第7回 特徴量エンジニアリング

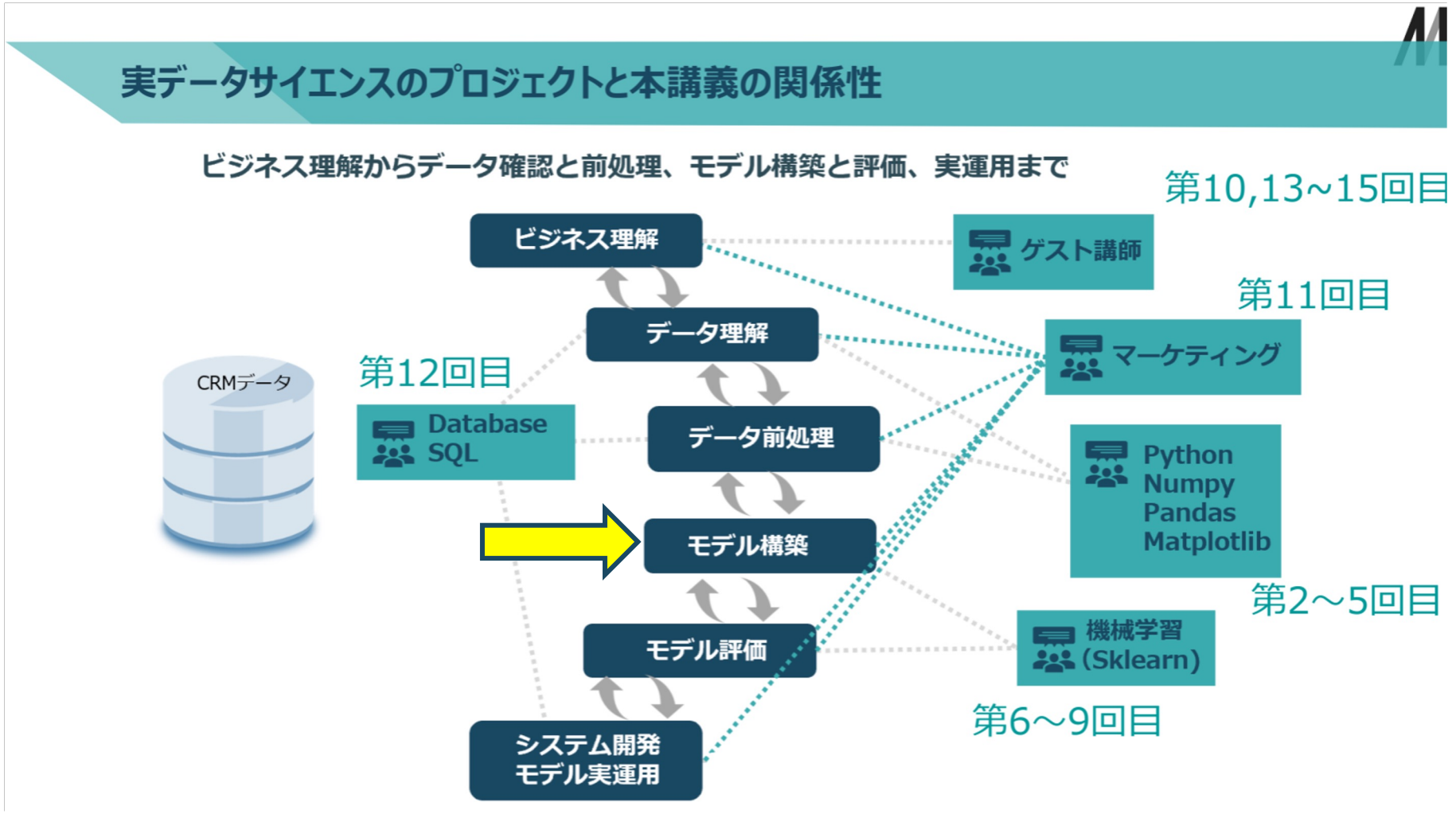
松尾・岩澤研究室
Matsuo-Iwasawa Lab UTokyo

講師：加藤

2024/11/19

特徴量エンジニアリングの位置付け

特徴量エンジニアリングは**モデルの構築部分**に位置付けられます



本日の講義の構成

本講義では特徴量エンジニアリングとして行われる代表的な処理を紹介します

1. 概要
2. 欠損値の扱い
3. 数値変数の変換
4. カテゴリカル変数の変換
5. 時系列データの扱い
6. 特徴選択

本日の講義の構成

本講義では特徴量エンジニアリングとして行われる代表的な処理を紹介します

1. 概要
2. 欠損値の扱い
3. 数値変数の変換
4. カテゴリカル変数の変換
5. 時系列データの扱い
6. 特徴選択

特徴量エンジニアリングとは

特徴量エンジニアリングとは、
生データを加工して、機械学習で扱いやすい形の特徴量を生成すること

- 特徴量エンジニアリングの重要性
  - モデルの性能の向上
  - 過学習の防止
  - モデルの解釈性の向上

- データの性質によって有効な特徴量は異なるので、試行錯誤が必要

特徴量エンジニアリングとは - 分類

特徴量エンジニアリングとは、
生データを加工して、機械学習で扱いやすい形の特徴量を生成すること

特徴量変換

異なるスケールの特徴量を統一する
9.2.1: 数値変数のスケーリング, 9.2.2: 数値変数の非線形変換

特徴量生成

モデルがより効果的に学習できるような特徴量を作成する
欠損値の処理, 9.2.3: 数値変数の交差項の作成,
9.3: カテゴリ変数のエンコーディング, 9.4: 時間変数のエンコーディング

特徴量抽出

元のデータの情報を圧縮して少数の重要な特徴量にまとめる
9.5.1: 次元削減

特徴量選択

モデルの性能向上や過学習の防止を目的として、不要な特徴量を除去する
9.5.2: 特徴選択

※分類は文献により異なるのでイメージをつかむ程度で理解してください

本日の講義の構成

本講義では特徴量エンジニアリングとして行われる代表的な処理を紹介します

1. 概要
2. 欠損値の扱い
3. 数値変数の変換
4. カテゴリカル変数の変換
5. 時系列データの扱い
6. 特徴選択

欠損値の扱い - 代表的な手法

基本的には、欠損値のままではモデルの学習に使えないので、処理する必要がある。

- 欠損値の扱いの代表的な手法
  - 代表値で埋める
  - 欠損値とわかるような値(例えば-999など)で埋める
  - 予測する
  - 欠損値を用いて新たな特徴量を生成する

欠損値の扱い

欠損していること自体に意味がある場合に注意して、処理する必要がある

- 欠損していること自体に意味がある場合に注意
  例:
  - { ユーザーID, 商品ID, カート追加日, 購入完了日 } というデータセットに購入完了日が欠損しているデータがいくつかある
  - アンケート結果をみると、ある質問の回答だけ欠損しているデータが多い

欠損値の扱い

欠損していること自体に意味があることに注意して、処理する必要がある

- 欠損していること自体に意味がある場合に注意
  例:
  - { ユーザーID, 商品ID, カート追加日, 購入完了日 } というデータセットに購入完了日が欠損しているデータがいくつかある
    - まだ購入していない場合に欠損しているとわかった
      - 購入完了日が存在する場合は1、欠損の場合は0
      - カート追加日から購入完了日までの経過時間
  - アンケート結果をみると、ある質問の回答だけ欠損しているデータが多い
    - この質問の回答は任意で、関心の高い人以外は未記入であるとわかった
      - 欠損の箇所に 1、そうでない箇所に0
      - 文字数

  ※あくまでも例なので状況に応じてよりよい処理をすることが重要

本日の講義の構成

本講義では特徴量エンジニアリングとして行われる代表的な処理を紹介します

1. 概要
2. 欠損値の扱い
3. 数値変数の変換
4. カテゴリカル変数の変換
5. 時系列データの扱い
6. 特徴選択

変数の型による分類

変数には、「数値変数」と「カテゴリカル変数」がある

	種類	説明	例
量的データ (数値変数)	比例尺度	量的な差と絶対ゼロ点を有し、 比較が可能な尺度	体重（50kg, 75kg）、 距離（5km, 10km）
	間隔尺度	量的な差はあるが 絶対ゼロ点を持たない尺度	温度（摂氏20度, 摂氏30度）、 年（西暦2020年, 西暦2023年 ）
質的データ (カテゴリカル 変数)	順序尺度	順序関係はあるが、 間隔が一定でない尺度	成績評価（A, B, C）、 映画の評価（★, ★★, ★★★）
	名義尺度	単に分類するための尺度で、 数量的な意味は持たない尺度	血液型（A型, O型）、 国籍（日本, 米国）、 商品番号（1, 2, 3）

本日の講義の構成

本講義では特徴量エンジニアリングとして行われる代表的な処理を紹介します

1. 概要
2. 欠損値の扱い
3. 数値変数の変換
4. カテゴリカル変数の変換
5. 時系列データの扱い
6. 特徴選択

数値変換 - スケーリング

スケーリングとは、異なる特徴量間のスケールを均一にすること

- スケールが大きく異なる特徴量を同時に扱うと、
  スケールが大きい特徴量が結果に過度な影響を及ぼす可能性がある
  - 「スケールが大きく異なる」の例
    - 年収と年齢

- 標準化: 平均が0、標準偏差が1になるように変換

$$x_{std} = \frac{x - \mu_x}{\sigma_x}$$

平均

標準偏差

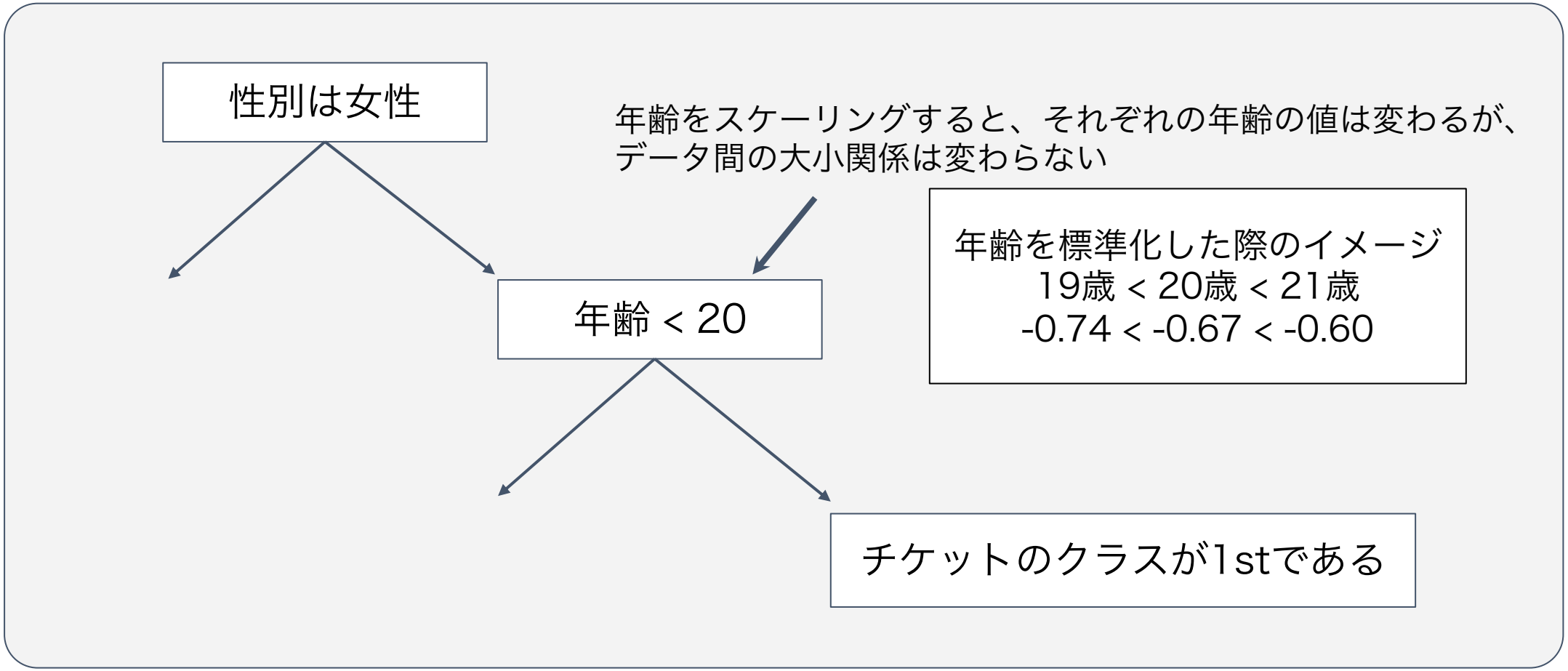
- 正規化: 最小値が0、最大値が1になるように変換

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

数値変換 - スケーリング

決定木系のモデルは、データ間の大小関係を見ているので、
スケーリングが効かない。

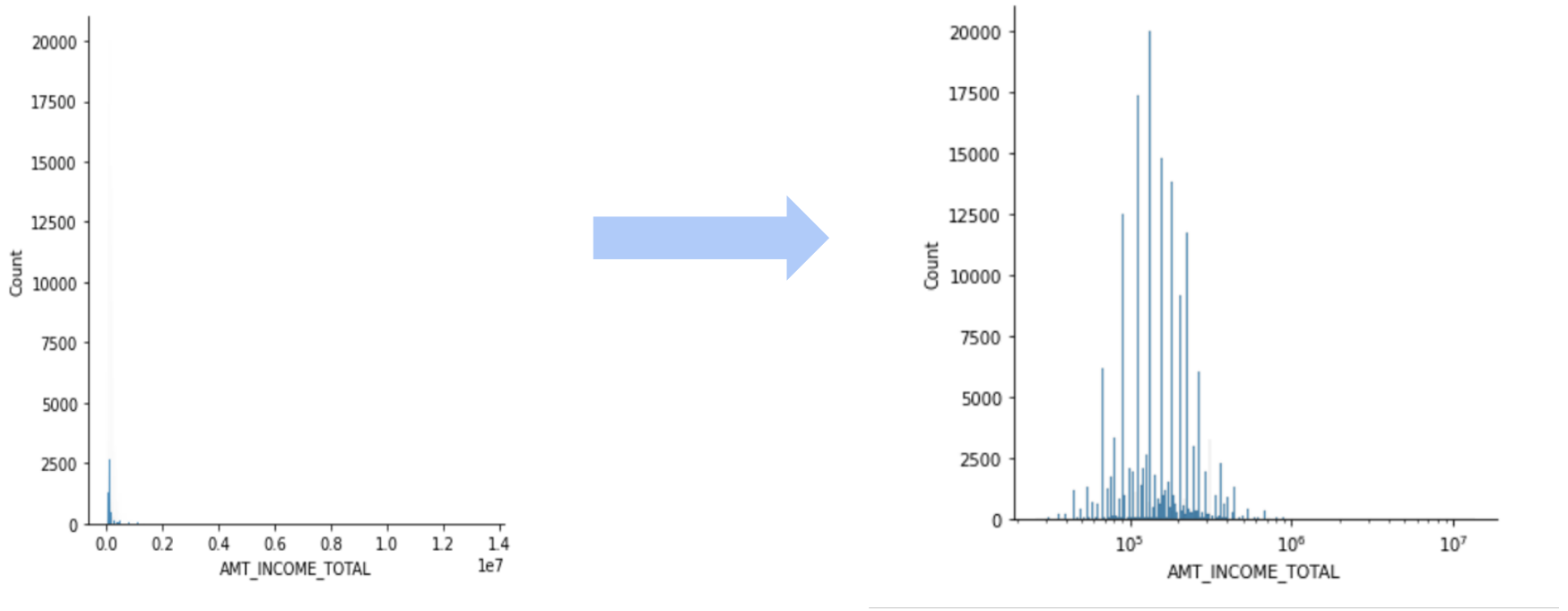
決定木のイメージ



数値変換 - 対数変換

対数変換により、歪んだデータ分布を正規分布に近づけることができる

- 一部のモデルは正規分布を仮定しているため、正規分布に近づけることで精度が向上する可能性がある
- 非常に広い範囲の値を持つデータや右に歪んだロングテールなデータなどに有効



数値変換 - 交差項の作成

交差項: 2つ以上の特徴量の積で生成される新たな特徴量

- イメージ: a, bからa×bという特徴量を作る
- 生成される特徴量の数が多くなる可能性があるため、メモリ使用量や計算時間、モデルの過学習に注意

- sckit-learnのPolynomialFeaturesで簡単に作成できる

```
# ライブラリの読み込み
from sklearn.preprocessing import PolynomialFeatures
# 交差項作成器の作成
pf = PolynomialFeatures(degree=2, include_bias=False, interaction_only=True)
# 交差項の作成
X_pf = pf.fit_transform(X)
```

数値変換 - 交差項の作成

交差項: 2つ以上の特徴量の積で生成される新たな特徴量

pf = PolynomialFeatures(degree=2, include_bias=False, interaction_only=True)

- degree
  - 最大次数
    - degree=2の場合、 [A, B] から新たに [A^2, A*B, B^2]をつくる
- include_bias
  - バイアス項を含むか否か
- interaction_only
  - 相互作用項だけを作成するか否か
    - Trueにすると、1つの特徴量の累乗は生成されない

本日の講義の構成

本講義では特徴量エンジニアリングとして行われる代表的な処理を紹介します

1. 概要
2. 欠損値の扱い
3. 数値変数の変換
4. カテゴリカル変数の変換
5. 時系列データの扱い
6. 特徴選択

カテゴリカル変数の変換 - Label Encoding

カテゴリごとに整数を割り当てる方法

- メリット
  - 数値形式に簡単に変換できる
- デメリット
  - 整数の大小関係がデータにない意味を導入し、モデルが誤解する可能性がある

Embarked
S
C
Q
S
S
C

Embarked
0
1
2
0
0
1

カテゴリカル変数の変換 - Count Encoding

カテゴリごとの出現頻度に基づいて数値を割り当てる方法

- メリット
  - 頻度情報を活用してカテゴリの重要性をモデルに伝えることができる
- デメリット
  - 異なるカテゴリが同じ出現回数を持つ場合、情報の損失が生じる

Embarked
S
C
Q
S
S
C

Embarked
3
2
1
3
3
2

カテゴリカル変数の変換 - Label-Count Encoding

ラベルエンコーディングした後に、出現頻度の順位を割り当てる方法

- メリット
  - ラベルと頻度の情報を組み合わせて、より豊かな表現が可能
- デメリット
  - 異なるカテゴリが同じ出現回数を持つ場合、情報の曖昧さが生じる
    - 登場順に順位を割り当てることで損失は防ぐことが可能

Embarked
S
C
Q
S
S
C

Embarked
1
2
3
1
1
2

カテゴリカル変数の変換 - One-Hot Encoding

カテゴリごとに独立した列を作成し、該当するカテゴリのみに1を割り当てる方法

- メリット
  - カテゴリ間の数値的な関係を排除し、モデルに明確なカテゴリ情報を提供する
- デメリット
  - 多くのカテゴリを持つ変数では次元の爆発を引き起こし、メモリ消費が大きくなる

Embarked
S
C
Q
S
S
C

Embarked_S	Embarked_C	Embarked_Q
1	0	0
0	1	0
0	0	1
1	0	0
1	0	0
0	1	0

カテゴリカル変数の変換

Embarked
S
C
Q
S
S
C

Embarked_S	Embarked_C	Embarked_Q
1	0	0
0	1	0
0	0	1
1	0	0
1	0	0
0	1	0

One-Hot Encoding

Embarked
0
1
2
0
0
1

Label Encoding

Embarked
3
2
1
3
3
2

Count Encoding

Embarked
1
2
3
1
1
2

Label-Count Encoding

本日の講義の構成

本講義では特徴量エンジニアリングとして行われる代表的な処理を紹介します

1. 概要
2. 欠損値の扱い
3. 数値変数の変換
4. カテゴリカル変数の変換
5. 時系列データの扱い
6. 特徴選択

時系列データの扱い

時系列データ: 時間に伴って変化するデータ

- 例
  - 株価データ(時間に伴って数値が変化)
  - 定期的に撮影した胃カメラのデータ(時間に伴って画像が変化)
  - 工場現場の音声データ(時間に伴って音声が変化)
  - X(旧Twitter)のポスト(時間に伴って言語が変化)

- 今回は**単変量時系列データ**を主に扱う

時系列データの扱い

時系列データ: 時間に伴って変化するデータ

- 今回は単変量時系列データ(1つの数値が時間に伴って変化する)を主に扱う

テーブルデータ

	TOEICの点数	1日の勉強時間	年齢	TOEICの学習歴
Aさん	660	3	40	5
Bさん	345	1	18	0.5
Cさん	700	2	35	2
Dさん	?	3	25	1

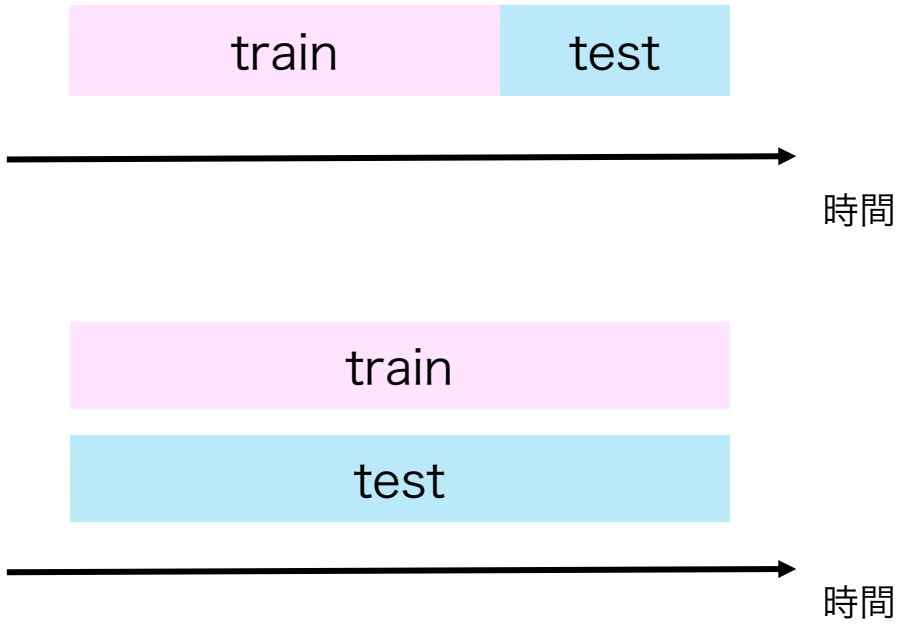
単変量時系列データ

	AさんTOEICの点数
2021	455
2022	580
2023	660
2024	???

時系列データの扱い

時系列データを扱う際は、リークしないように注意が必要

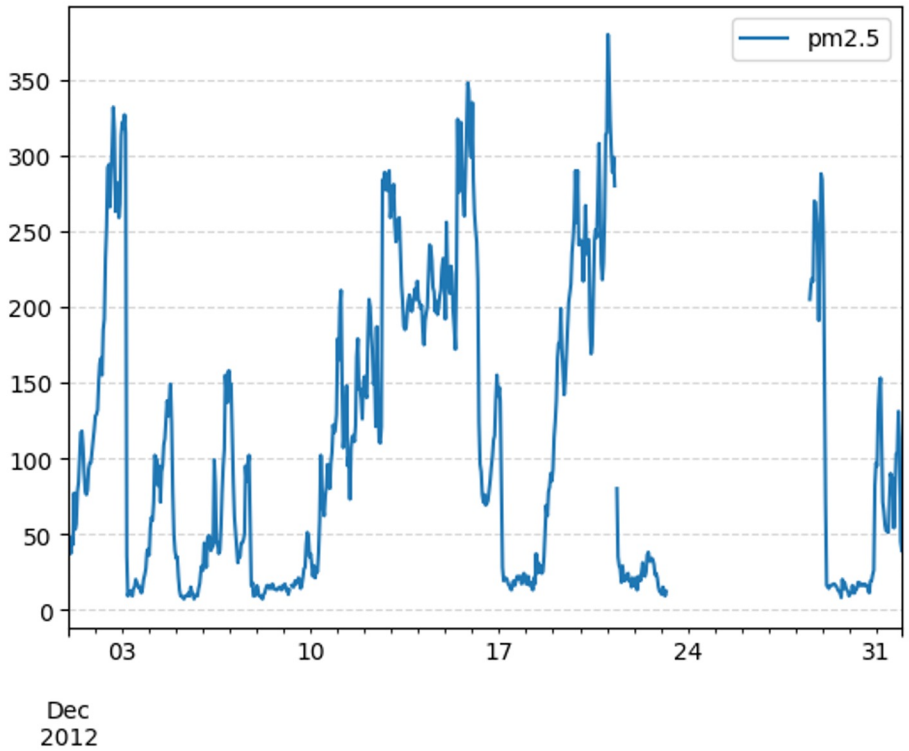
- 基本的に、時系列データを用いる場合、未来予測の場合が多い
- 学習データに未来のデータが入らないように気をつける



時系列データの扱い - 可視化

時系列データも、他のデータと同様に可視化することでデータ全体が把握できる

- 可視化することのメリット
  - 季節変動やトレンドを明確に把握できる
  - 異常値や外れ値の発見が容易になる（→必要な前処理がわかる）



時系列データの扱い

時系列データの特徴量生成の例

- 比率をとる（前日との比をとる例）

2012-12-01	119.0	NaN	NaN
2012-12-02	313.0	119.0	2.630252
2012-12-03	40.0	313.0	0.127796
2012-12-04	39.0	40.0	0.975000
2012-12-05	29.0	39.0	0.743590

← 313.0 ÷ 119.0

← 40.0 ÷ 313.0

- 移動平均（3時間の移動平均を計算する例）

2012-12-01 00:00:00	41.0	NaN
2012-12-01 01:00:00	46.0	NaN
2012-12-01 02:00:00	37.0	41.333333
2012-12-01 03:00:00	48.0	43.666667
2012-12-01 04:00:00	43.0	42.666667

← (41.0 + 46.0 + 37.0) ÷ 3

← (46.0 + 37.0 + 48.0) ÷ 3

時系列データの扱い - エンコーディング

時系列データの処理は「周期性を考慮しないエンコーディング」と「周期性を考慮するエンコーディング」にわけられる。

- ［周期性の考慮なし］年月日をそれぞれ数値として分割する

```
df_dates['year'] = df_dates['ymd'].apply(lambda x: x.year)
df_dates['month'] = df_dates['ymd'].apply(lambda x: x.month)
df_dates['day'] = df_dates['ymd'].apply(lambda x: x.day)
```

ymd	year	month	day
2019-06-21	2019	6	21
2014-04-27	2014	4	27
2018-07-02	2018	7	2

時系列データの扱い - エンコーディング

時系列データの処理は「周期性を考慮しないエンコーディング」と「周期性を考慮するエンコーディング」にわけられる。

- ［周期性の考慮なし］ 基準日からの差分を変数としてとる

```
start_date = pd.Timestamp('2010-01-01 00:00:00')
df_dates['total_days'] = df_dates['ymd'].apply(lambda x: (x - start_date).days)
```

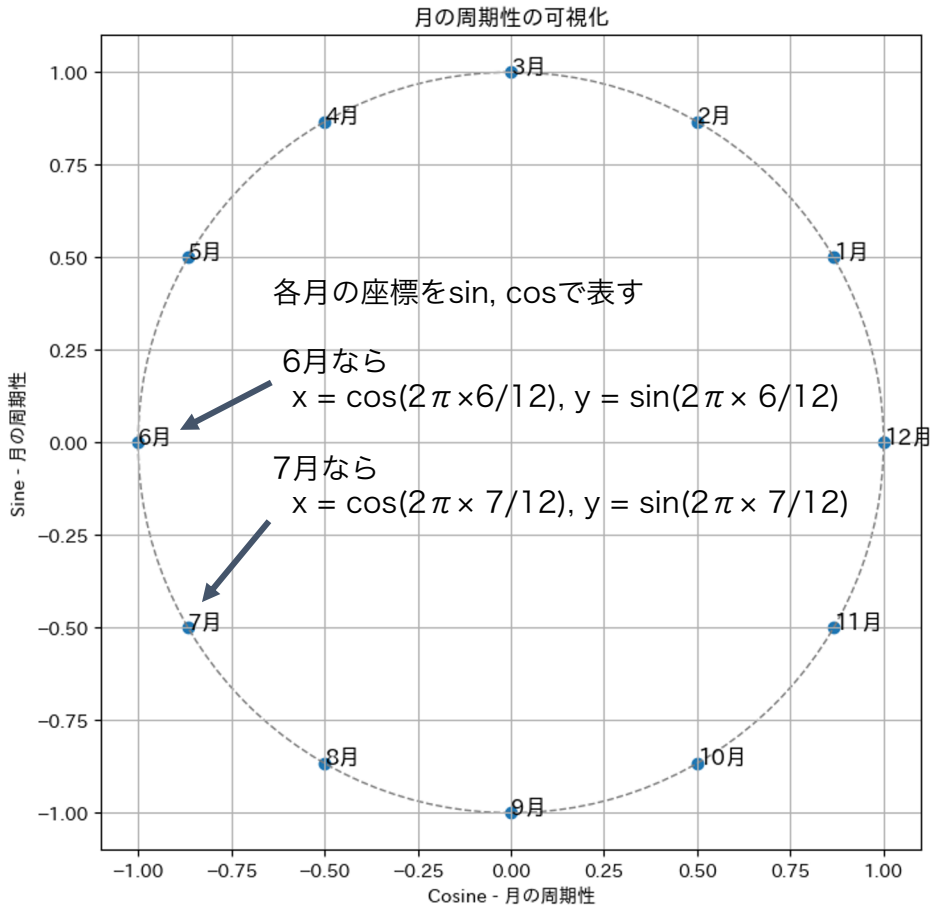
ymd	year	month	day	total_days
2019-06-21	2019	6	21	3458
2014-04-27	2014	4	27	1577
2018-07-02	2018	7	2	3104

時系列データの扱い - エンコーディング

時系列データの処理は「周期性を考慮しないエンコーディング」と「周期性を考慮するエンコーディング」にわけられる。

- ［周期性の考慮あり］ 月や日の周期性を三角関数で表現する

- 1月と12月のような時間的に近いが数値的には離れた期間も適切に近いと認識できる
- 元の1次元の時間変数を2次元に拡張することで、モデルが時間データのパターンをより複雑に捉えることができる



時系列データの扱い - エンコーディング

時系列データの処理は「周期性を考慮しないエンコーディング」と「周期性を考慮するエンコーディング」にわけられる。

1. 基準日からの経過日数(total_days)
    - 時間の経過を表すことができた
2. 三角関数の利用(month_cos, month_sin, day_cos, day_sin)
    - 時間として連続な関係を表すことができた
        - 12月→1月, 31日→1日

	ymd	year	month	day	total_days	month_cos	month_sin	day_cos	day_sin
0	2019-06-21	2019	6	21	3458	-1.000000e+00	1.224647e-16	-0.440394	-0.897805
1	2014-04-27	2014	4	27	1577	-5.000000e-01	8.660254e-01	0.688967	-0.724793
2	2018-07-02	2018	7	2	3104	-8.660254e-01	-5.000000e-01	0.918958	0.394356

本日の講義の構成

本講義では特徴量エンジニアリングとして行われる代表的な処理を紹介します

1. 概要
2. 欠損値の扱い
3. 数値変数の変換
4. カテゴリカル変数の変換
5. 時系列データの扱い
6. 特徴選択

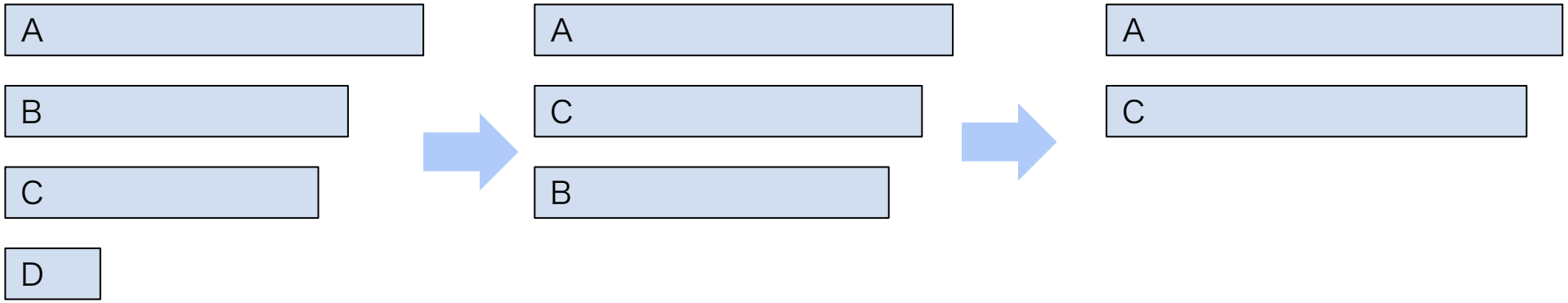
特徴選択 - RFE(再帰的特徴消去)

RFE(Recursive Feature Elimination)は、最も特徴量重要度が低いものを消去することを再帰的に繰り返し、指定した特徴量の数になるまで、消去を行う手法。

```
# 特徴選択器の作成
rfr = RandomForestRegressor(random_state=42)
rfe = RFE(estimator=rfr, n_features_to_select=10)

# 交差項を含むケースで特徴選択の実行と選択された特徴量のデータフレームを作成
rfe.fit(X_pf_df, y)
X_rfe = X_pf_df.loc[:, rfe.support_]
```

- 単に、特徴量重要度の高い上位n個を選択した場合とは異なる結果になりうる



本日の講義の構成

本講義では特徴量エンジニアリングとして行われる代表的な処理を紹介します

1. 概要
2. 欠損値の扱い
3. 数値変数の変換
4. カテゴリカル変数の変換
5. 時系列データの扱い
6. 特徴選択

講義資料にはない特徴量エンジニアリングの手法

- 次元削減
  - 次元削減の結果を特徴量として追加する
- クラスタリング
  - クラスター番号を特徴量として追加する
- 特徴量の組み合わせ
  - 交差項のように、特徴量同士を組み合わせたものを追加する
    - 例: 人口/面積で人口密度を追加する
- ドメイン知識を活かす
  - データのドメイン固有の知識を活かした特徴量を追加する
    - 例: LTV＝平均顧客単価×収益率×購買頻度×継続期間

参考資料

- Kaggleで勝つデータ分析の技術
  - 門脇 大輔, 阪田 隆司, 保坂 桂佑, 平松 雄司