しいことではない)
- しかし、
  - 思い出すときに、速くできるようにする、
  - データが大量でも対応できるようにする

のは、容易ではない。

- 今回は、これらの問題は考えない(つまり、基本技術だけ考える)

目次

- 丸暗記(?)
- 作ってみよう(ぴったり一致するもの)
  - 簡単なプログラム例
- 少し発展させよう(ぴったりでなくとも)
  - 近さとは
    - 次元と単位と距離
  - どの事例を参考に
    - 複数個と多数決
  - プログラム例

作ってみよう

- 課題
- 四字熟語のリストがある。
- 1文字欠けたある四字熟語が与えられたとき、そこに一文字入れて、正しい（つまり与リストにある）四字熟語にしなさい。

概略仕様

- プログラムはRで記述する
- 四字熟語のリストは、sjisコード（Rで読める）で作成してある
- 質問の「1文字欠けたある四字熟語」は、例えば、「縦横無〇」のように、欠け字は「〇」で表現してあるものとする。
- 解は複数ある可能性を排除しない。解の四字熟語を配列（リスト）で返す。
- 簡単のため、
  ・プログラム内で四字熟語のリストを読み、
  ・質問の「1文字欠けたある四字熟語」は定数で変数に代入するとする。

何を行えばよいか

cwlist.txt

：
：
一期一会
一五一十
一言一句
一汁一菜
一日一善
：
：

for-ループです

```
"D:/R/Sample" を working directory にする
x <- "cwlist.txt", を文字列のファイルだと思って読む
y <- "縦横無〇"

for ( i in 1:(xの第一列の長さ) ) {
  score <- 0
  for ( j in 1:4 ) {
    if ( xのi行目のj番目の文字 == yのj番目の文字 ) score <- score +1
  }
  if ( score >= 3 ) { print( i ); print( xのi行目 ) }
}
```

同じ文字の個数が3以上なら、印字する

出力

```
[1] 1219
[1] "縦横無尽"
```

プログラム完成

cwlist.txt

：
：
一期一会
一五一十
一言一句
一汁一菜
一日一善
：
：

for-ループです　　列を指定する方法です　　文字列として読む

```
setwd("D:/R/Sample")
x <- read.table("cwlist.txt", as.is=1)
y <- "縦横無〇"

for ( i in 1:length(x[,1]) ) {
  score <- 0
  for ( j in 1:4 ) {
    if ( substr(x[i,],j,j) == substr(y,j,j) ) score <- score +1
  }
  if ( score >= 3 ) { print( i ); print( x[i,] ) }
}
```

部分文字列の取り出し

同じ位置に同じ文字がある個数を求める

その個数が3以上なら、印字する

出力

```
[1] 1219
[1] "縦横無尽"
```

例1: 結果を記憶する

```
setwd("D:/R/Sample")
x <- read.table("cwlist.txt", as.is=1)
y <- "縦横無〇"

s <- c( )
for ( i in 1:length(x[,1]) ) {
  score <- 0
  for ( j in 1:4 ) {
    if ( substr(x[i,],j,j) == substr(y,j,j) ) score <- score +1
  }
  if ( score >= 3 ) s <- c(s,i)
}
print( s ); print( x[s,] )
```

ベクトルの最後尾に i を追加する

出力

```
[1] 1219
[1] "縦横無尽"
```

これはskipしましょう

例2: Rの機能活用

```
setwd("D:/R/Sample")
x <- read.table("cwlist.txt", as.is=1)
y <- "縦横無〇"

s <- c( )
for ( i in 1:length(x[,1]) ) {
  if ( length(agrep( y, x[i,] )) != 0 )   s <- c(s,i)
}
print( s ); print( x[s,] )
```

第一引数の文字列にほぼ等しい（defaultでは一文字間違いまで）文字列が引数2にあるとき、第二引数での位置を返す。ないとき、長さ0のベクトルを返す。

出力

```
[1] 1219
[1] "縦横無尽"
```

これは学習か？

- 単なるプログラム！
  - 否定するわけではないが
- 学習の基本であることは確かである。
  人間でも、
  - 文字を覚える
  - 数を覚える
  - 九九を覚える、
- このプログラムでは、学習らしくない。
- しかし、後で使いやすいように、データの記憶方法を工夫をすると、学習らしくなる。
  - つまり、記憶するにあたって、データの性質を（データの内なる構造を）利用する。これこそ、学習である。

目次

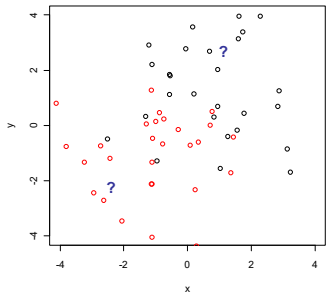
- 丸暗記（?）
- 作ってみよう（ぴったり一致するもの）
  - 簡単なプログラム例
- 少し発展させよう（ぴったりでなくとも）
  - 近さとは
    - 次元と単位と距離
  - どの事例を参考に
    - 複数個と多数決
  - プログラム例

少し発展させよう

- 右の問題で、穴埋めする時
- 説明変数値が
  - 知っているものと同じか
  - かなり近いときに、

被説明変数の値を答えにする
という方法を考えてみよう

図で見ると分りやすい



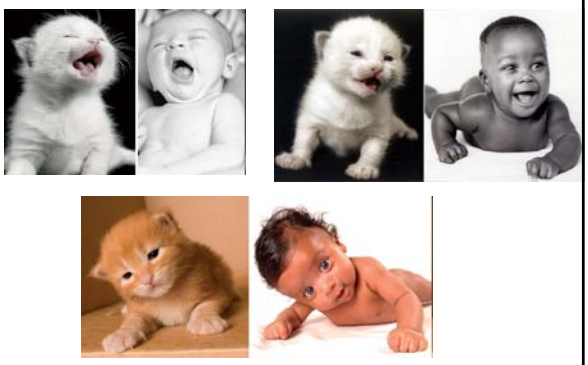
目次

- 丸暗記（?）
- 作ってみよう（ぴったり一致するもの）
  - 簡単なプログラム例
- 少し発展させよう（ぴったりでなくとも）
  - 近さとは
    - 次元と単位と距離
  - どの事例を参考に
    - 複数個と多数決
  - プログラム例

小目次

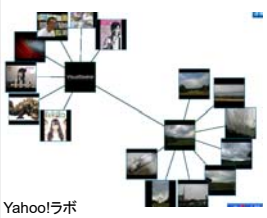
- 近さとは
  - 定義することの困難さ
    - 人間は、似ているものの発見が得意
      - しかし、言葉にできない
  - 「距離」でいいのか？
    - 単位の問題
    - 次元の問題
    - 正規化で解決

「似ている」の困難さ



http://gigazine.net/news/20071006_cat_similar_baby/

なかなか難しい



Yahoo!ラボ
VisualSeeker V2 - ウェブ類似画像検索

Yahoo!ラボ
FashionNavi

脱線: 逆手にとって: CAPTCHA

Completely Automated Public Turing Test To Tell Computers and Humans Apart

http://www.captcha.net/

似ているとは近いこと(?)

- 似ているとは近いことと、考えよう
- 近いとは「距離」が小さいこと。つまり、「近さ」とは「距離」である。
  - 同語反復。でもない。
  - 「距離」というと、数学では、距離の公理を満たす概念。
  - 距離の公理とは？
    - 非負性: $d(x,y)\geq 0$
    - 同一性: $x=y \Leftrightarrow d(x,y)=0$
    - 対称性: $d(x,y)=d(y,x)$
    - 三角不等式: $d(x,y)+d(y,z) \geq d(x,z)$
  - 代表的なものは:
    - ユークリッド距離: $\sqrt{((x_1-y_1)^2+(x_2-y_2)^2+(x_3-y_3)^2)}$
    - マンハッタン距離: $|x_1-y_1|+|x_2-y_2|+|x_3-y_3|$

小目次

- 近さとは
  - 定義することの困難さ
    - 人間は、似ているものの発見が得意
      - しかし、言葉にできない
  - 「距離」でいいのか？
    - 単位の問題
    - 次元の問題
    - 正規化で解決

「距離」でいいのか？

- 例として、身長と体重で、AさんとBさんの近さをみることにしよう
  - Aさんは (h_A, w_A), Bさんは (h_B, w_B) としよう
- 距離は？
  - $\sqrt{((h_A-h_B)^2+(w_A-w_B)^2)}$? or $|h_A-h_B|+|w_A-w_B|$? or …
- そもそも単位は？
  - 身長は、cm? mm? μm? m? km?
  - 体重は、kg? g? mg? pg? t?

閑話休題。
√の横棒がないけどおかしくはない。そもそもその横棒って何だ？

単位の問題とは？

- 身長と体重を2個の属性とし、次の三人の距離をみよう
  - A: (170cm, 65kg)
  - B: (180cm, 60kg)
  - C: (175cm, 70kg)
- マンハッタン距離を考えよう
  - d(A,B)=10+5=15
  - d(B,C)=5+10=15
  - d(C,A)=5+5=10

え？そう？

- cm と kg を用いるのは不公平。単位の名称からいって、m と g だろう。となると、
  - d(A,B)=0.01+5000=5000.01
  - d(B,C)=0.005+10000=10000.005
  - d(C,A)=0.005+5000=5000.005
- 他にも、いくらでも、考えられる。さて、どうしたものか。

単位の問題だけではない

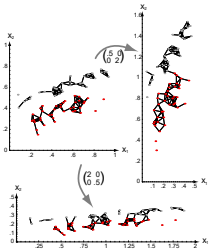
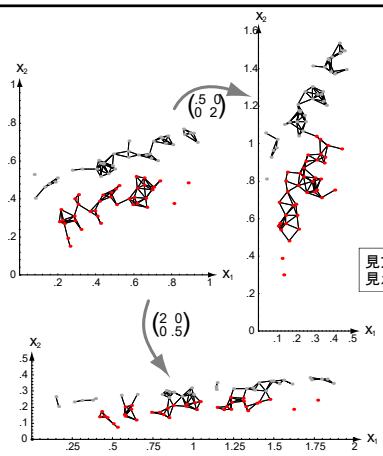


FIGURE10.8. Scaling axes affects the clusters in a minimum distance cluster method. The original data and minimum-distance clusters are shown in the upper left; points in one cluster are shown in red, while the others are shown in gray. When the vertical axis is expanded by a factor of 2.0 and the horizontal axis shrunk by a factor of 0.5, the clustering is altered (as shown at the right). Alternatively, if the vertical axis is shrunk by a factor of 0.5 and the horizontal axis is expanded by a factor of 2.0, smaller more numerous clusters result (shown at the bottom). In both these scaled cases, the assignment of points to clusters differ from that in the original space. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification* . Copyright 2001 by John Wiley & Sons, Inc.



つまり: 次元と単位

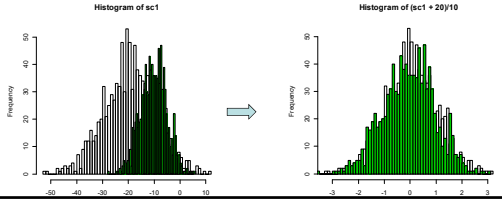
- 問題は2つある: 次元と単位
- 次元: 異なる物理量を区別する概念。次元が異なる物理量は比較さえできない。
  - 「1kgと1mとは、どちらがより偉いか？」は有意味か
  - なお、この例に表れる次元は3個。重さ、長さ、偉さ。
    - 「偉さ」が物理量かというと、まぁ、そうではないが。
- 単位: 物理量を数値で表すとき、数値の1に対応する物理量
  - 長さの単位の例: m, mile, 尺, Å
  - 勿論、1cm を単位にしても、3.14km を単位にしてもよい

次元と単位と距離

- 次元解析: 物理量を、質量、長さ、時間、電荷、温度などの次元の有理数冪で表現し、方程式や仮説の妥当性を調べること
- 目下の話題は、「次元」
  - 異なる次元の量は、比較・加算ができない。
    - 当然、(170cm, 65kg) と (180cm, 60kg) の距離をユークリッド距離的に、また、マンハッタン距離的には定義できない。
  - では、どうする？
- もう一つの話題は、「単位」
  - 異なる単位の量は、比較・加算ができない。
    - 身長と指の長さが属性のとき、(1.7m, 10cm) と (1.8m, 12cm) の距離をユークリッド距離的に、また、マンハッタン距離的には定義できない。
  - では、どうする？

正規化

- もしデータが大量にあれば、そして、各属性値が独立に正規分布していれば（ある確率分布に従っていれば）、「値/標準偏差」という量を使えばよい
  - この量は、無次元量。なぜなら、「値」もその「標準偏差」も同じ次元。
  - 「無次元量だから比較・加算できる」という訳ではないが、
  - 無次元の上に、各属性値の標準偏差が1に規格化されたのだから、よかろう

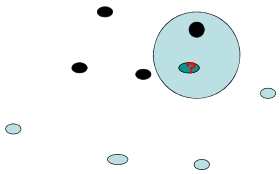


目次

- 丸暗記(?)
- 作ってみよう(ぴったり一致するもの)
  - 簡単なプログラム例
- 少し発展させよう(ぴったりでなくとも)
  - 近さとは
    - 次元と単位と距離
  - どの事例を参考に
    - 複数個と多数決
  - プログラム例

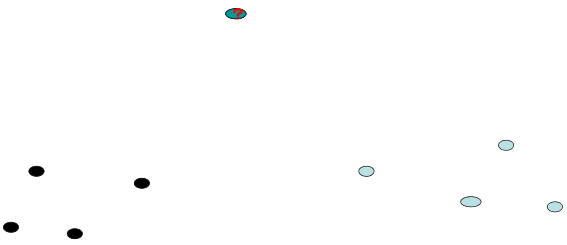
どの事例を参考に?

- 「最も近い事例」にしよう。そうすれば、
- 「どのくらい近ければよい」という基準を考えなくてすむ。
  - つまり、(適当に決めた基準で)見つからなかった場合にも対処できる



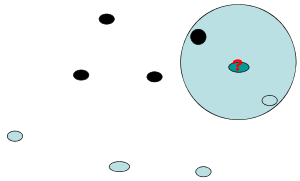
最も近いものが遠いと?

- 「「どのくらい近ければよい」という基準を考えなくてすむ。」とはいえ



近い事例が複数あると?

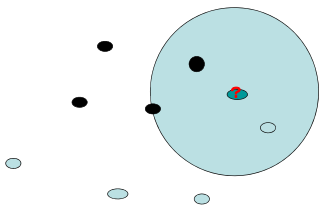
- 「一つ」に決めてしまったが、ちょっと不安。
- 例えば、



  - どっち?

多数決はどうか?

- 近い方から3人(とか5人とか7人とか、、、)の意見を聞くことにしたらどうだろうか。
- そして多数決をとればよいだろう
- 一般には、近くの k 人から聴くことにする、この方法を k-nearest neighbor 法という



目次

- 丸暗記(?)
- 作ってみよう(ぴったり一致するもの)
  - 簡単なプログラム例
- 少し発展させよう(ぴったりでなくとも)
  - 近さとは
    - 次元と単位と距離
  - どの事例を参考に
    - 複数個と多数決
  - プログラム例

深みに：多数決のプログラムは？

- まず、「多数」を選ばないといけない。
  - プログラムはちょっと複雑
- 例えば、集まった100人のうち、身長が最も低い5人を選ぶ方法を考えることになる。

どうすればいい？

- 一つの方法は、昔とった杵柄（のはず）のバブルソートです。
- 100人をバブルソートする途中で、つまり、上位5人が決まった時点で、バブルソートを中止すればよい。

深みに：多数決のプログラムは？（続）

- しかし、長いプログラムを書くのはサボることにしよう。
  - その代わりプログラム実行時間はかかるようになるのだが、まぁ、我慢。
- テストする x との距離を、すべての学習データに対して計算し、
  それを、全部（ここが無駄）ソートしてしまい（R内の関数を使えばプログラムは書かなくてすむので）、
  それから先頭の必要個数（今回は k 個）を使えばよい。
- 朗報！ R で sort の help を見てください。部分的にソートしてくれそうです。
  partial=5 とすると、最小値5個が得られたところで、ソートは終了するようです。
- しかし、index（何番目の要素かを示す値）は返してくれない。返すように指定する（ index.return=TRUE）のは完全ソートのときのみ可能
- partial は諦めて、index.return=TRUE としよう。
- なお、rank という関数もある。

実プログラム例: 2次元の場合

- 2次元かつ 0 or 1 に分類する場合の k-NN プログラムを作ってみよう。
- 入力の近さ判定にはユークリッド距離、出力の近似には、多数決を使う
- 実験用のデータとして、平均が (1,1) と (-1,-1)、分散共分散行列が 1.5^2 I（Iは単位行列）である2個の正規分布を考える。二つのクラスの事例は、それぞれの正規分布に従い生成されると考える。
- 何個かデータを生成する

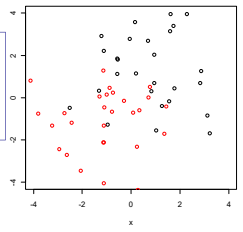
実験用データの作成

```
class1<-data.frame(x=rnorm(30,mean=1,sd=1.5),y=rnorm(30,mean=1,sd=1.5),c=rep(1,30))
class2<-data.frame(x=rnorm(30,mean=-1,sd=1.5),y=rnorm(30,mean=-1,sd=1.5),c=rep(2,30))
data<-rbind(class1,class2)

plot(subset(data,data$c==1)[,1:2],xlim=c(-4,4),ylim=c(-4,4))
points(subset(data,data$c==2)[,1:2],col="red",xlim=c(-4,4),ylim=c(-4,4))
```

```
> head(class1)
          x         y c
1  1.5121442 -1.721736 1
2 -0.9393179  1.716194 1
3 -0.6819147 -1.406909 1
4 -0.3132993  1.331400 1
5  0.5958607  1.326750 1
6  2.2467210  3.636144 1
> head(class2)
          x         y c
1 -1.1141637 -3.16300000 2
2 -0.2754737  0.08535921 2
3 -0.1761440  1.38123139 2
4 -0.9974249 -1.31036347 2
5 -0.4343134 -4.01635416 2
6  1.9064339  0.74648529 2
>
```

```
> tail(head(data,n=33),n=6)
           x         y c
28 -0.6473127  1.35403476 1
29  0.5357084  4.11995931 1
30  1.9639820  0.26578581 1
31 -1.1141637 -3.16300000 2
32 -0.2754737  0.08535921 2
33 -0.1761440  1.38123139 2
>
```



何をやっているか

cという名の列に1を30個

```
class1<-data.frame(x=rnorm(30,mean=1,sd=1.5),y=rnorm(30,mean=1,sd=1.5),c=rep(1,30))
```

データフレームにするにあたり　xという名の列　yという名の列に平均1、標準偏差1.5の正規乱数30個

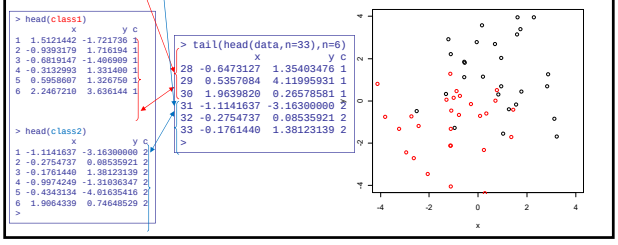
```
data<-rbind(class1,class2)
```

行方向につなげる。

```
plot(subset(data,data$c==1)[,1:2],xlim=c(-4,4),ylim=c(-4,4))

points(subset(data,data$c==2)[,1:2],col="red",xlim=c(-4,4),ylim=c(-4,4))
```

subset(data,data$c==1) は data の中で data$c==1 が成立しているものだけからなる部分集合



多数決はどうしよう？

- table() 関数を用いて、数えることにする。
  - table() は分割表（contingency table, cross tabulation, or cross tab) を作る関数

```
> x <- c(0,1,0,1,1,0,0,0)
> (tbl<-table(x))
x
0 1
5 3
> as.data.frame(tbl)
  x Freq
1 0    5
2 1    3
> (count1 <- tbl["0"])
[1] 5
> (count2 <- tbl["1"])
[1] 3
>
```

0 が5個, 1が3個

多数決

- 頻度を求め、その名前（factor の level）からクラスを表す数値を取り出したい。

```
# majority(c(0,1,2,2,0,1,0,2,1,2,0,1,1)) = 1 となるような関数
majority <- function( x ) {
  return( as.numeric(names(which.max(table(x)))) )
}
```

まとめれば k-NN

多数決関数の定義

```
# majority(c(0,1,2,2,0,1,0,2,1,2,0,1,1)) = 1 となるような関数
majority <- function( x ) {
  return( as.numeric(names(which.max(table(x)))) )
}
```

実験用データの作成

個数です

```
class1<-data.frame(x1=rnorm(30,mean=1,sd=1.5),
                   x2=rnorm(30,mean=1,sd=1.5),c=rep(1,30))
class2<-data.frame(x1=rnorm(30,mean=-1,sd=1.5),
                   x2=rnorm(30,mean=-1,sd=1.5),c=rep(2,30))
data<-rbind(class1,class2)
```

Main program

注意: これはスカラー、これはベクトル. 引算の結果はベクトル

ベクトルの要素の2乗からなるベクトル

```
# k-NN  majority
k <- 3
x1<- 0          # テスト点の x1 座標
x2<- 0          # テスト点の x2 座標
sorted <- sort(  ( x1 - data$x1 )^2 + ( x2 - data$x2 )^2, index.return=TRUE)
majority( data$c[ sorted$ix[ 1:k ] ] )
```

境界線が表示できます

```
# k of k-NN
k  <- 3

# test data which, in this case, are used for the plot
xtest1 <- seq(min(data$x1),max(data$x1),length=30)
xtest2 <- seq(min(data$x2),max(data$x2),length=30)
xtest <- expand.grid(xtest1,xtest2)

# use k-NN to get predictions on the grid
cPredicted <- rep(0,30*30)
for ( i in 1:length(xtest[,1]) ) {
  x1 <- xtest$Var1[i]
  x2 <- xtest$Var2[i]
  sorted <- sort(  ( x1 - data$x1 )^2 + ( x2 - data$x2 )^2, index.return=TRUE)
  cPredicted[i] <- majority( data$c[ sorted$ix[ 1:k ] ] )
}

# cPredicted in a matrix form to be used for "plot"
cPredictedMatrix <- matrix( cPredicted, 30, 30 )
```

ここから、ここまで、30等分

```
# "data" はこんなつもり
> head(data)
         x1         x2 c
1  1.8678381  0.1878979 1
2  2.7810833 -0.7073069 1
3 -0.3151720  0.8773681 1
4 -1.0610735 -0.7644417 1
5 -0.3411839  0.6615055 1
6 -1.5258154  1.6463212 1

 > head(xtest)
        Var1      Var2
1 -3.941878 -4.142661
2 -3.700731 -4.142661
3 -3.459584 -4.142661
 ……
```

何を行おうとしているか？へのヒント

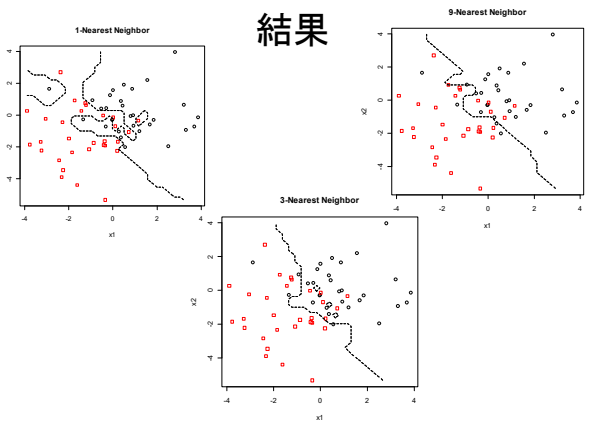
```
plot(subset(data,data$c==1)[,1:2],xlim=c(-4,4),ylim=c(-4,4))
points(subset(data,data$c==2)[,1:2],col="red",xlim=c(-4,4),ylim=c(-4,4))
points(xtest,pch=".",col="blue")
```

プログラム（続）

```
# plot "data"
plot(data[1:2],
  xlab="x1", ylab="x2",                    # labels of axes
  main=paste(k,"-Nearest Neighbor",sep=""), # title of the graph
  col=data$c, pch=as.double(data$c)+20 )   # color and characters to be used

# draw boundaries
for(i in 1:max(cPredicted))
  contour(
    xtest1, xtest2, cPredictedMatrix==i,   # x, y, and z where z is 0 or 1
    levels=0.5,                            # levels=0.5 i.e. the middle of 0 and 1
    drawlabels=FALSE, add=TRUE,
    col=1, lty=3)                          # color is black and line type is broken
```

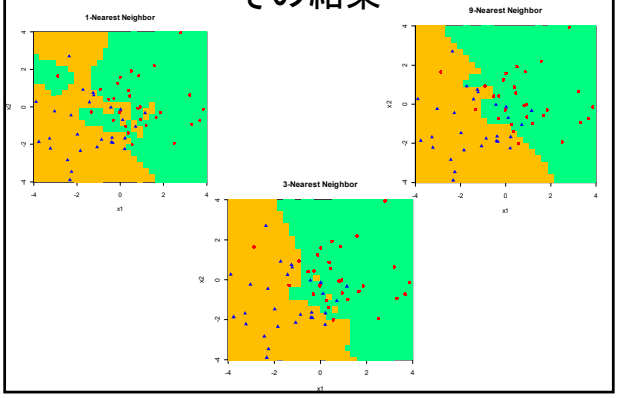
結果



別のプロット

```
# another way to plot
# the following is for just two classes
image( xtest1, xtest2, cPredictedMatrix==2,
       col=c("#00FF80","#FFBF00"), axes=TRUE,
       xlim=c(-4,4),ylim=c(-4,4),
       xlab="x1",ylab="x2",main=paste(k,"-Nearest Neighbor",sep="") )
points( data[1:2], col=ifelse(data$c==1,"red","blue"),
                   pch=ifelse(data$c==1,16,17) )
```

その結果



まとめにかえて（後半）

目次

- どの「穴埋め」か？
- k-NN（k nearest neighbor, k 最近傍法）とは
- なぜ「境界」がでこぼこか？
- k はどう決めるか

解いた問題

- 穴埋め問題1
- 穴埋め問題2-3
- 説明変数値が
  - 知っているものと同じか
  - かなり近いときに、

  被説明変数の値を答えにする

  という方法

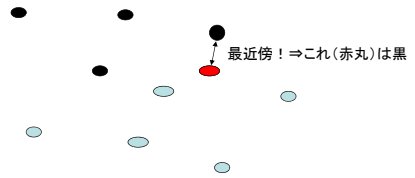
目次

- どの「穴埋め」か？
- k-NN（k nearest neighbor, k 最近傍法）とは
- なぜ「境界」がでこぼこか？
- k はどう決めるか

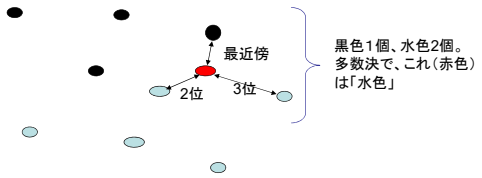
最近傍法

- 最近傍法（Nearest neighbor）
  - 問合せx_q に対し、最近接のx_nを見つけ、
    $f(x_q) \leftarrow f(x_n)$ と（f を近似）する
- k-Nearest neighbor
  - k個の最近接データの間で、多数決、または
  - k個の最近接データの間で、平均値、または、、、

1-Nearest Neighbor



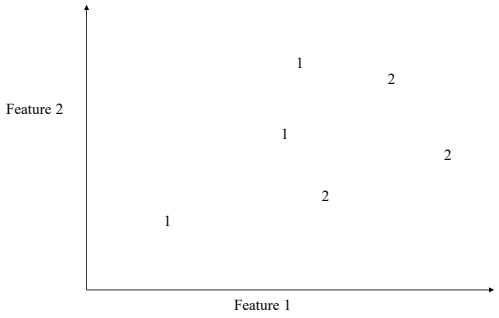
3-Nearest Neighbor



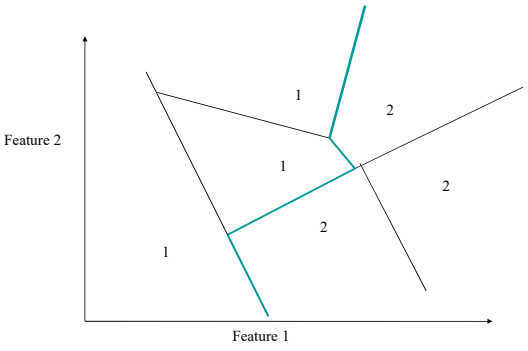
目次

- どの「穴埋め」か？
- k-NN（k nearest neighbor, k 最近傍法）とは
- なぜ「境界」がでこぼこか？
- k はどう決めるか

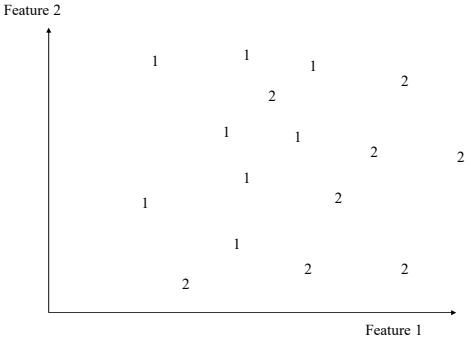
1-NNの境界線の幾何的解釈



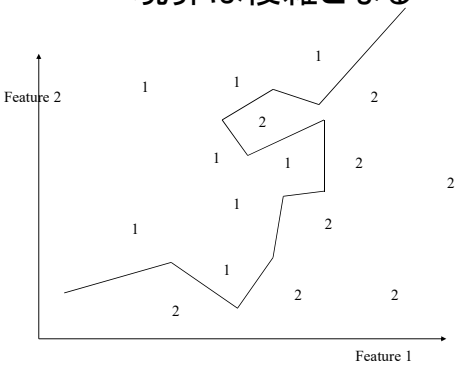
境界



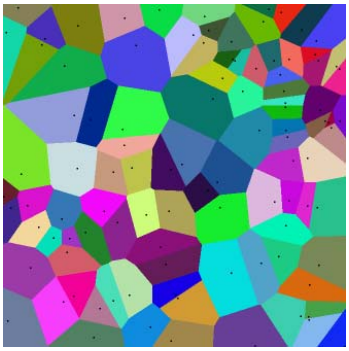
データ点が多いとき



境界は複雑となる



Voronoi図



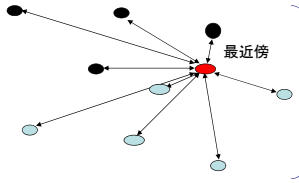
目次

- どの「穴埋め」か？
- k-NN（k nearest neighbor, k 最近傍法）とは
- なぜ「境界」がでこぼこか？
- k はどう決めるか

k が大きすぎると

9-Nearest Neighbor

四年前、いい点に気が付いた人がいました。



黒色4個、水色5個。
多数決で、これ（赤色）は「水色」

これって、赤丸をどこにおいても成立するのでは？

その通り！

つまり、全員に意見を聞くと、各員の答えはいつも同じだから、答えはいつも（どこで聞いても）同じ

k の選択

- こうしたパラメータの設定（この講義でもこれから頻繁に出てくる）は、一般に、難しい
- 理論的根拠をもって、または、データ（validation data set）を用いて決めることになる
- k-NNについては、bootstrapによる方法が提案されている

Peter Hall, Byeong U. Park, and Richard J. Samworth.
Choice of neighbor order in nearest-neighbor classification.
The Annals of Statistics, 2008, Vol. 36, No. 5, 2135–2152.

即レポ2

練習問題

- この講義で作成した k-NN を用いて、データもこの講義で作成したものを用いて、k を大きくしたとき、境界は、(1,1) と (-1,-1) の垂直2等分線に近くなるか？
  - k を変えてみるだけでなく、データ数も変えてみよ。
  - なぜ「この特別な2点の垂直二等分線」なのだろうか、その理由を考えてみてください。

一昨昨昨年、全然垂直2等分線らしくないのに、「垂直2等分線になりました」とレポートに書いた人がいました。これは「白を黒という」に等しい。