KYOTO UNIVERSITY

アルゴリズムとデータ構造⑧

～ 近傍探索 ～

鹿島久嗣

DEPARTMENT OF INTELLIGENCE SCIENCE AND TECHNOLOGY

近傍探索：
あるデータに最も類似するデータを探すためのデータ構造

- ボロノイ図
- k-d木
- ランダム射影

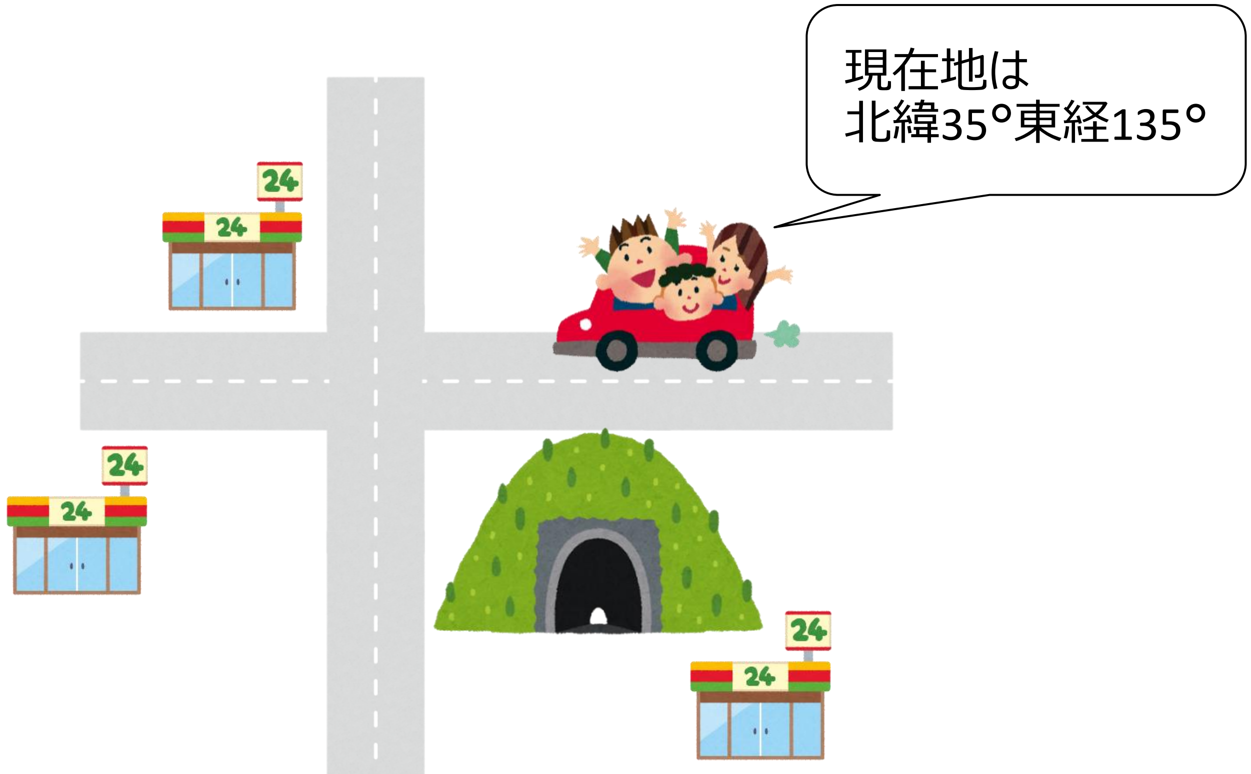
近傍探索問題

近傍探索：類似データを探す問題

- これまで考えてきた探索問題は、質問と同一のキーをもつデータを探す問題
- 近傍探索：質問に類似したデータを探す
  - 最近傍探索：最も類似したデータを探す
  - もしくは一定度以上類似したデータを探す
- たとえば：
  - 近隣施設の検索（地理情報システム）
  - 文字認識（パターン認識）

最近傍探索の例：近隣施設の検索

- 「OK、G☺☺gle。最寄りのコンビニはどこ？」



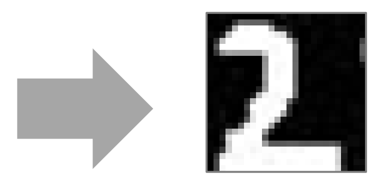
手書き文字認識：
過去の手書き文字データをもとに認識する

- 手書き数字認識：手書きの数字を読み込み０～９の数字を認識する

- 正解の分かっている手書き数字データをもとに、新たな手書き数字を認識する

収集した手書き文字データ

これは何？



手書き数字の表現:
手書き数字を多次元のベクトルとして表現

- 各数字を $16\times20=320$次元の2値ベクトルとして表現

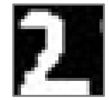
```
0,0,@,1,1,1,1,0,0,0,0,0,0,0,0,0
0,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0
1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0
1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0
0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0
0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0
0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0
0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0
0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0
0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0
0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0
0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0
0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0
0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0
0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0
0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0
0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
```

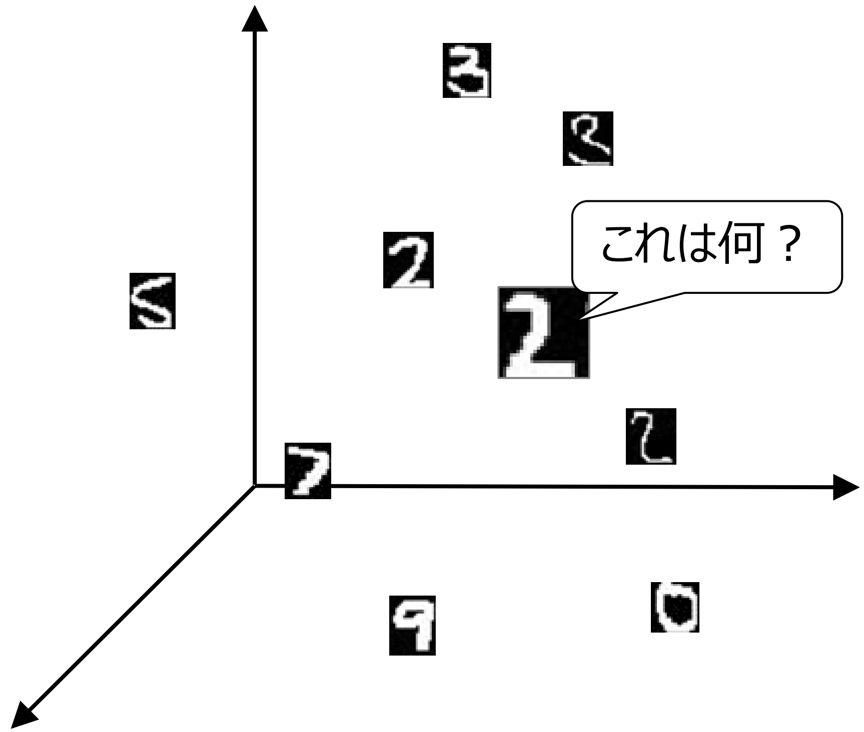
```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1
0,0,0,0,1,0,0,1,1,1,1,1,1,1,1,1
0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1
0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0
0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0
0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0
0,1,1,1,0,1,1,1,0,1,1,1,0,0,0,0
0,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0
0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0
0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0
0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0
0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0
0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0
0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0
0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0
0,1,1,0,0,0,0,0,0,0,1,1,1,1,1,0
0,1,1,1,1,1,0,0,0,0,1,1,1,0,0,0
0,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0
```

- ピクセルの濃淡がある場合は実数値ベクトルで表現

- あるいは局所的なパターンの出現で特徴ベクトルを構成する

最近傍識別：
過去の例の中から対象に最も近いものを見つける

- 最近傍識別：対象の手書き文字に最も近いものを多次元空間内でみつけ、その正解を利用して識別



最近傍識別：データ間の距離の定義

- 2つのベクトル $\mathbf{x} = (x_1, x_2, \dots, x_D)$ と $\mathbf{x}'$ の間の距離：

$$D(\mathbf{x}, \mathbf{x}') = \sqrt{(x_1 - x_1')^2 + \cdots + (x_D - x_D')^2} \quad \text{（ユークリッド距離）}$$

$$D(\mathbf{x}, \mathbf{x}') = |x_1 - x_1'| + \cdots + |x_D - x_D'| \quad \text{（絶対距離）}$$

$$D(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \mathbf{x}')} \quad \text{（マハラノビス距離）}$$

- k-近傍識別： $\mathbf{x}$ に近いデータ k 個で多数決をとる

ボロノイ図・ドロネー図

ボロノイ図・ドロネー図：
空間全体をn個の点のいずれに近いかで分割する

- 空間（通常2次元）全体を点集合$S = \{P_1, P_2, \dots, P_n\}$（母点とよぶ）のどれに近いかで分割したもの

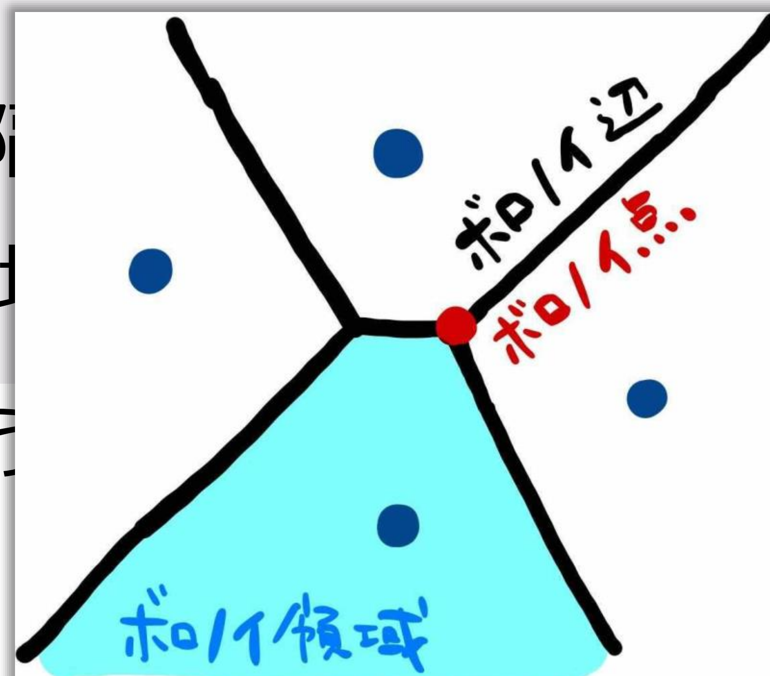
- P_iのボロノイ領域： $d(P, P_i) < d(P, P_j)$ $(i \neq j)$となるP全体（ dは空間上の距離）
  - ボロノイ辺：2つのボロノイ領域を隔てる境界辺
  - ボロノイ点：3つ以上のボロノイ領域の境界が共有する点

- ドロネー図：ボロノイ領域が隣り合う母点同志を線分で結んでできる図形

ボロノイ図・ドロネー図：
空間全体をn個の点のいずれに近いかで分割する

- 空間（通常2次元）全体を点集合$S = \{P_1, P_2, \dots, P_n\}$（母点とよぶ）のどれに近いかで分割したもの

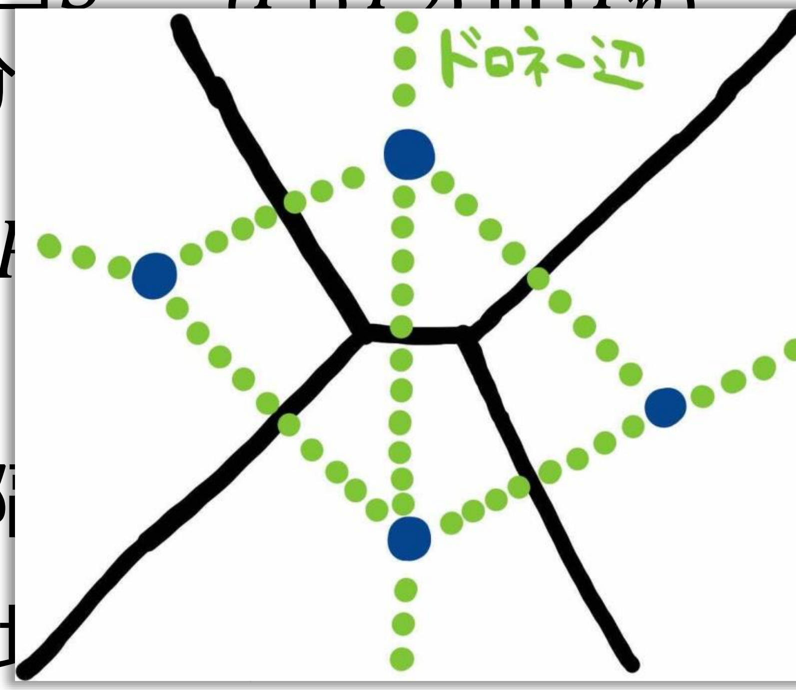
- P_iのボロノイ領域： $d(P, P_i) < d(P, P_j)$ $(i \neq j)$となるP全体（ dは空間上の距離）
  - ボロノイ辺：2つのボロノイ領域を
  - ボロノイ点：3つ以上のボロノイ領



- ドロネー図：ボロノイ領域が隣り合
んでできる図形

ボロノイ図・ドロネー図：
空間全体をn個の点のいずれに近いかで分割する

- 空間（通常2次元）全体を点集合$S = \{P_1, P_2, \dots, P_n\}$（母点とよぶ）のどれに近いかで分
- P_iのボロノイ領域： $d(P, P_i) < d($
体（ dは空間上の距離）
  - –ボロノイ辺：2つのボロノイ領域を
  - –ボロノイ点：3つ以上のボロノイ領



- ドロネー図：ボロノイ領域が隣り合う母点同志を線分で結んでできる図形

ボロノイ図を使った最近傍点探索：空間全体をn個の点のいずれに近いかで分割する

- アルゴリズム：検索点Pに対するSの最近傍点を出力

1. 任意の母点P_iを選ぶ
2. P_iに隣接する母点の中でPに最も近い点P_jを見つける
   - 隣接：ボロノイ領域が隣り合う（ドロネー辺がある）
3. $d(P, P_i) < d\left(P, P_j\right)$なら$P_i$を最近傍点として出力し終了
4. そうでなければ$P_i \leftarrow P_j$としてステップ2へ

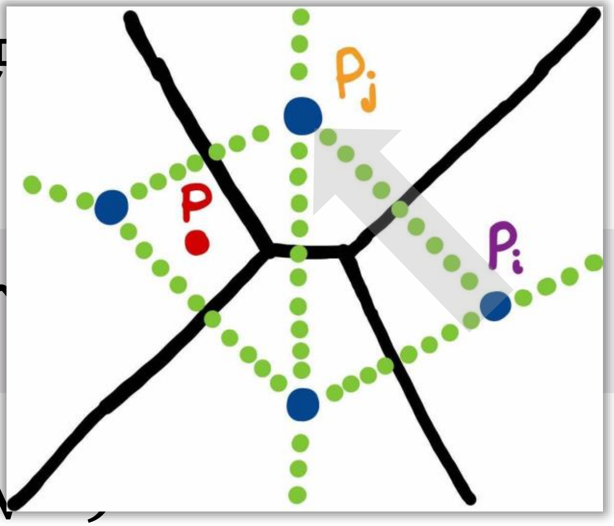
- 点が概ね一様に分布しているとして$O(\sqrt{n})$

ボロノイ図を使った最近傍点探索：空間全体をn個の点のいずれに近いかで分割する

- アルゴリズム：検索点Pに対するSの最近傍点を出力

1. 任意の母点P_iを選ぶ
2. P_iに隣接する母点の中でPに最も近い点P_jを見つける
   - 隣接：ボロノイ領域が隣り合う（ドロネー辺がある）
3. $d(P, P_i) \le d(P, P_j)$ならP_iを最近傍
   了
4. そうでなければ$P_i \leftarrow P_j$としてステップ2

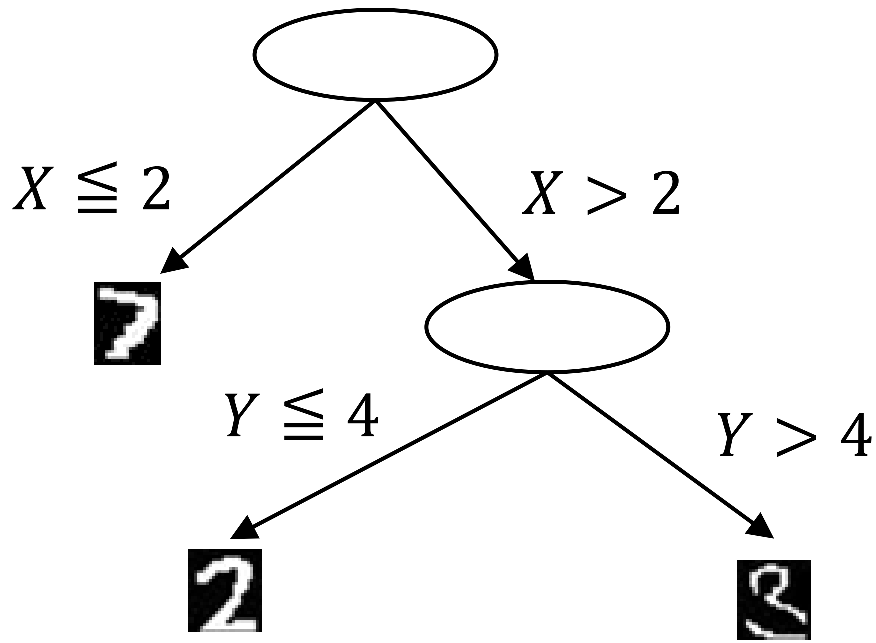
- 点が概ね一様に分布しているとして$O(\sqrt{}$



k-d木

二分木による近傍探索：空間探索のための探索木

- 空間を探索するための探索木をつくる
- 空間全体の領域分割を各次元における二分割を繰り返すことで行う

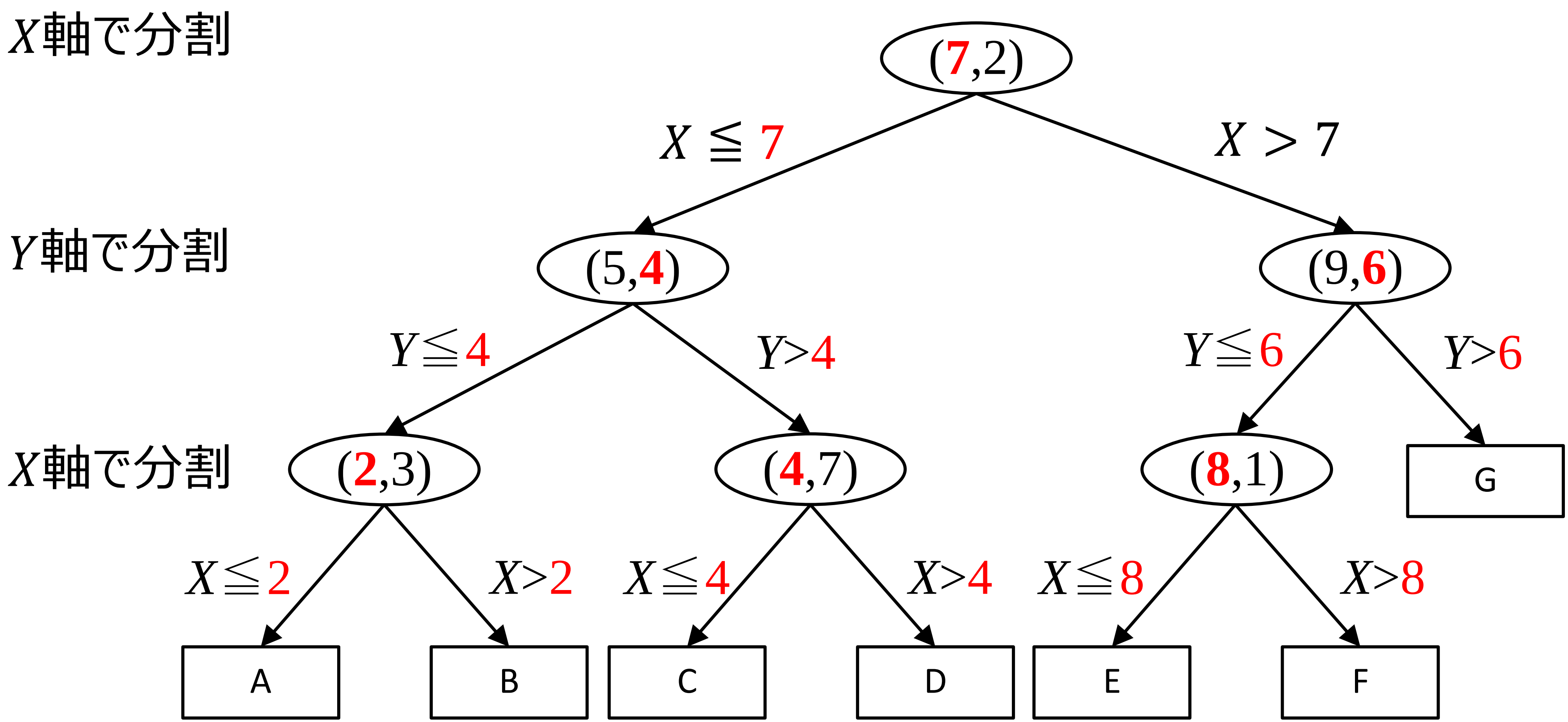


k-d木：
空間探索のための探索木

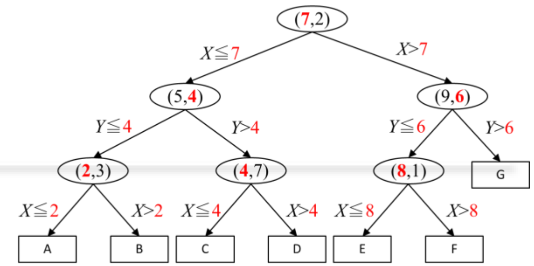
- k-d木： k次元の空間を探索するための2分探索木
- k-d木において、各々の分割はデータ点において行われる

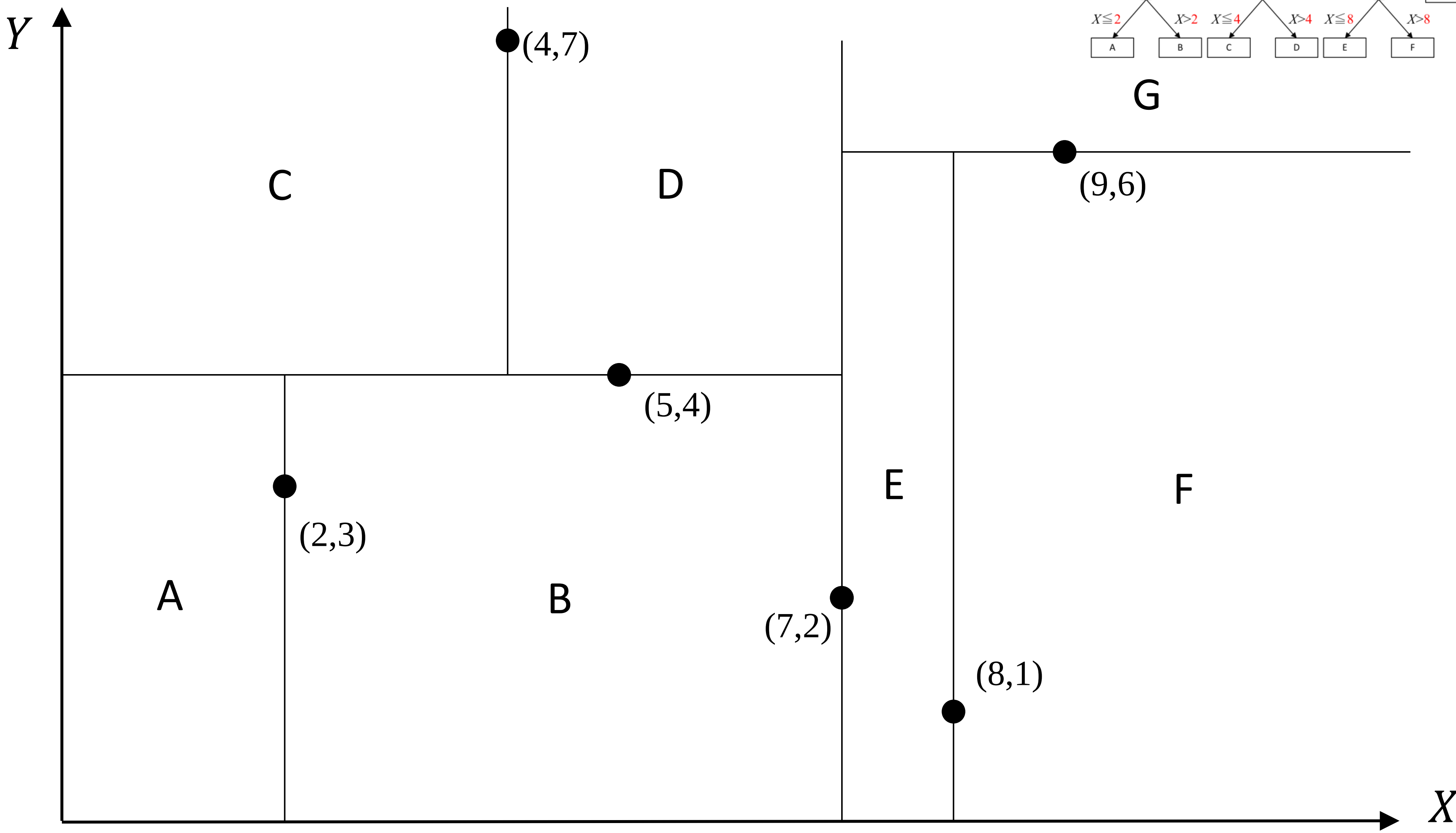
k-d木の例：空間をデータ点で領域に分割

- 例：2次元データ (2,3), (5,4), (9,6), (4,7), (8,1), (7,2)



k-d木の例：
空間をデータ点で領域に分割

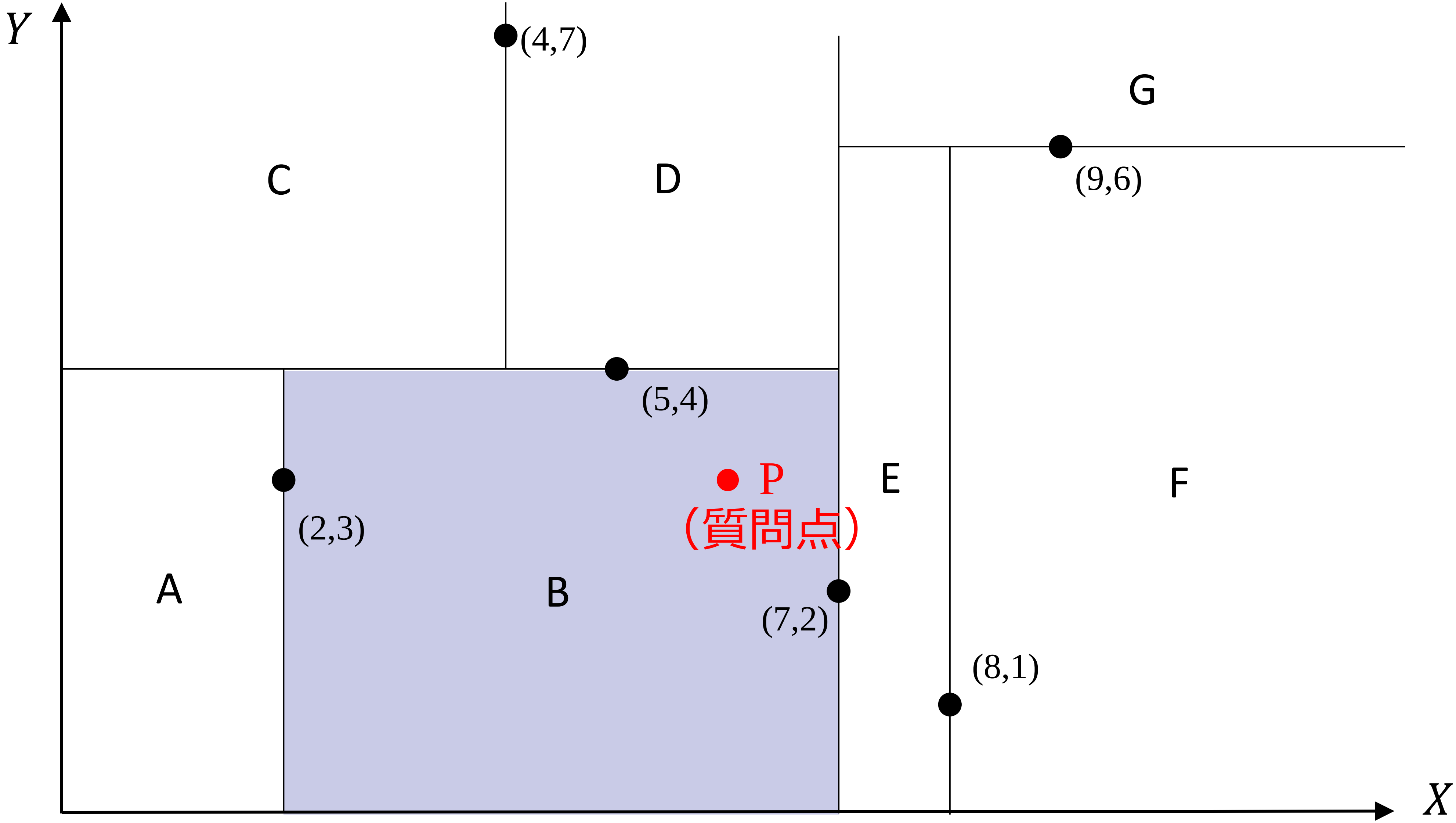




k-d木の構成：分割軸を決めて中央値で分割する

- 構成法（のひとつ）：領域を分割していく
  1. 分割する軸を選ぶ
     （例えば領域内データの分散が最大の軸）
  2. 選んだ軸について、領域に含まれるデータの中央値となるデータで分割する
  3. 再帰的に分割を繰り返す
     これ以上分割できなくなったときに終了
- 探索木は平衡木が望ましい：中央値で分割を繰り返すので、高さはおおむね $\log n$ を達成

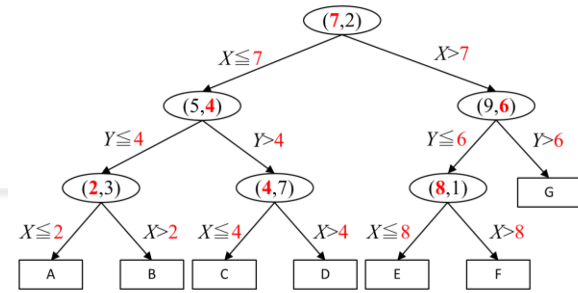
k-d木による質問点の探索：探索木を下ることで質問点を含む領域を$O(\log n)$で発見

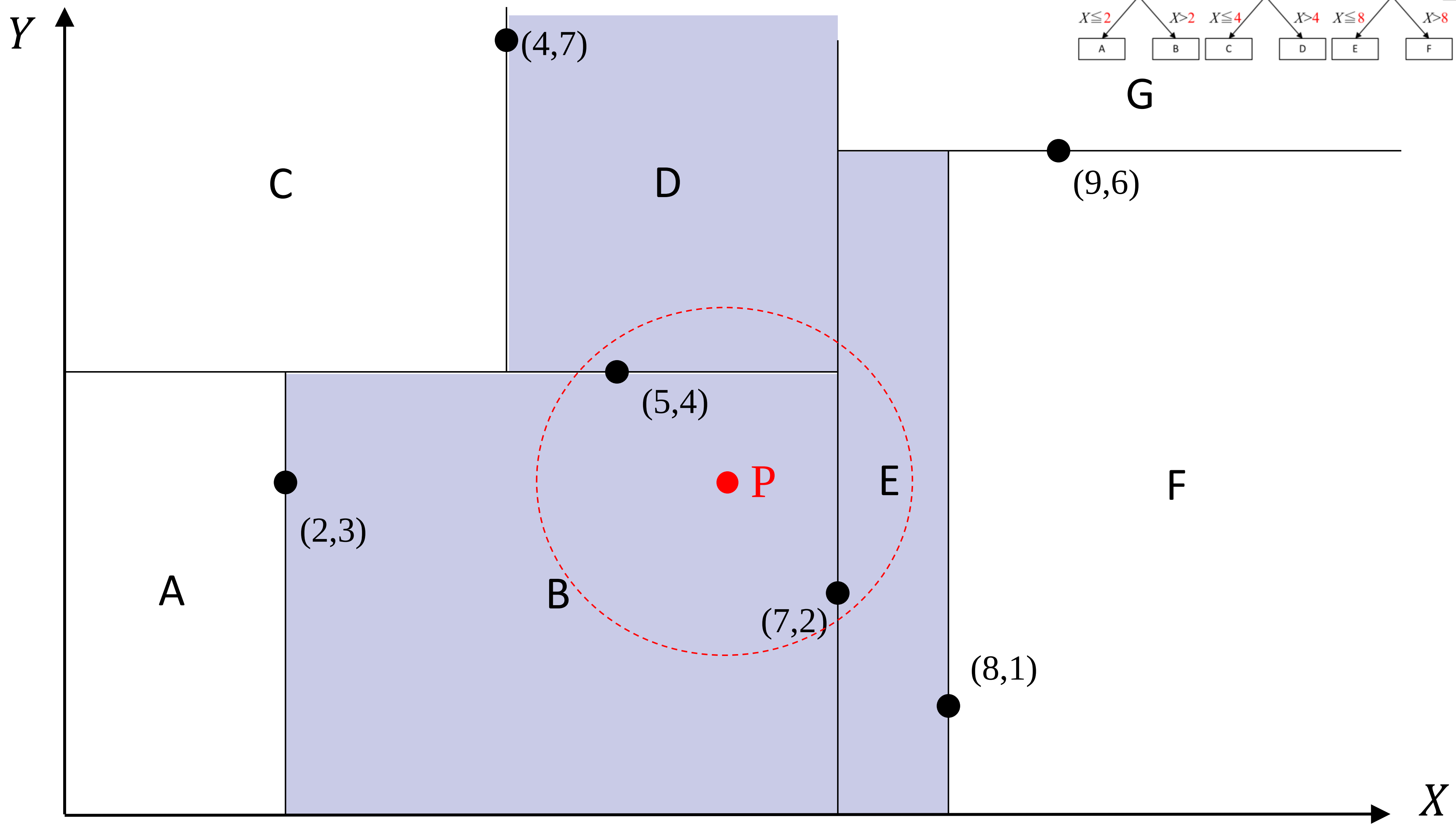


k-d木による近傍点の探索：枝刈りによって効率的に探索を行う

- 質問点Pを含む半径rの領域にデータがあるかを調べたい
- 基本方針：質問点Pを含む半径rの領域と、分割された各領域に重なりがあるかをチェックしながらk-d木を下る
- k-d木の各分岐において：質問点Pを含む半径rの領域に…
  - 分岐点が含まれれば解として記録しておく
  - 分岐先の領域が重なるなら、そちらの探索を続行する
    - 両方の分岐先の領域と重なるなら、両方探索する
    - 重ならない領域については、以降の分岐の探索は打ち切れる
- 最悪の場合、すべての分岐をチェックする必要が生じる

k-d木による近傍点の探索：枝刈りによって効率的に探索を行う

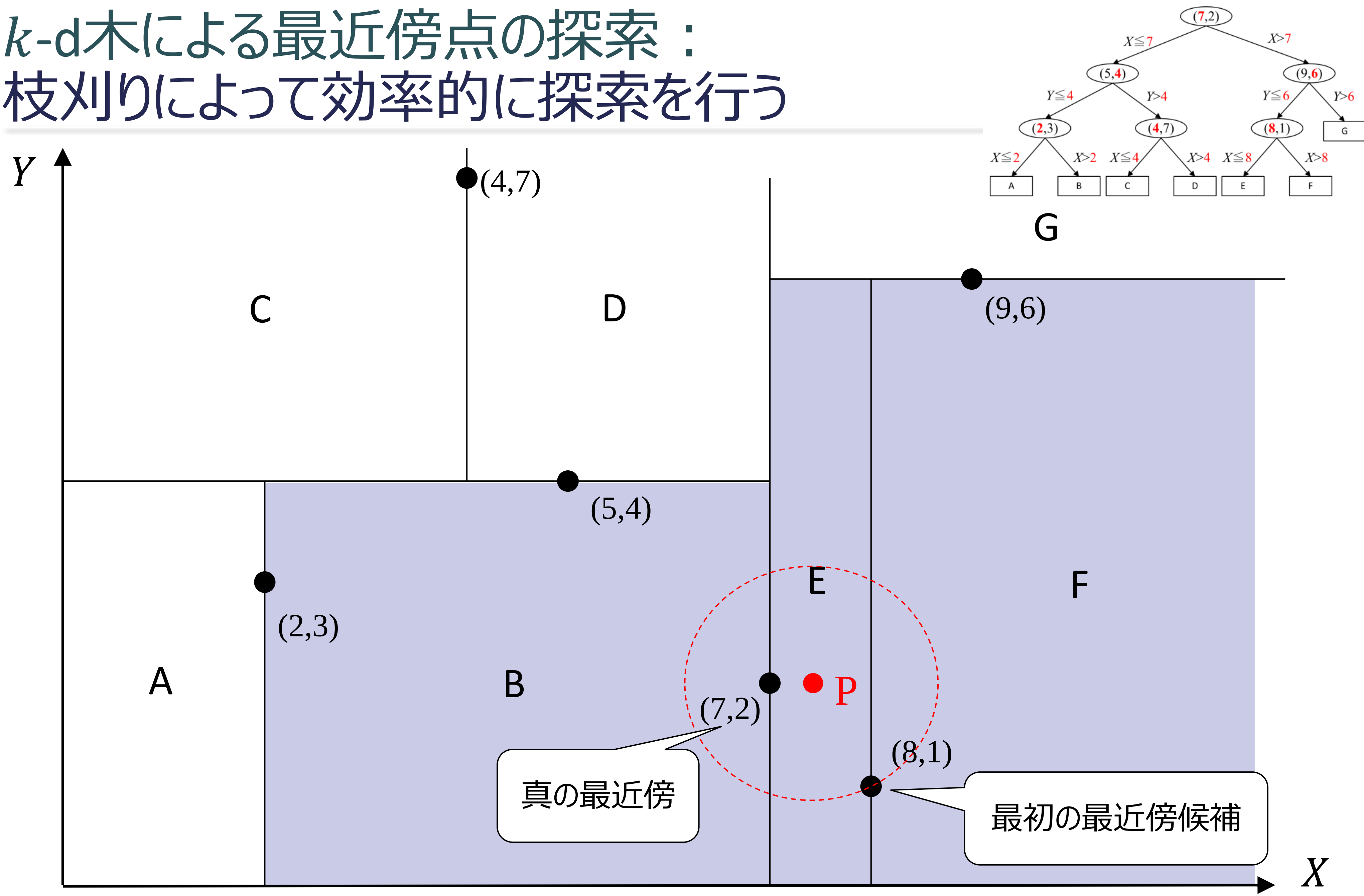




k-d木による最近傍点の探索：近傍点探索を少し変更

- Pに近い点が見つかる度に、暫定的な最近傍点P'と距離rを更新
  - 初めはPを含む領域を見つけ最も葉に近い分割点を初期P'とする
  - Pから半径r以内により近い点がなければP'が真の最近傍点
- もう少し効率のよい方法：深さ優先探索
  - 現在の領域から木を上に上がっていきながら分岐点をチェック
  - 分岐点の反対側の領域が、Pから半径r以内の領域と被っていれば、他方の分岐先も調べる
  - より近い点を見つけたら、P'とrを更新
  - 根において、探索すべき方向がなくなったら終了

k-d木による最近傍点の探索：枝刈りによって効率的に探索を行う



高次元の探索：
k-d木は高次元で効率が悪いので次元削減が必要

- k-d木が有効なのは数次元程度
  - データ数 $n \gg 2^k$ が望ましい
- 高次元の場合に、探索効率が悪くなる（多くの点を調べることになる）
- 次元削減によって次元を落としてからk-d木を適用する
  - ランダム射影
  - 主成分分析
  - …

ランダム射影

次元削減：データの低次元表現を見つける

- D次元ベクトル$\mathbf{x}$に対し$D' \times D$行列AをかけてD'次元ベクトル$\mathbf{x}'$に変換する：$\mathbf{x}' = A\mathbf{x}$
  - $D' < D$、つまり$\mathbf{x}'$の次元は$\mathbf{x}$より小さい
- Aの決め方：
  - $\mathbf{x}'$が元の$\mathbf{x}$の情報をできるだけ保持するようにAを決める
  - たとえば再構成誤差を最小化すると、主成分分析
    - $D \times D'$行列Bで再構成 $\tilde{\mathbf{x}} = B\mathbf{x}' = BA\mathbf{x}$
    - 再構成誤差$||\tilde{\mathbf{x}} - \mathbf{x}||$を最小化する$A$と$B$を決める

ランダム射影：距離を保存する次元削減

- Aをランダムに決めてみる：
  Aの各要素を平均0の一様分布や正規分布で生成

- 定理：２つのD次元ベクトル$\mathbf{x}_1, \mathbf{x}_2$の距離は、変換前後で「あまり」かわらない

$$\| \mathbf{x}_1 - \mathbf{x}_2 \|_2 \approx \sqrt{\frac{D}{D'}} \| A\mathbf{x}_1 - A\mathbf{x}_2 \|_2$$

ランダム射影木：
ランダム射影とk-d木の組み合わせ

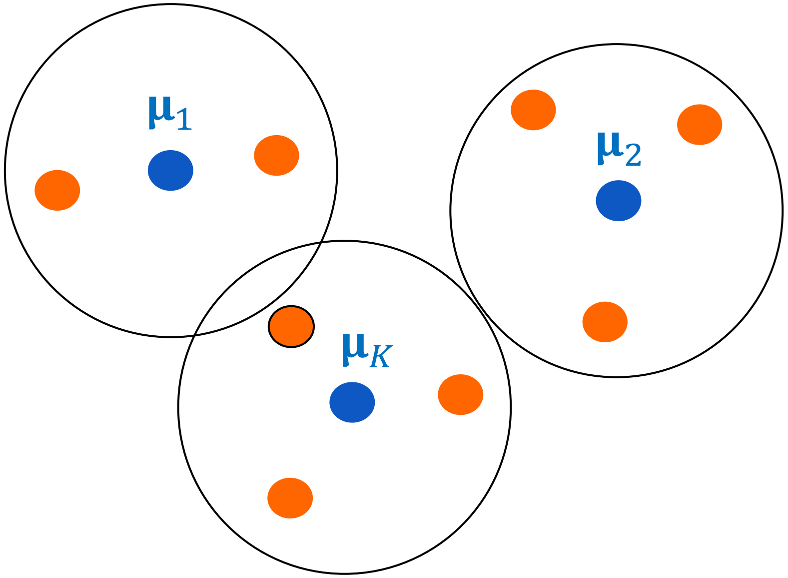
- 分割軸を、もともとの次元の一つを選ぶのではなく、1次元へのランダム射影によってつくる

- 分割のたびに新しい軸を作って分割を行う

付録：クラスタリング

クラスタリング：
データのグループ化

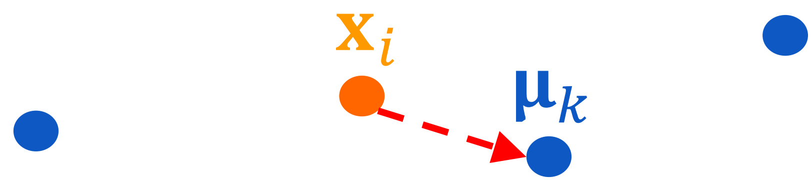
- クラスタリング：データ$\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$を$K$個のグループに分ける
  - データ解析の基本タスク
- K-平均クラスタリング：
  - 各グループをK個の代表点$\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_K\}$によってあらわす
  - 各データは最寄の代表点に所属



K-平均クラスタリングアルゴリズム：グループ割り当てと代表点推定を交互に行うアルゴリズム

- 以下のステップを収束するまで繰り返す

1. 各データ$\mathbf{x}_i$を、最寄の代表点$\boldsymbol{\mu}_k$に割り当てる



2. 各代表点に所属したデータの平均として代表点を新たに求める