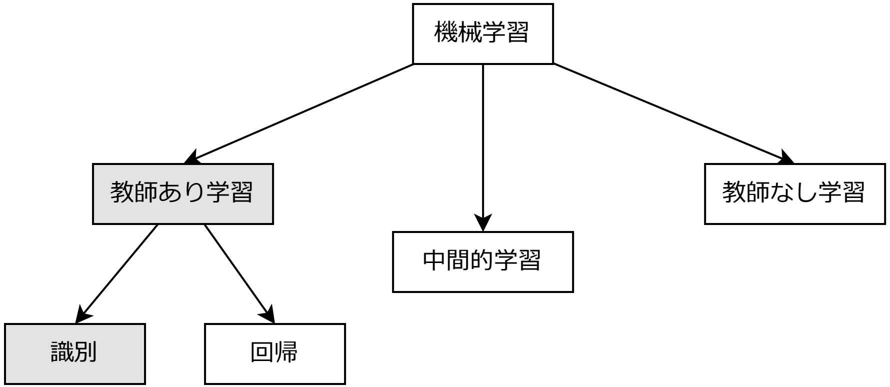
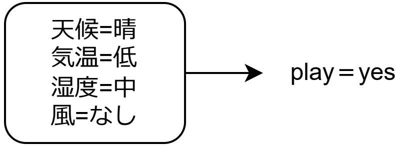
識別　ー統計的手法ー

- 第3章(決定木)：正解を表現する概念を得る(説明性が高い)
- 第4章(統計)：識別結果の確率を得る(意思決定に役立つ)



・カテゴリ特徴



weather.nominalデータ

No	outlook	temperature	humidity	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	vercast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes

weather.nominalデータ

- outlook(天候)
  - sunny, overcast, rainy
- temperature(気温)
  - hot, mild, cool
- humidity(湿度)
  - high, normal
- windy(風)
  - TRUE, FALSE
- play(=クラス)
  - yes, no

統計的識別とは

- 特徴ベクトル $\boldsymbol{x}$ が観測されていないとき
  - 事前確率 $P(\mathrm{yes}), P(\mathrm{no})$ だけから判断するしかない
- 特徴ベクトル $\boldsymbol{x}$ 観測後
  - 事後確率 $P(\mathrm{yes}|\boldsymbol{x}), P(\mathrm{no}|\boldsymbol{x})$ の大きい方に判定
- 多クラス $(\omega_i : \; i = 1, \ldots, c)$ に一般化
  - 最大事後確率則による識別（ベイズ識別）

$$C_{MAP} = \arg\max_{i} P(\omega_i|\boldsymbol{x})$$

統計的識別とは

- 事後確率の求め方
  - 単純な方法: 特徴ベクトルが完全に一致する事例を大量に集めて、その正解ラベルの割合を求める
    - 例) $\boldsymbol{x}$ =(晴, 高, 中, TRUE) 100事例中 yes:70事例, no:30事例 $\rightarrow P(\text{yes}|(\text{晴}, \text{高}, \text{中}, \text{TRUE})) = 0.7$
  - しかし、上記の推定が行えるようなデータセットが得られることはほとんどない
  - そのため、事後確率に対して式変形・近似を行って、現実の規模のデータセットから確率を推定できるようにする

統計的識別とは

- ベイズの定理

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- 事後確率を尤度と事前確率の積に変形
  - 事後確率にベイズの定理を適用
  - 最大値を求める際に無関係な分母を払う

$$C_{MAP} = \arg\max_i P(\omega_i|\boldsymbol{x}) = \arg\max_i \frac{p(\boldsymbol{x}|\omega_i)P(\omega_i)}{p(\boldsymbol{x})} = \arg\max_i p(\boldsymbol{x}|\omega_i)P(\omega_i)$$

    - 尤度：特定のクラスから、ある特徴ベクトルが出現する尤もらしさ

統計的識別とは

- ベイズ統計とは
  - 結果から原因を求める
    - 通常の統計学は原因から結果を予測する
  - ベイズ識別
    - 観測結果 $\boldsymbol{x}$ から、それが生じた原因 ω_i を求める（事後確率）
    - 通常、確率が与えられるのは　原因→結果（尤度）
    - ベイズ識別では、事前分布 $P(\omega_i)$ が、観測 $\boldsymbol{x}$ によって事後分布 $P(\omega_i|\boldsymbol{x})$ に変化したと考えることができる

カテゴリ特徴に対するベイズ識別

- 事前確率：$P(\omega_i)$
  - 特徴ベクトルを観測する前の各クラスの起きやすさ
- 事前確率の最尤推定
  - 学習データ中のクラスの割合から推定する
  - N: 全データ数、n_i: クラス ω_i のデータ数

$$P(\omega_i) = \frac{n_i}{N}$$

学習データの対数尤度

- 尤度の導出
  - 特徴ベクトル $\boldsymbol{x}$ を生成するモデルを考え、そのモデルが（クラスごとの）パラメータ $\boldsymbol{\theta}_j$ に従ってデータを生成していると仮定

$$p(\boldsymbol{x}|\omega_j;\boldsymbol{\theta}_j)$$

  - これらはクラス毎のデータから推定することになるので、以後、1クラス分のデータを全データDとみなして ω_j を省略し、$\boldsymbol{\theta}_j$ を $\boldsymbol{\theta}$ と表記
  - i.i.d. (independent and identically distributed)を仮定
    - 全データ D は、各データが同じ分布から独立に生成されていると仮定して尤度を計算

$$P(D|\boldsymbol{\theta}) = \prod_{i=1}^{N} p(\boldsymbol{x}_i|\boldsymbol{\theta})$$

学習データの対数尤度

- 対数尤度: $\mathcal{L}(D)$
  - 確率の積のアンダーフローを避けるため、尤度を対数で計算

$$\mathcal{L}(D) = \log P(D|\boldsymbol{\theta}) = \sum_{i=1}^{N} \log p(\boldsymbol{x}_i|\boldsymbol{\theta})$$

- 尤度関数の仮定の例
  - 特徴ベクトルが1次元、値0 or 1で、ベルヌーイ分布に従うと仮定
    - ベルヌーイ分布：確率 θ で値1、確率 $1-\theta$ で値0をとる分布

$$\mathcal{L}(D) = \sum_{i=1}^{N} \log \theta^{x_i} (1-\theta)^{(1-x_i)} = \sum_{i=1}^{N} x_i \log \theta + (N - \sum_{i=1}^{N} x_i) \log(1-\theta)$$

学習データの対数尤度

- 対数尤度を最大にするパラメータ:$\hat{\theta}$
  - $\frac{\partial \mathcal{L}(D)}{\partial \theta} = 0$ の解である $\hat{\theta}$ を求める

$$\begin{aligned}\frac{\partial \mathcal{L}(D)}{\partial \theta} &= \sum_{i=1}^{N} x_i \frac{1}{\theta} + (N - \sum_{i=1}^{N} x_i)\frac{1}{1-\theta} \\ &= \frac{1}{\theta(1-\theta)}\{(1-\theta)\sum_{i=1}^{N} x_i - \theta(N - \sum_{i=1}^{N} x_i)\} = 0\end{aligned}$$

$$\hat{\theta} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

  - 値1がでる確率の最尤推定値として、値1がでた回数を全データ数 N で割ったものが得られた

ナイーブベイス識別

- 多次元ベクトルの尤度関数を求める
  - 特徴値のすべての組合せが、データセット中に何度も出てくる必要があるが、これも非現実的
- ナイーブベイズの近似
  - すべての特徴が独立であると仮定

$$P(\boldsymbol{x}|\omega_i) = P(x_1, \ldots, x_d|\omega_i) \approx \prod_{k=1}^{d} P(x_k|\omega_i)$$

$$C_{NB} = \arg\max_i P(\omega_i) \prod_{k=1}^{d} P(x_k|\omega_i)$$

ナイーブベイス識別

- 尤度の最尤推定
  - n_j : クラス ω_j のデータ数
  - n_k : クラス ω_j のデータのうち、k 次元目の値が x_k であるデータ数

$$P(x_k \mid \omega_j) = \frac{n_k}{n_j}$$

- ゼロ頻度問題: n_k が0の場合、確率の推定値も0となってしまう
  - 解決法 → スムージング
    - m 種類の k 次元目の値が、事前に α 回ずつ生じていたと仮定する
    - $\alpha = 1$ のときをラプラス推定とよぶ

$$P(x_k \mid \omega_j) = \frac{n_k + \alpha}{n_j + \alpha m}$$

scikit-learnのナイーブベイズ識別

- カテゴリ特徴は `OrdinalEncoder` で整数値に置き換える
  - 変換情報

```
enc = OrdinalEncoder()
X_en = enc.fit_transform(X)
enc.categories_
```

```
[array(['overcast', 'rainy', 'sunny'], dtype=object),
 array(['cool', 'hot', 'mild'], dtype=object),
 array(['high', 'normal'], dtype=object),
 array([False, True], dtype=object)]
```

- 変換例
  - [’sunny’, ’hot’, ‘high’ , False] $\rightarrow$ [2, 1, 0, 0]
  - 異なる値に異なる整数値を割り当てているだけであって、数値の近さが概念の近さを表しているのではないことに注意

scikit-learnのナイーブベイズ識別

- 正解のラベルは `` `LabelEncoder` `` で整数値に置き換える

```
le = LabelEncoder()
y_en = le.fit_transform(y)
le.classes_
```

```
array(['no', 'yes'], dtype=object)
```

- no → 0, yes → 1

scikit-learnのナイーブベイズ識別

- カテゴリ特徴に対するナイーブベイズ識別は `CategoricalNB` を用いる
  - 識別器のパラメータ
    - `arpha` ：事前に仮定するサンプル数。教科書の mp に対応
    - `fit_prior` ：事前確率を学習の対象とするかどうか
    - `class_prior` ：事前確率を別途与えるときに用いる
- 典型的なコード

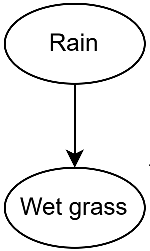
```
clf = CategoricalNB()
clf.fit(X, y)
clf.predict_proba(X_test[1])
```

ベイジアンネットワーク

- ベイジアンネットワークの仮定: 変数の部分集合が、ある分類値のもとで独立である
  - $Parents(X_k)$ は値 x_k をとるノードの親ノード集合の値

$$P(x_1, \ldots, x_d) \approx \prod_{k=1}^{d} P(x_k | Parents(X_k))$$

- 条件付き確率のベイジアンネットワークによる表現



$P(R) = 0.4$

$P(W|R) = 0.9$

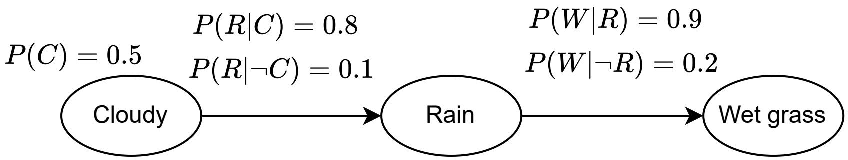
$P(W|\neg R) = 0.2$

ベイジアンネットワーク

- 変数間の独立性を表す基本パターン
  - Head-to-tail
  - Tail-to-tail
  - Head-to-head

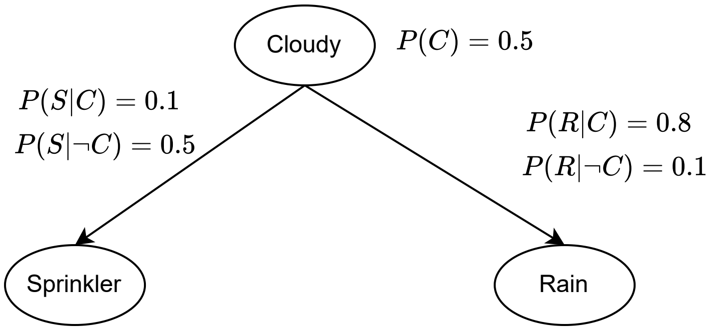
ベイジアンネットワーク

- Head-to-tail
  - 真ん中のノードの値が与えられると、左右のノードは独立



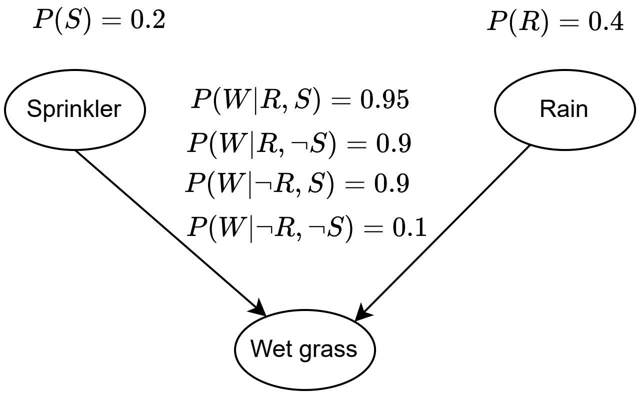
ベイジアンネットワーク

- Tail-to-tail
  - 親ノードの値が与えられると、子ノードどうしは独立



ベイジアンネットワーク

- Head-to-head
  - 子ノードの値が与えられると、親ノードどうしが独立でなくなる



ベイジアンネットワーク

- 確率計算
  - 正確な計算：周辺化によって、すべての値に対する確率を合計する
  - 近似計算：乱数を用いてベイジアンネットワークの確率分布に従った事例を生成し、確率を推定する

$$P(y|x_1,...,x_d) \approx \frac{C(y,x_1,...,x_d)}{C(x_1,...,x_d)}$$

ベイジアンネットワーク

- ベイジアンネットワークの学習

```
ノードの順番を決める（通常はクラスを表す特徴を最初に）
repeat
  for n in Node:
    for n' in n+1以降のノード:
      if (n から n' へのアークを追加することにより対数尤度が増加}
        n から n' へアークを追加
  ノードの順番を変える
until 対数尤度が変化しない
return 学習されたベイジアンネットワーク
```

まとめ

- カテゴリ特徴の識別問題に対する統計的識別
  - ベイズ識別
    - 事後確率 $P(\omega_i|\boldsymbol{x})$ を最大とするクラス ω_i を求める
    - 事後確率をデータから推定するのは難しいので、ベイズの定理を用いて尤度 $P(\boldsymbol{x}|\omega_i)$ と事前確率 $P(\omega_i)$ の積に分解
  - ナイーブベイズ法
    - 特徴のすべての次元が独立であると仮定して、尤度をそれぞれの次元の確率の積に分解
    - 確率が0となることを避けるためにスムージングを行う
- ベイジアンネットワーク
  - 変数の部分集合が、ある分類値のもとで独立であるとして構造を推定