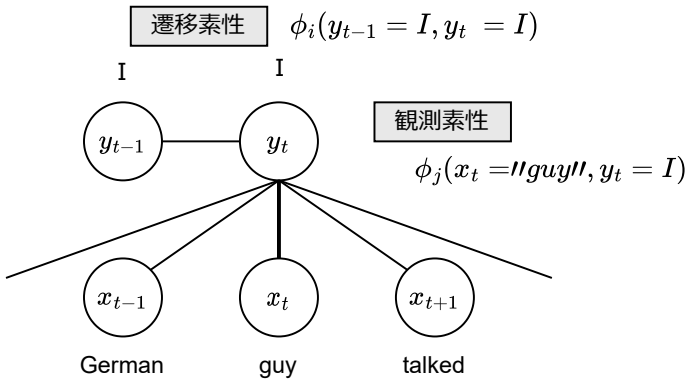
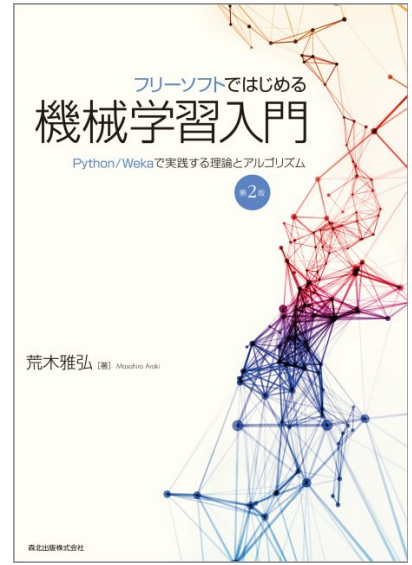
13. 系列データの識別



- 13.1 ラベル系列に対する識別
- 13.2 系列ラベリング問題
- 13.3 系列識別問題
- 系列変換問題
- 時系列信号の予測



- 荒木雅弘:『フリーソフトではじめる機械学習入門(第2版)』(森北出版, 2018年)
- スライドとJupyter notebook
- サポートページ

13.1 ラベル系列に対する識別

- ラベル系列に対する識別問題の分類
  - 入力の系列長と出力の系列長が等しい
    - 例）形態素解析、固有表現抽出
    - 系列ラベリング問題　⇒ CRF
  - 入力の系列長に関わらず出力の系列長が1
    - 例）動画像の分類、文書分類
    - 系列識別問題　⇒ HMM, RNN
  - 入力の系列長と出力の系列長に対応関係がない
    - 例）連続音声認識、機械翻訳
    - 系列変換問題 ⇒ Seq2Seq+Attention, Transformer

13.2 系列ラベリング問題 — CRF —（1/7）

- 系列ラベリング問題とは
  - 入力系列の個々の要素に対して出力ラベルを付与する
  - 出力ラベルの出現確率は前後の出力ラベル系列に依存
    - 1入力1出力の識別器を連続的に適用する方法では性能が低い

      ⇒ 入力や出力の系列としての特徴を使う
  - 可能な出力ラベル系列の数は膨大
    - すべての出力ラベル系列をリストアップすることは現実的に不可能

      ⇒ 探索によって（準）最適解を求める

13.2 系列ラベリング問題 — CRF —（2/7）

- 系列ラベリング問題の事例
  - 形態素解析

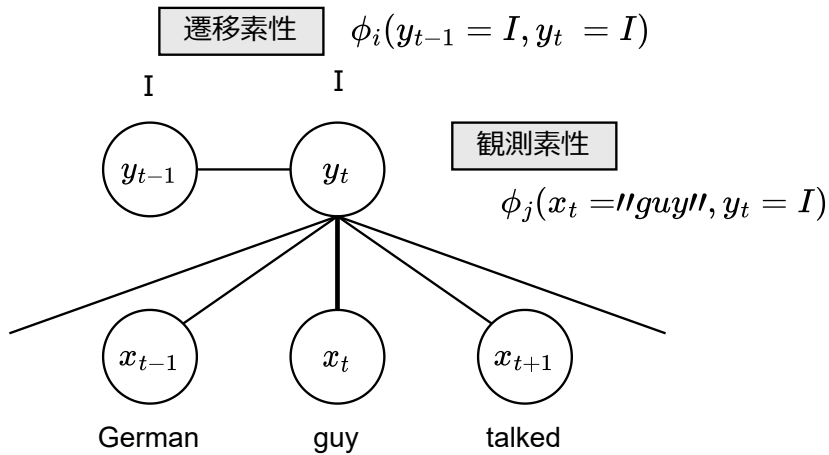
入力	系列	で	入力	さ	れる	各	要素
出力	名詞	助詞	名詞	動詞	接尾辞	接頭辞	名詞

  - 固有表現抽出
    - B: begin, I: inside, O: outside

入力	Apple	is	looking	at	buying	U.K.	startup	for	$1	billion
出力	B-ORG	O	O	O	O	B-GPE	O	O	B-MONEY	I-MONEY

13.2 系列ラベリング問題 — CRF —（3/7）

- 対数線型モデルによる系列ラベリング
  - 素性関数の導入
    - 入力系列 $\boldsymbol{x}$ と出力系列 $\boldsymbol{y}$ との間に定義される関数 $\phi(\boldsymbol{x}, \boldsymbol{y})$
      - $\boldsymbol{x}$, $\boldsymbol{y}$ のうち、ϕ の計算に関与しない要素は省略
    - ϕ で定義する関係が成立すれば1、不成立なら0を値とする



13.2 系列ラベリング問題 — CRF —（4/7）

- 対数線型モデル（多クラスロジスティック回帰）

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z_{\boldsymbol{x},\boldsymbol{w}}} \exp(\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}))$$

$$Z_{\boldsymbol{x},\boldsymbol{w}} = \sum_{\boldsymbol{y}} \exp(\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}))$$

- 出力の決定

$$\begin{aligned}
\boldsymbol{y}^* &= \arg\max_{\boldsymbol{y}} P(\boldsymbol{y}|\boldsymbol{x}) \\
&= \arg\max_{\boldsymbol{y}} \frac{1}{Z_{\boldsymbol{x},\boldsymbol{w}}} \exp(\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})) \\
&= \arg\max_{\boldsymbol{y}} \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})
\end{aligned}$$

13.2 系列ラベリング問題 — CRF —（5/7）

- 素性関数の制限：出力系列を隣接するものに限定
  - ビタビアルゴリズムによって探索が可能

$$\boldsymbol{y}^* = \arg\max_{\boldsymbol{y}} \sum_t \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, y_t, y_{t-1})$$

13.2 系列ラベリング問題 — CRF —（6/7）

- ビタビアルゴリズム

1. すべての出力記号について先頭の要素(y_1)についてのスコアを計算

$$\alpha(1, y_1) = \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, y_1)$$

2. $t = 2$ から始めて最後の出力要素に至るまで以下を繰り返し

$$\alpha(t, y_t) = \max_{y_t} \{\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, y_t, y_{t-1})\}$$

$$B(t, y_t) = \arg\max_{y_{t-1}} \{\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}, y_t, y_{t-1})\}$$

3. 最終要素での最大値に対応するBを先頭まで遡る

13.2 系列ラベリング問題 — CRF —（7/7）

- CRFの学習
  - 基本的には多クラスロジスティック回帰と同様の手順
    - 勾配降下法などでクロスエントロピーを最小化する
    - L1, L2 の正則化項も導入可能
  - 系列に対する拡張
    - ある時点での重みの更新は、その前後の系列の出力確率に影響を与えるが、その計算を動的計画法で行う（forward-backward アルゴリズム）

13.3 系列識別問題 — HMM —（1/4）

- 系列識別問題の事例
  - PC操作系列による熟練度の判定
    - k: キーボード、g: マウス、e: エラー
    - 初心者の入力系列例
      - ｋ ｅ ｋ ｇ ｋ ｅ ｋ ｇ ｇ ｋ ｇ ｋ ｋ ｅ ｇ ｅ ｅ ｋ ｅ ｅ ｅ ｇ ｅ
      - キーボード、マウスの操作が行き来し、後半になるほど疲労してエラーが多い
    - 熟練者の入力系列例
      - ｋ ｋ ｅ ｋ ｇ ｋ ｋ ｋ ｅ ｋ ｇ ｋ ｇ ｇ ｇ ｅ ｇ ｋ ｇ
      - キーボード、マウスの操作が比較的集中して効率が良く、エラーが少ない
    - 識別したい入力系列
      - ｋ ｇ ｅ ｋ ｇ ｋ ｋ ｇ ｅ ｋ ｇ ｅ ｋ ｅ ｅ ｋ ｅ ｇ ｅ ｋ
      - これは初心者／熟練者のどちらか

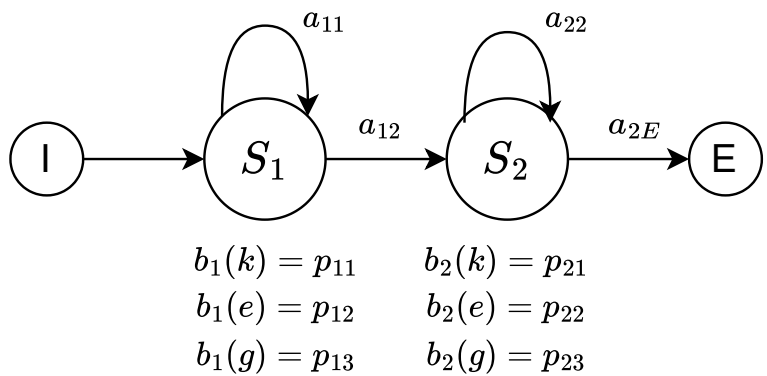
13.3 系列識別問題 — HMM —（2/4）

- 生成モデルによるアプローチ
  - 事後確率 $P(y|\boldsymbol{x})$ が最大となるクラスを識別結果とする
  - 事後確率をデータから直接的に推定することは難しいので、ベイズの定理を用いて尤度と事前確率の積に変形する
  - 系列識別問題ではクラスの事前確率$P(y)$が得られることが多い

$$
\begin{aligned}
y^* &= \arg\max_y P(y|\boldsymbol{x}) \\
&= \arg\max_y \frac{P(\boldsymbol{x}, y)}{P(\boldsymbol{x})} \\
&= \arg\max_y \frac{P(\boldsymbol{x}|y)P(y)}{P(\boldsymbol{x})} \\
&= \arg\max_y P(\boldsymbol{x}|y)P(y)
\end{aligned}
$$

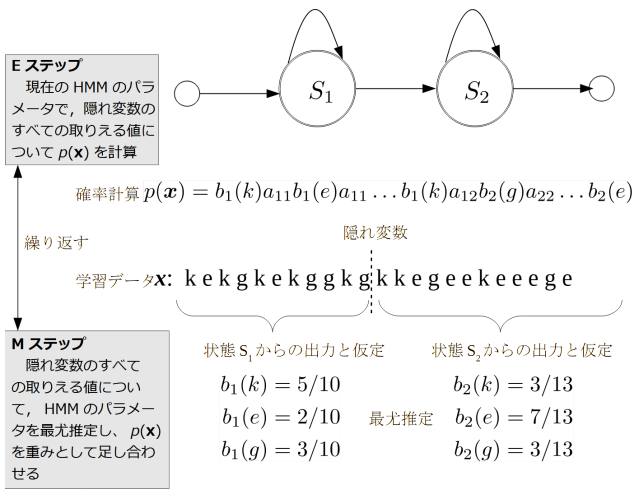
13.3 系列識別問題 — HMM —（3/4）

- 不定長入力に対する尤度計算法
  - 自己遷移を持つ確率オートマトンを用いる
    - I：初期状態、E：終了状態
  - 尤度計算はビタビアルゴリズム



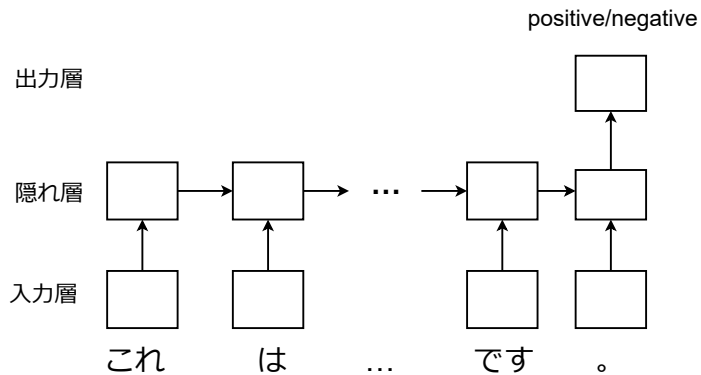
13.3 系列識別問題 — HMM —（4/4）

- HMMの学習：EMアルゴリズム
  - Eステップ
    - 現在のHMMのパラメータで、隠れ変数のすべての取り得る値について $p(\boldsymbol{x})$ を計算
  - Mステップ
    - 隠れ変数が取り得る値全てについてHMMのパラメータを最尤推定し、$p(\boldsymbol{x})$ を重みとして加算



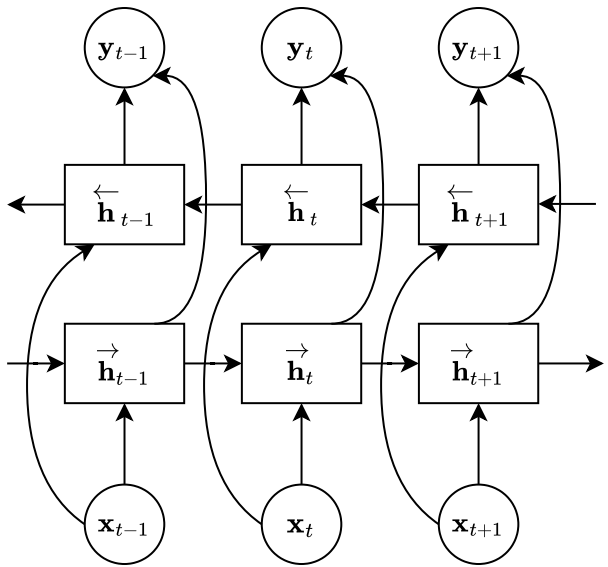
系列識別問題 — RNN —（1/2）

- RNNによる系列識別
  - 隠れ層にはLSTMやGRUを使う
  - 最終入力以外の出力は使わない
  - 最終入力に対する出力を識別結果とする
- 欠点：入力前方の情報があまり反映されない



系列識別問題 — RNN —（2/2）

- bidirectional RNNによる系列識別
  - 前向き・後向きそれぞれの最終状態を結合



系列変換問題 (1/2)

- 系列変換問題の定式化
  - 入力系列 $\boldsymbol{x} = x_1, \ldots, x_T$
  - 出力系列 $\boldsymbol{y} = y_1, \ldots, y_L$
  - 系列処理と探索を組み合わせた複雑な処理が必要
- End-to-Endアプローチ
  - 入力から出力への変換をニューラルネットワークで学習
    - CTC
    - Encoder-Decoderモデル(9章)

系列変換問題（2/2）

- CTC（Connectionist Temporal Classification）
  - 出力記号にblank記号_を加えて、入力長と出力長を合わせる
  - 正解系列に変換可能な出力系列の確率の和を求める
    - 例: /h/ /a/ /i/ という正解系列に対して、同じ音素またはblankからなる系列を1つの音素に置き換える
      - _h___a___i
      - hhh_aaaa_iii
      - hh_____a_i

時系列信号の予測 (1/7)

- 時系列信号を定常過程と仮定した場合
  - 定常過程:平均値, 分散が観測時刻によらず一定であるような時系列信号 $y_1, \ldots, y_n$
  - AR (Auto Regressive) モデル
    - 時刻 t の値 y_t が p 個前までの値の重み付き和と誤差で表現されるとするモデル

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t$$

  - MA (Moving Average) モデル
    - 時刻 t の値 y_t が q 個前までの誤差の重み付き和と現在の誤差との差で表現されるとするモデル

$$y_t = \epsilon_t - \theta_1 \epsilon_{t-1} - \theta_2 \epsilon_{t-2} - \cdots - \theta_q \epsilon_{t-q}$$

  - ARMAモデル
    - ARモデルとMAモデルを組み合わせたもの

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t - \theta_1 \epsilon_{t-1} - \theta_2 \epsilon_{t-2} - \cdots - \theta_q \epsilon_{t-q}$$

時系列信号の予測（2/7）

- 時系列信号を非定常過程と仮定した場合
  - 平均値が時間的に変動する時系列信号 $y_1, \ldots, y_n$
  - 変動をモデル化するために時系列の階差を計算する
    - 1次階差 $x_t = y_t - y_{t-1}$
    - 2次階差 $z_t = x_t - x_{t-1}$
  - ARIMA（Auto Regressive Integrated Moving Average）モデル
    - d 次階差をとった時系列信号に対する p 次のAR, q 次のMAモデル
    - 平均値の変動がなくなるまで d を上げてゆき、その時系列信号に対してARMAモデルを適用する
    - ハイパーパラメータ d, p, q は自動調整可能

時系列信号の予測（3/7）

- 時系列機械学習用ライブラリ sktime
  - https://www.sktime.org/
  - scikit-learnと共通のインタフェース
  - 時系列信号に対する予測・識別・回帰・クラスタリングをサポート

時系列信号の予測 (4/7)

- 例題:旅客機の乗客数の予測
  - Airline Passengersデータ:1949年から1960年までの月ごとの飛行機の乗客数

```
import numpy as np
import pandas as pd
from sktime.datasets import load_airline

y = load_airline()
y
```

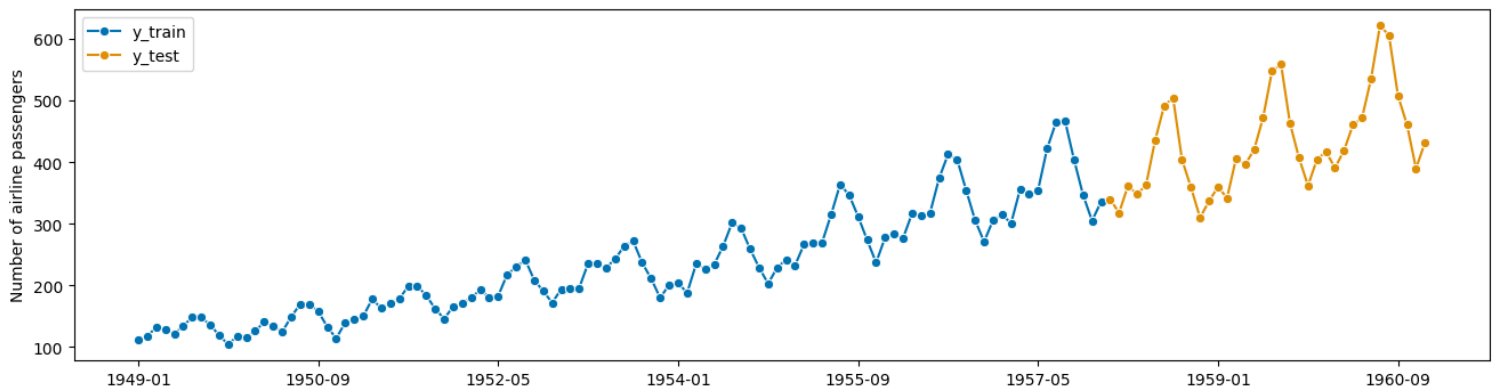
```
1949-01    112.0
1949-02    118.0
1949-03    132.0
1949-04    129.0
1949-05    121.0
           ...
1960-08    606.0
1960-09    508.0
1960-10    461.0
1960-11    390.0
1960-12    432.0
```

時系列信号の予測（5/7）

- 問題の設定
  - 直近36ヶ月のデータをテストデータ、それ以前を学習データとして予測問題を設定

```
from sktime.forecasting.model_selection import temporal_train_test_split
from sktime.utils.plotting import plot_series

y_train, y_test = temporal_train_test_split(y,test_size=36)
plot_series(y_train, y_test, labels=["y_train", "y_test"])
```



時系列信号の予測（6/7）

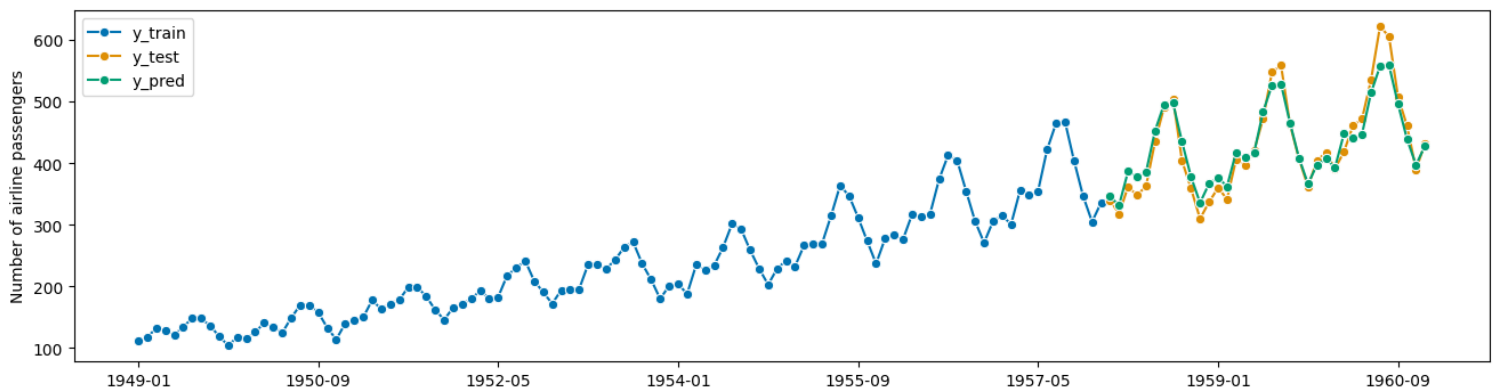
- 学習
  - 周期を表すパラメータspを1年の月数である12にしてAutoARIMAのインスタンスを作成し、学習

```
from sktime.forecasting.arima import AutoARIMA
forecaster = AutoARIMA(sp=12, suppress_warnings=True)
forecaster.fit(y_train)
```

時系列信号の予測（7/7）

- 予測
  - テストデータと同じ系列長だけ予測を行い、結果をプロット

```
y_pred = forecaster.predict(list(range(1, len(y_test)+1)))
plot_series(y_train, y_test, y_pred, labels=["y_train", "y_test", "y_pred"])
```



13.4 まとめ

- ラベル系列に対する識別問題
  - 入力の系列長と出力の系列長が等しい
    - 識別モデルCRFが有効
  - 入力の系列長に関わらず出力の系列長が1
    - HMMで可変長系列の処理が可能
    - RNNやTransformerも有効
  - 入力の系列長と出力の系列長に対応関係がない
    - RNNやTransformerが有効
- 時系列信号の予測
  - 自己回帰計数や誤差のインパクトをパラメータとして回帰