

# AIエンジニアリング講義 第4回

---



2025/05/07

許諾なく撮影や第三者  
への開示を禁止します

- 1 AI開発は”汚いデータ”との戦い
- 2 学習/検証/評価データの設計が「精度」を決める
- 3 特徴量エンジニアリングによってモデルにコンテキストを与える
- 4 データが少ない時の戦い方
- 5 現場あるあるから学ぶ”使われるAI”とは



## 清水 茂樹

松尾研究所 共同研究チーム  
マネージャー



Competitions Expert  
1,252 of 203,354

Competitions  
Expert

MEDALS



ハードウェアエンジニア

AIリサーチャー

コンサル  
データサイエンティスト

大手自動車会社 関連会社



- 機械設計
- 自動操縦ロボット



- 感情推定モデル
- 生体データ分析

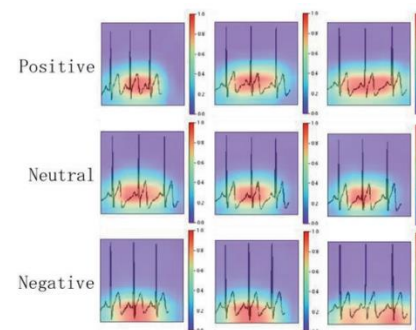


Fig.11 Feature map of Participant 1 by Grad-CAM

大手Sler



- インフラ点検診断自動化支援
- トンネル、道路
- 類似工事現場探索支援
- 小売実需予測支援
- 潜在的事故リスク分析支援



## キーメッセージ

- ・ AIは現場でみんなで育てていく
- ・ 綺麗なデータも正確なラベルも評価指標も最初から揃っていることはありません
- ・ 今あるもので仮説を立てて、工夫して補い、試行錯誤を繰り返すことで現場に定着させる



- ・ AIを育てるとは、データの取得/理解を自ら行いモデルの精度を上げていくことです。
- ・ 泥臭く面倒なことにこそ価値があります。

# 1. AI開発は”汚いデータ”との戦い

---

よく言われることではありますが、「AI開発」の実態はデータとの格闘が大半を占めています。  
データを素早く「理解」し、「コントロール」できる術を身につけることでAI開発の効率が格段に上がります。

## 前処理

## モデル作成

## エラー分析

## 報告

- 構造化/画像/自然言語それぞれで使われるモデルは大体同じようなモデルが使われる
  - 構造化: LightGBM
  - 画像: EfficientNet, SwinTransformer
  - 言語: DeBERTa-v3, LLama, Qwen
- 差がつくのは前処理とエラー分析のデータエンジニアリング

## 前処理

- 欠損, 異常値の処理: 単純な補完ではなく、意味解釈が重要
- 表記揺れへの対応
- データの結合、最小単位でデータを持つ

## エラー分析

- マクロな精度とミクロなケース
  - マクロな精度: モデルの全体性能。評価データ全体での精度評価
  - ミクロなケース: どのケースが使えないか。正解/不正解のケースを列挙。不正解が当てられるようになったか

# データ整備とは？



データ整備とは、データにおける「構造」「粒度」「意味」を揃える作業です

	説明	揃っていない例	整備例
構造	データの型/形式が統一されているか	日付が文字列で入っている primary keyにnullや重複あり	型変換 ファイルの結合/分離
意味	各列名が何を表しているかが定義され解釈できる	各列ごとのカテゴリやラベルが揃っていない 部署名: 営業部、第1営業部など	スキーマの作成 コンテキスト/定義の明文化
粒度	データの集約レベルや範囲が統一されているか	日単位と週単位が混在 データ1行のサンプリング周期が不明	粒度を合わせる(週次で集約など) primary keyの設計

データを集計、分析する際に自由入力列では情報を正規化する必要や粒度の細かいデータのみを使用するケースもあります

## 一つの列の中で書き方がバラバラ

日付	部署名	商品名
2025/02/03	第1営業部	冷却機
20250203	第 1 営業部	空調冷却機
2025-02-03	営業1	空冷機

どれも同じ部署、商品のケースもある

- ・ TF-IDFなどにより正規化
- ・ 辞書を作成  
→最近ではLLMに任せても良さそう

## 長文自由入力による情報粒度がバラバラ

日付	社員コード	営業日報
2025/02/03	xxxxxx	10:00～10:30 株式会社AA：見積り説明（提案書Ver3を使用）。担当: 田中様。 課題: 「操作の煩雑さ」とのこと。次回: 5/8再訪。 11:00～12:00 株式会社BB：機能説明（DemoB使用）、競合にSmartXが入ってるとのこと。
2025/02/03	yyyyyy	・ A社訪問。説明して終了。 ・ B社にも行った。

綺麗なデータ(情報粒度が細かいデータ)のみから始めるケースも多い



# “汚いデータ” | ケース② 意味を持たない列



データとして存在していても、実質的に意味を持たない列は多く存在します。  
欠損・0埋め・一意な値しかない列は、モデルにも分析にも使えない無駄な列となりうるため確認を必ずしましょう。

## 欠損だらけ、0だらけのデータ

日付	訪問先	社用車利用フラグ
2025/04/01	A社	0
2025/04/01	B社	0
2025/04/02	C社	0

フラグがあるだけ  
特定の訪問先だけで利用し、実質意味を持たない

## カテゴリのユニーク数が1つしかない

顧客ID	購入額	キャンペーン対象
C001	4,980円	○
C002	3,280円	○
C003	5,000円	○

キャンペーン対象しかデータ化されておらず  
集計、学習に意味を持たない

データをどのように入力しているか、どの時点のものをデータ化しているか  
など現場やドメイン知識によって残すべきかどうかを判断

# “汚いデータ” | ケース③ 突然変わる仕様



センサー仕様や新製品がデータ上に現れモデル性能が著しく下がるケースもあります。  
人には同じように見えてもモデルには全く異なるデータであるため、

## 搭載センサーの仕様変更



来月からセンサーを新製品に置き換えます。新しいセンサーにも対応できるモデルにしてください

センサースペックが異なり、再学習の必要あり  
場合によっては前処理から見直す

## これまでと違う製品の発売

商品名	単価	売上部数
月刊〇〇 4月号	780円	12,300部
月刊〇〇 5月号	660円	8,200部
ムック：□□大全集	1,280円	3,500部
豪華愛蔵版BOX（ 全巻収納・グッズ 付）	12,000円	10部

パッケージ商品などこれまでの販売チャンネルや  
プロセスと異なる商品が混在する  
特殊商品である旨を別カラムで管理

# “汚いデータ” | ケース④ 人によって変わるラベル



人がラベル付けしているものは必ずラベル揺れが存在します。  
判断基準の明確化や検出対象を明文化して、揺れを可能な限り少なくする必要があります。

## 担当者によって判断が割れる①

訪問日	コメント	ラベル
2025/02/03	説明が丁寧でした	ポジティブ
2025/02/04	機種変更は思ったよりスムーズにできましたが、SIMカードの説明が少し分かりづらかったです。	ネガティブ

人によっては両方ともポジティブと捉える  
例えば複数人でラベルをつけて過半数を  
正解ラベルとする など人に依存しない工夫が必要

## 担当者によって判断が割れる②



どこまで細かいヒビを検出するか、アノテーション時点で揺れる

データの理解には「分布の確認」と「類型化」の往復によって、データの特徴・違和感・意味が立ち上がってきます

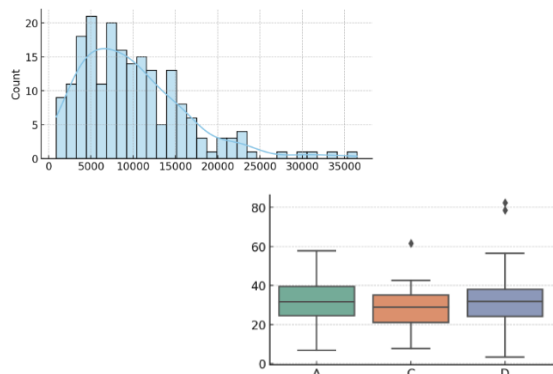
## 分布の確認

### ポイント

- 数値の分布
- カテゴリの出現頻度
- 欠損値やハズレ値の確認

### 方法

- histplot
- boxplot
- heatmap



## 類型化

### ポイント

- パターンを見つけて分類（グループ分け）
- 行動パターン/売れ筋/よくある組み合わせを探る
- 定性的に似ているものを“言語化”する
- クラスタリングの前に分布の可視化から探る

### 例

- 発売直後にピーキーに売り上げが立つ商品やじわじわとずっと売れ続ける商品
- 特定の季節にだけ売れる商品 など

データのことを少しずつ理解を深めていくためにドキュメント化が必要です。  
データ分析はチームで行うことが多いので、どう使うべきかを言語化し、備考欄に情報を貯めていきましょう。

列名	説明	型	値域	ソース	備考/メモ
id	商品識別ID	str	aaaa-zzzz	商品マスター.csv	旧IDはid_old.csvに書かれている
created_at	タイムスタンプ	datetime	yyyy/mm/dd- yyyy/mm/dd	-	売上から2日後にシステムに登録される
sales_cnt	売り上げ数	int	-10 - 10,000	yyyymm_sales.csv	負の値は返品

良いモデルを作るためにデータ整備は理解とドキュメント化の往復が大切です。  
ステークホルダー全体で「徐々に良くなる」という姿勢が将来的に助けてくれます。



- データ入手: 「これが最新版です」「全部入ってます」ともらえるデータは疑ってかかったほうが良い
- 初期チェック: データをもらった際の説明と合っているかを自分の目で確認
- データ理解: 分布と類型化が重要
- ドキュメント化: 今の自分ではなく、半年後の他人が読んでわかるようにまとめる

- 最初から「整備のフロー」を何周も回すという気持ちで取り組むことが大切
- いきなり100点のデータを目指すのではなく、60点くらいから徐々に100点に近づけるイメージ
- 特にデータを準備してくれた部署や人に対して、都度説明しながらフローを回すことでステークホルダー全体で理解を深めていく

## キーメッセージ

- ・綺麗なデータは基本存在しません
- ・構造/粒度/意味の3軸でデータについてのドキュメントを作しましょう
- ・Bad Dataの例
  - ・自由記述の揺れ(構造、粒度)
  - ・意味を持たない列
  - ・主観的なラベル
- ・分布と類型化でデータについて深く理解することが重要

## 2. 学習/検証/評価データの設計が「精度」を決める

---



# なぜ評価設計が必要なのか？



これから開発するAIのゴールを見定める。AIの予測結果を基に「意思決定」や「アクション」が変わることが大切。

## 評価指標の考え方

### 技術的 指標

AUC, F1 Score, RMSEなど  
仮説検証が合っているかを確認するための指標

### ビジネス的 指標

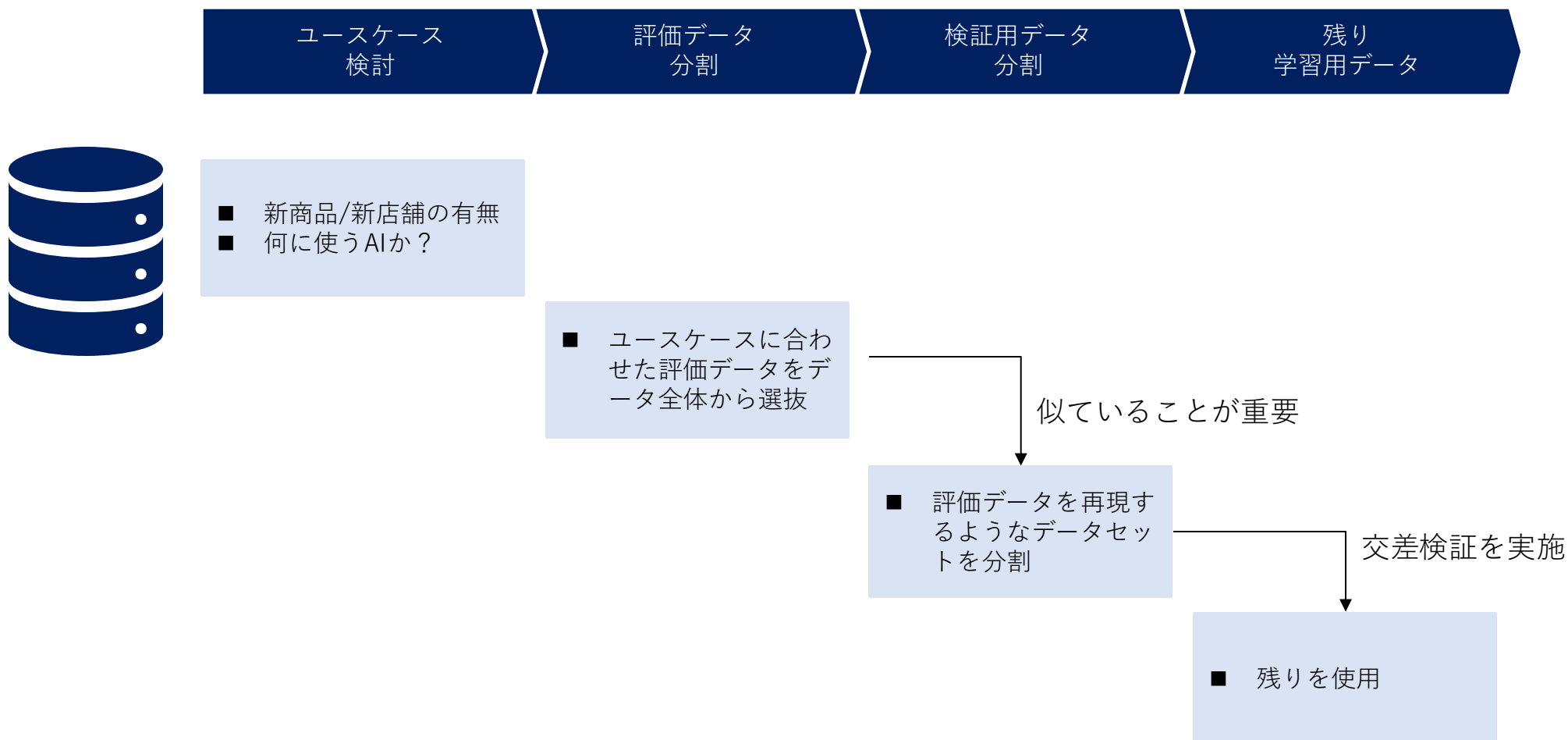
業務削減効果: 抜け漏れがない状態で確認するべきデータ量がどれだけ減るか  
売上予測: ロット生産で生産量が減らせるか、輸送するトラックを減らせるかまで落とし込む

分析結果や予測結果を受けて、ユーザーの行動が変わることが望ましい

例えば、天気予報の例では降水確率を54.1%など小数点第一位まで出すことに意味はなく、傘マークなどで指標化することで今日傘を持っていくべきかを判断し行動します

この指標で予測することで、ユーザーは嬉しいかどうか  
が評価指標を考える上で非常に重要です。

まずは評価データを先に考えた上で、評価データに合わせた検証用データと学習用データに分割する設計を進めます  
評価データを考える上では「未知」「当てたいケース」をユーザー部署と考える必要があります



# 失敗ケース①: 精度は良いが現場で使えない




特に異常検知系でよく起こりがちですが評価指標とユースケースが合っていないことにより、導入の見送りが散見されます

## モデル開発時

- 仮説検証を回すためにAUC/PR-AUC/F1-Scoreで評価
- F1-Scoreの閾値も最適化によって探索

## 現場ニーズ

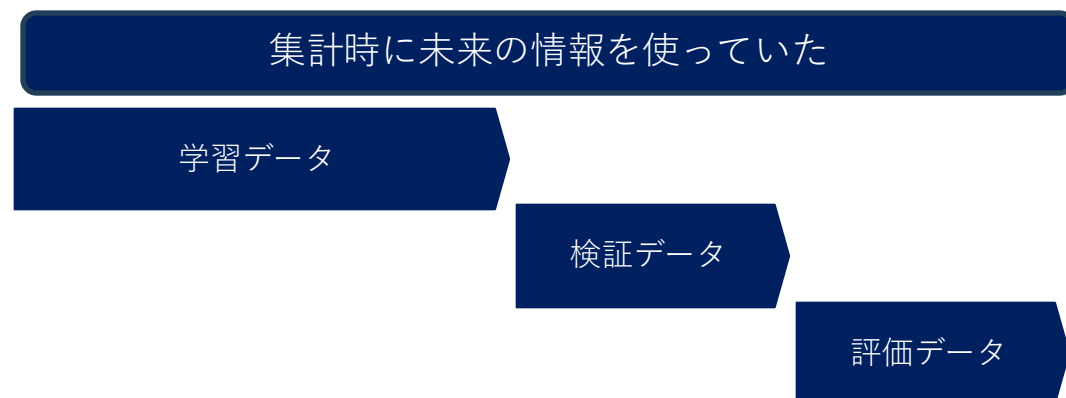
- 見逃しがあると結局人が全て確認しないといけない
- 確認すべきデータを減らして欲しい

- 
- 技術的指標の他にビジネス効果を算出
    1. 検証用(validation)データで見逃しがない閾値を設定
    2. 予測確率が閾値以上のデータはAIによる予測を実施、閾値未満のデータは人が確認
    3. 閾値未満のデータが評価用(test)データでどれくらい削減できたか を算出

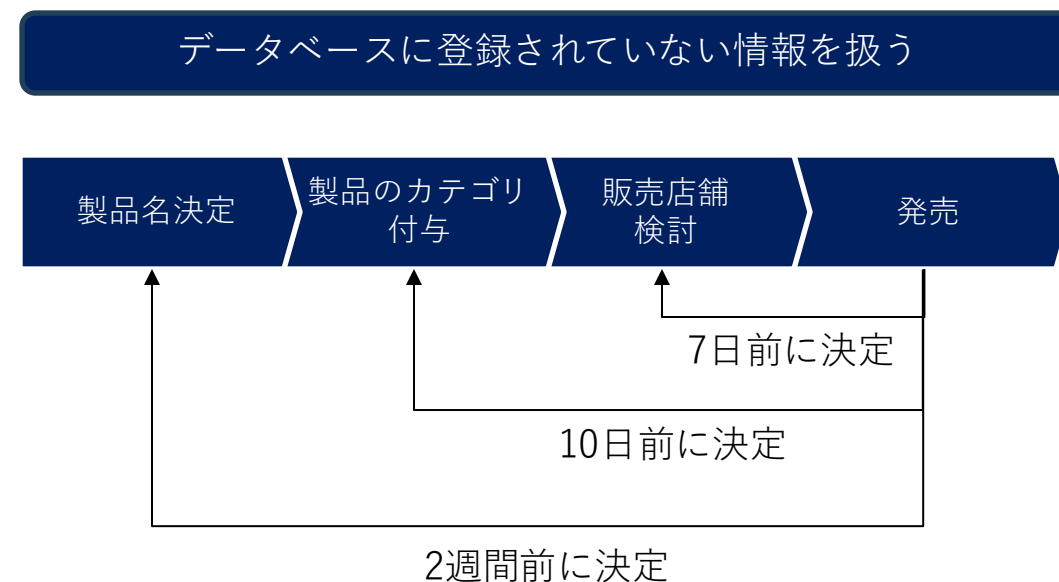
## 失敗ケース②: リークageが起きている



データ分割はリークageが起きていなくても、特徴量の集計時に未来の情報を使ってしまうたり、集計をする時点では本来まだデータベースに登録されていない情報を扱ってしまうケースがあります



データ分割は分けていても、特徴量を作る際にデータ全体でgroupbyをしている例などよく見かけます。



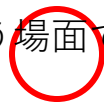



すでにデータには登録されている情報として特徴量を使用しますが、実際に予測する時点では使えない情報が混在しています

予測時点でどの情報までが使えるのかを確認しておきましょう

結果が良すぎる時は喜ぶ前に疑いましょう！

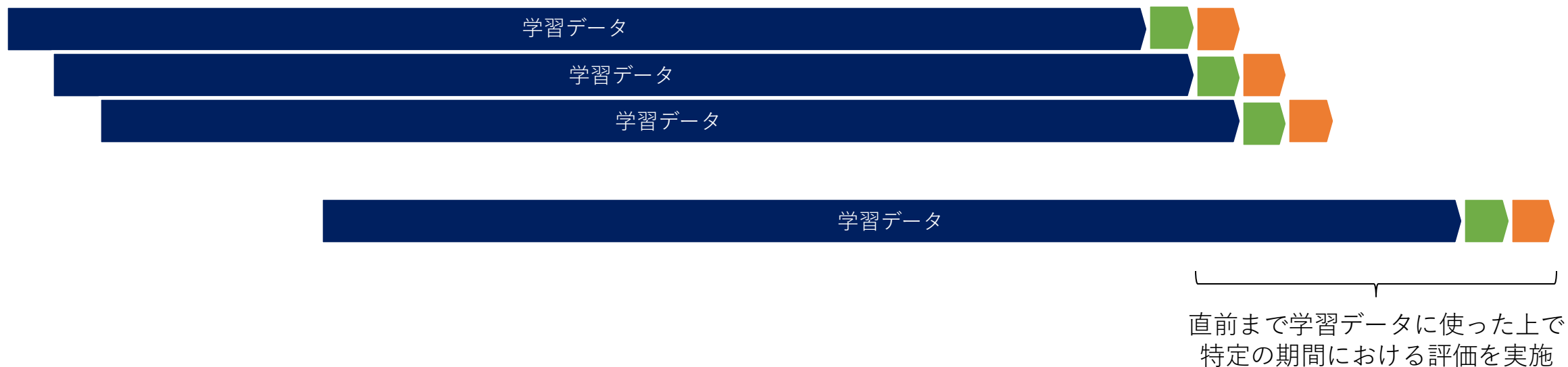
データの分割は「何を未知とするか」で決めます。ユースケースに応じた汎化対象を明確にした上で適した分割方法を使いましょう

	使う場面	実用性
StratifiedKFold	不均衡データでラベルの比率を保ちたい	異常検知ではよく使うが、ラベルだけのStratifiedKFoldでは不十分 
GroupKFold	新商品や店舗などを未知としたい	ラベルの偏りや分布が崩れるため不安定になりがち 
StratifiedGroupKFold	新商品や店舗などを未知としつつ、分類タスクでラベル比を保ちたい	GroupKFoldを使う場面ではこちらを使用することが多い 
TimeSeriesSplit	未来予測がしたい	使われがちだが日時バッチで予測を回す際には不適切 

## よく見かける例



## 運用に近づけた例(日次や週次でモデル更新を実施する想定)



モデルの精度を高めるだけでなく、「何を測るべきか」「どこで評価すべきか」を定めること自体がスキルです。  
AIを使える形に落とし込むための設計力を身につけましょう

具体化	汎化対象(商品/店舗/ケース)を言語化しデータに落とし込むことができる
説明力	技術的指標とビジネス的効果のそれぞれを設計し理解を得られる
想像力	アラートが鳴りすぎないなど、ユーザーや意思決定者の使われ方を想像できる

一度決定したら動かさないわけではないので、何度も試行錯誤と現場への説明を行い決定していきましょう

## キーマッセージ

- ・ 技術的指標とビジネス的効果の両方を出すことが大切
- ・ ビジネス効果は「使う人にとって意味がある指標か」を考える
- ・ 未知のデータに合わせて、データ分割を実施する
- ・ 評価指標を正しく設計できるスキルも重要



### 3. 特徴量エンジニアリングによってモデルにコンテキストを与える

---

# なぜ特徴量を作るのか？



特徴量とはモデルに「何を見るべきか」を教えるためのものです。  
人が意味的に理解しているものをモデルにもわかるように翻訳する気持ちで作しましょう

## 特徴量とは？

生データのまま渡してもモデルには通じないため、コンテキストを教える

構造

周期性や祝日、休日など  
経過時間や駅からの距離など算出も必要

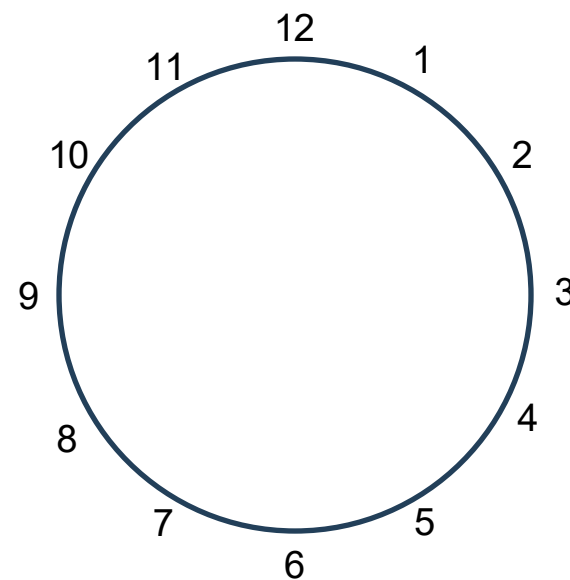
意味

レアなものや新しいものであること  
ハズレ値や欠損値も教える

違い

差分、変化、先週比率

## 特徴量の例



月や時間などの特徴量がある場合にそのまま使うのではなく、単位円の座標に変換する

人は12月の次に1月が来ることを知っているがモデルは知らない。12月と1月が近いことを教えてあげる必要がある

構造化データではよく使うものとして、カテゴリカル変数を用いたGroupbyによる集計を使うことが多いです。  
全体やカテゴリの中で相対的な位置を教えることが重要です。

コンテキスト

商品分類をカテゴリカルな変数にしたところで意味が通じない  
Aグループは平均購入単価が高い、返品率が低いなどの性質を伝える

特徴量の例

傾向

ユーザーの情報は少ないとしても年齢や地域ごとの  
購入回数など類似ユーザーの傾向を教える

pandasのgroupbyで  
max, sum, min, count, mean, std  
をよく使います

Excelでピボット集計をすることと同じです

相対位置

カテゴリの中での平均売り上げ金額に対しての売り上げ金額など  
属しているカテゴリ内での相対的に高い or 低いを教える

カテゴリ変数がない時には、数値系  
変数からカテゴリを作ったりします  
例) 年齢など

画像でt-亜における特徴量設計は、適切なAugmentationを通じて汎化したい対象を教え込むことに近いです。  
特にテストケース(現場での使われ方)に合わせて、組み合わせた設計をします。

幾何学  
変換

HorizontalFlip, Rotate,  
ShiftScaleRotate, RandomCrop

位置依存性の低減

色変換

HueSaturationValue,  
BrightnessContrast, RGBShift

明るさや照明条件のばらつき低減

情報の  
欠落  
付与

Cutout, CoarseDropout, Blur,  
RandomShadow

一部欠損・ノイズへの耐性

混合

Mixup, CutMix

組み合わせた画像によるロバスト性

実用例

albumentationsという  
ライブラリを使うことが  
多いです

不必要なAugmentationは  
毒にもなり得ます

テキストはそのままではモデルに渡せないため、数値化(ベクトル化)する必要があります。  
シンプルなTF-IDFや文脈を捉えるBERTもありますが、直近はRAGでよく使われているので馴染みも多いかと推察します。

従来  
手法

TF-IDF, BM25

辞書  
ベース

キーワードの出現回数、単語数

埋め込み

word2vec, fasttext  
Sentence-BERT, LLM

カウント

段落の数、文章の数、句読点の数 など

特徴量の使われ方

他の構造化データにマージして  
LightGBMなどのモデルを使うこと  
やLLMなどのDeep系のモデルに入れ  
ることが多いです

時系列データでは、「過去の値そのもの」ではなく、時間的な変化や傾向を特徴量として取り出すことが重要です。  
変化量・周期性など、時間を圧縮・抽象化した指標を工夫して使います。

ラグ

1日前、3日前、1週間前など過去のデータを活用

変化量

ラグ特徴量からの差分を取り、上昇下降トレンドを定量化

移動平均

ノイズを滑らかにし、傾向を抽出  
pandasのrolling(window=3)など

周期性

季節性(週、月、季節、年、時間帯)の依存関係を抽出

累積系

1週間分の累積量など、長期的な傾向  
pandasのcumsumなど

特徴量の使われ方

皆さんも「AIを使わずに商品の売り上げを予測して」と言われたら、前日何個売れたんだろう？とかこの時期によく売れるから多めに仕入れておくかなど過去の情報を参考にしたいと思います

その情報をモデルに教え込むことが大切です

# 特徴量設計におけるセンスとは？



特徴量設計のセンスとは「対象となるケースをどうモデルに伝えるか？」を考える力です。  
モデルの気持ちを考え、モデルに寄り添いましょう。

変化

過去や他の値に比べてどれくらい変化しているか？

相対性

カテゴリ内での相対位置や比率はどれくらいか？

ドメイン知識

業務的にこの値が大きいつてことはどういうこと？

モデルの気持ち

モデルは何を学ぼうとしているか？

センスの養い方

特徴量重要度やエラー分析、特徴量の  
ON/OFFで精度を見ながら仮説検証サイクル  
を多く回しましょう

絶対効くと思った特徴量が効いていないとき  
はモデルへの渡し方が雑なのかもしれません

## キーメッセージ

- ・ 特徴量エンジニアリングとはモデルに何を見るべきかを伝える翻訳をすること
- ・ 構造化データ： Groupby集計によるカテゴリ内での相対位置を教える
- ・ 画像データ: albumentationsを使い幾何変換/色変換/欠落/付与など
- ・ 言語データ: TF-IDF, BERTなどのベクトル化
- ・ 時系列データ: ラグ/差分/累積/周期性を特徴量化



## 4. データが少ない時の戦い方

---

# データは基本的に揃っていない

データが全部揃ってから始めることはほとんどありません。  
データが足りない、ラベルがついていないという前提で前に進めるかが問われます。

## 揃っていない現実

### データが 不足

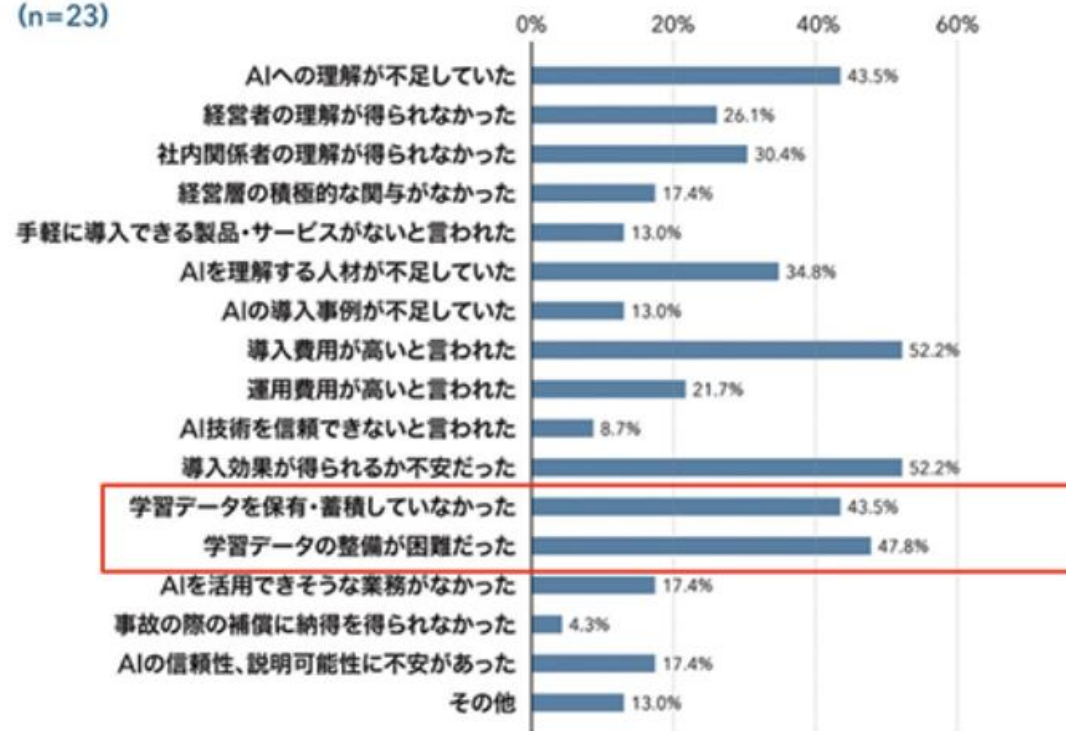
- データ化されていない
- データ化はされているものの整備されていないため使える状態ではない
- 目的には合わないデータはある

### ラベルが 不足

- データはあるが、アノテーションはされていない
- アノテーションが曖昧で信頼性が低い

図表3-4-13 開発が中止となった経験がある場合のその理由(複数回答)

(n=23)



注：図表3-4-12において「ある」と回答した23社に尋ねた。

出典: AI白書2023

データが少ないときでも、できることはたくさんあります。  
小規模だからこそ、仮説を立てて検証し、構造を理解する力が問われます

## 試すこと

問いを  
単純化する

- 特徴量の傾向を見ることが仮説検証に重きをおく
- 分類タスクに落とし込む

類似例を  
参考にする

- 期待できる目安(チャンピオン精度)を知る
- 論文やオープンデータを活用

## 期待する効果

- データが溜まればこういうこともできそうと想像を膨らます
- 少量で難しいケースを抽出

- データが集まればここまでできるんだという期待を持つことが重要
- ステークホルダー全体でデータを集める

少量データでは特に「学習データに合いすぎて、実用では効かない」状態=Overfitが起きやすくなります。  
モデルだけでなく、データ設計・評価設計の工夫によってもOverfitは防げます。

## モデル

- 小さなモデルで試す
  - SVM, 線形モデル
  - Deepでもsmall系のモデル
- EarlyStopping

## データ

- Foldを慎重に切る
- 特徴量を減らす

## 評価

- スコアだけでなく、Fold間のばらつきを見る
- fold構成を複数パターン試す

評価結果にこだわらず、仮説検証の正しさにこだわる  
データが増えたときの道標を作るイメージ

少量データに対して“OverSamplingすればいい”と思い込むのは危険です。  
特に少量データでは、ノイズの複製／分布の崩壊／過学習など新たな問題を引き起こします。

## OverSamplingで起きがちな失敗パターン

### SMOTEの 乱用

- 実データではあり得ない例が学習され  
精度低下やリークageを誘発する

### 異常データの 過剰補完

- 少数クラスを複製、Pseudo Labeling  
による擬似データの追加を行い実際の  
比率と乖離した精度検証になる

- 少量データでは簡単にOverfitするという前提に立  
った上でモデル、データ、検証に気を使いましょう
- 精度を上げることに囚われてはいけません

地道なアノテーションと、その効率化こそがモデル精度を決める鍵です。  
モデル開発だけでなく、“データをどう賢く増やすか”の戦略を持てる人が、最終的に強いAIを作れます。

## データ収集

- 自ら実験台になることや身の回りで協力者を探し簡易的な実験でデータを増やす
- LLMでデータも作りやすくなっているが、バイアスや類似性については要注意

## 手動アノテーション

- Label StudioやStreamlitを活用し自分でラベルをつける
- データに対するカンコツや難しい要素も理解できる

## データの ドキュメント化

- アノテーション基準の明文化、判断基準を統一
- データが溜まった時に素早くアノテーターに渡せる準備を整える

泥臭いことを取り組んでこそ  
モデルの気持ちやデータの  
ことが理解できます。

いつデータが増えてもいいよ  
うな準備を進めつつ、地道に  
自分でも増やしていく活動が  
大切です。

## キーメッセージ

- ・少量でも仮説検証を回す
- ・少量の場合はOverfitが容易に起こりうるため、評価のばらつきに注目
- ・安易なOversamplingは危険であり、地道に増やすことが大切

## 5. 現場あるあるから学ぶ”使われるAI”とは

---



自分が被験者になることで、「どう貼ればちゃんと測れるか」「何がノイズか」がわかります  
自分でやったからこそ、「このセンサーの値はこう変わるべき」と仮説を生み出し、モデル設計にも反映できます

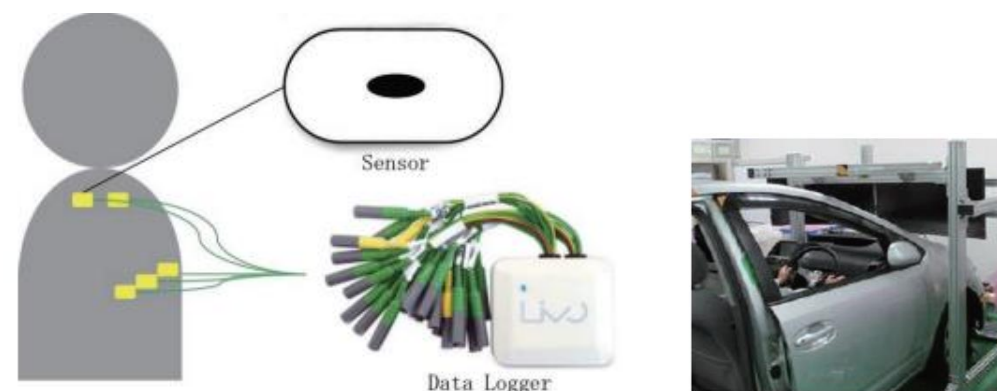
## 感情誘起のデータセット作成

自らデータ  
取得

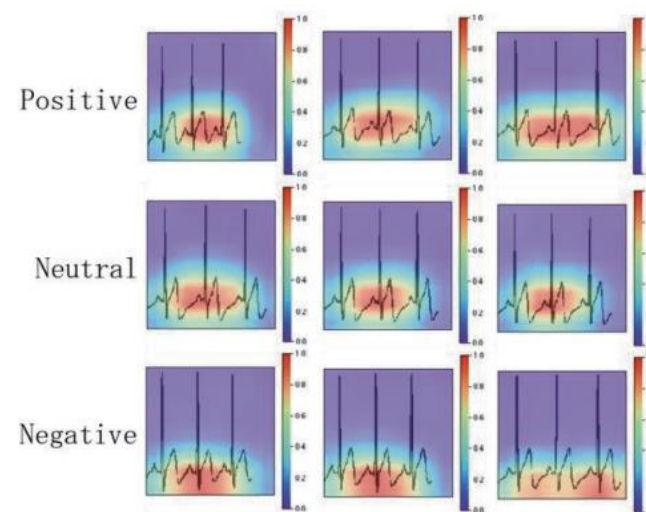
初期分析

実験方法  
アップデート

少量実験

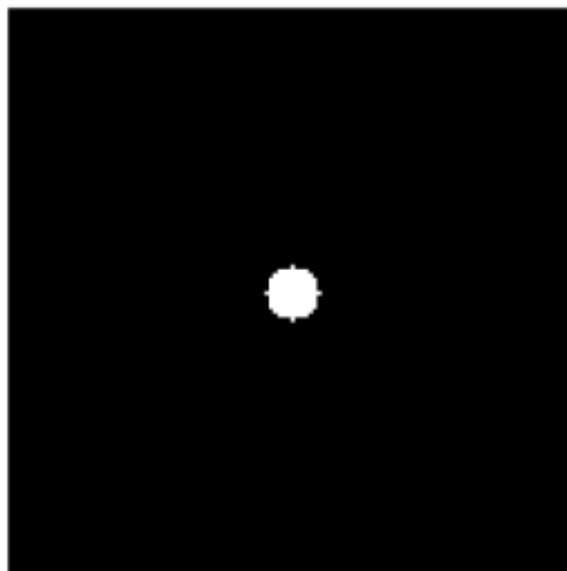


- 自分が被験者となってデータ取得
- 心電図センサーの貼り方もデータの見方もよくわからずという状況からドメイン知識を獲得
- データはなければ自ら作るという姿勢が大事

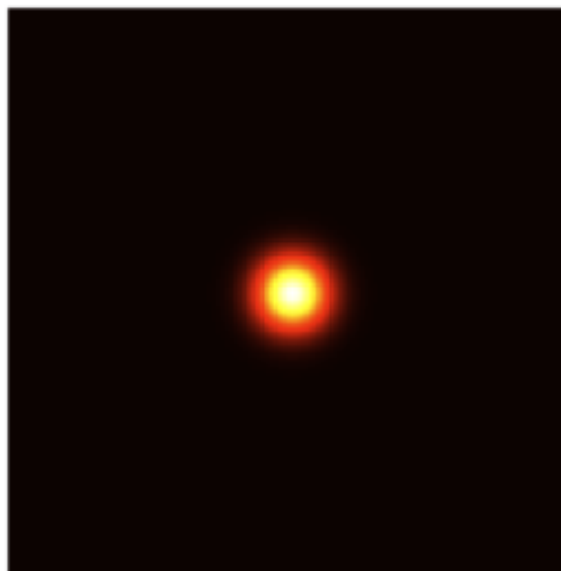


人がラベルをつける場合にはラベルがブレることが当たり前です。道路下の空洞を検出するというプロジェクトでは境界も曖昧で実際にどこまで空洞か判断が付きにくい状態でした。そこで、広がりを持たせるようなラベル付で揺れに強いモデルを作成しました

人のラベル付

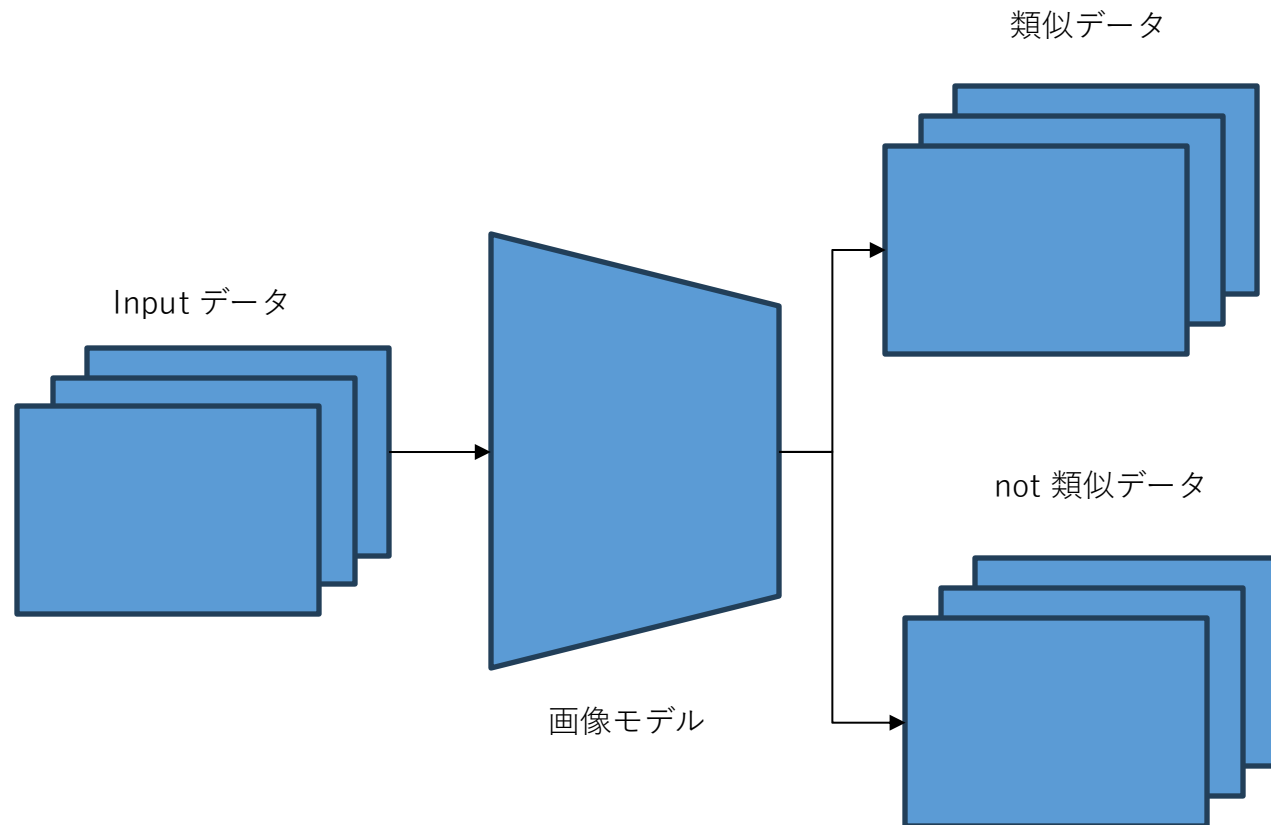


中心から徐々に確信度を落とす



- 実際にどこまで空洞が広がっているかは、穴を掘ってみないことには判断できません
- 空洞の広さをハードラベルでつけることよりソフトラベルで学習させる

走行データや工事現場の「類似シーン」のレコメンドエンジンを作成するプロジェクトにおいて、類似シーンの定義が部署によって異なっていました。ユースケースに合わせて類似している/していないを明確に定め現場に即したモデルを作成しました。



## 走行データにおける類似データ

- 夜間など映像における大部分を占めるデータに引っ張られて類似データが抽出される
- しかし開発現場では、停止線があるかなど開発ロジックの検査を行うためのシーン検出をす

## 工事現場における類似データ

- 工事のリソースや状況だけでなく、周辺道路状況や周辺の工務店の数など現場以外の情報が重要

# 1件単位での予測に価値はないことがある



売り上げ冊数を予測するような回帰予測では、ビジネスにおいて意思決定を行うために意味がある単位での予測が大切です。  
問いの立て方を深く考え

## 出版物の予測

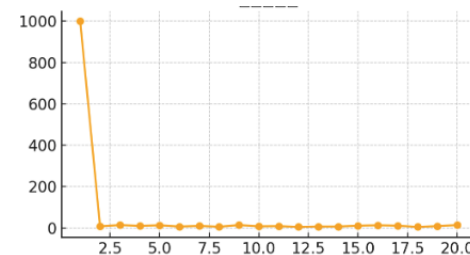
- 生産の現場ではロットで生産されることや各書店での受け入れ冊数の上限が決まっていることもあるため、1冊単位での回帰予測はビジネス効果の算出で意味がないこともありうる

- 100部以上売れる、1000部以上売れるなどの出版物が持つポテンシャルを当てるようなOrdinalRegressionとして解く
- 5,000部以上売れる確率が100%で10,000部以上売れる確率が50%の時に、その出版物に営業部として勝負をかけるべきかどうかの意思決定支援ツールとしてのAI開発

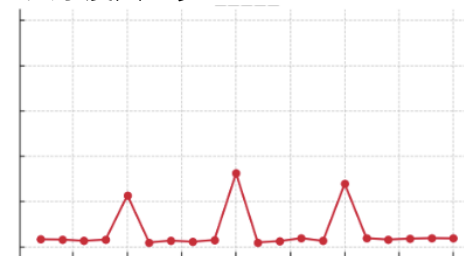
## 出版物の売り上げカーブ類型化

- ビジネスの観点ではさらに「いつ重版をかけるべきか」という問いもありました
- 重版のタイミングについては、時系列に売り上げカーブを並べることで類型化を実施しパターンに応じた戦略をとることとしていました

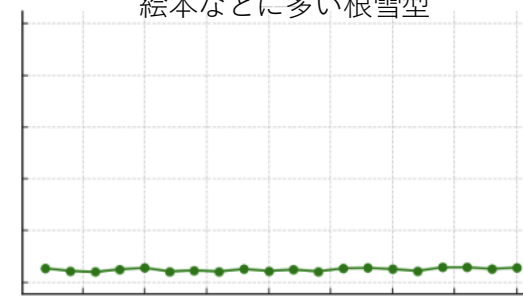
ジャニーズなどの写真集に多いジャニーズ型



大手漫画に多いビッグタイトル型



絵本などに多い根雪型



モデルの複雑さが精度向上につながることもありますが、継続運用・保守の連携も考えた場合に持続可能なシンプルな構造が求められるケースもあります

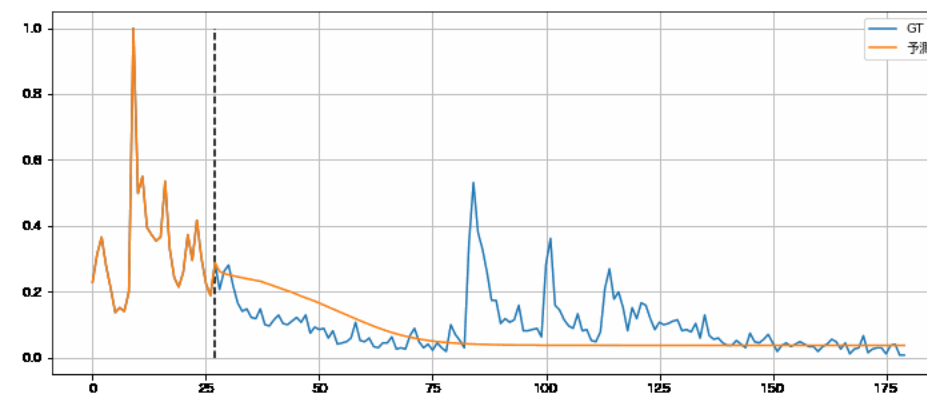
## 出版物の予測

- 最初プロジェクトに入った時には、週ごとの売り上げ予測モデルが作られており12週先まで予測するために12個のモデルが運用されていた
- モデルの精度が落ちてきた際に特定のモデルだけ再画集をかけるのか、運用ルールが複雑になりがちでデータ分析に明るくない組織では運用が立ち行かなくなる懸念がありました



- シンプルに一つのモデルで解くように、LightGBMの特徴量を下記のように前処理を実施
- $n-14$ 日目の売り上げ、 $n-13$ 日目の売り上げ、 $n-12$ 日目の売り上げ、 $\dots$   $n-2$ 日目の売り上げ
- 連続する13日間の売り上げ情報 + 出版物の属性情報をベースに  $n$ 日目の売り上げを予測

## 予測結果の見せ方の工夫

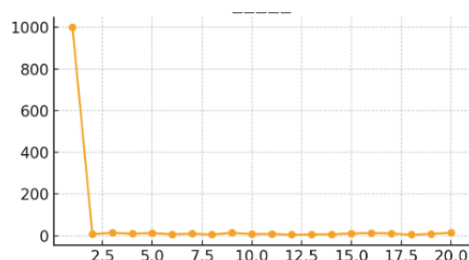
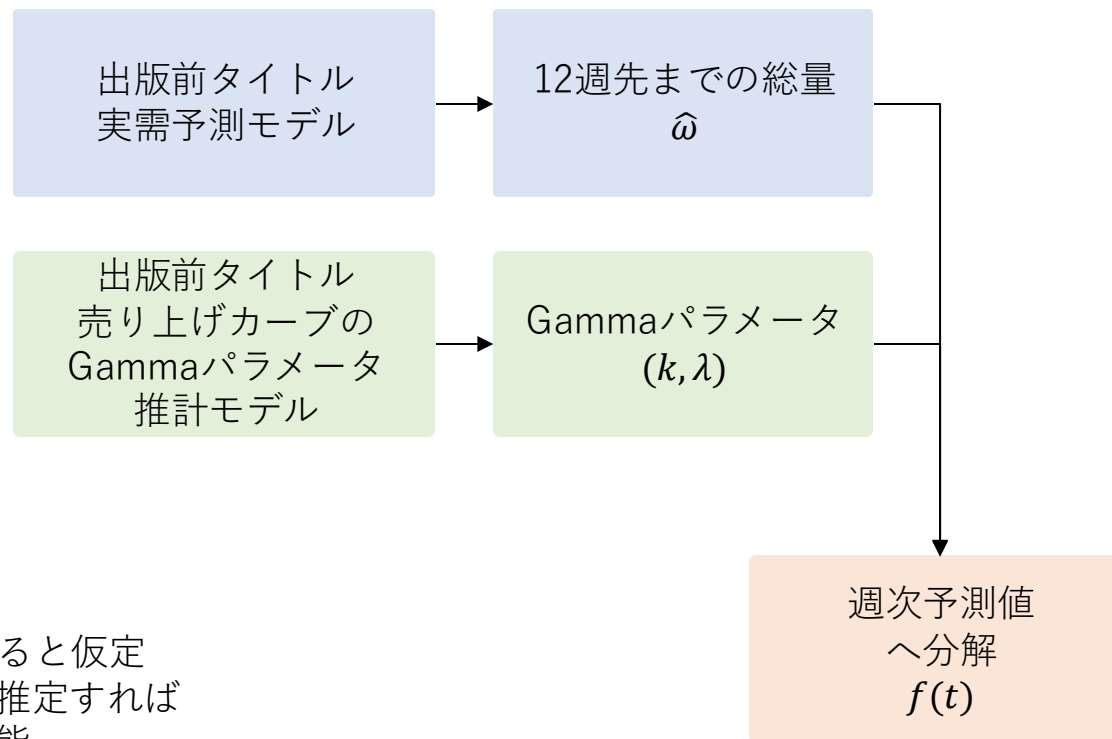


- $n$ 日時点までのデータを使って、 $n+12$ 週のデータを予測
- 売り上げが上がったタイミングから追従するような形を見えやすくするため動画にして呈示

モデルの数だけでなく、売り上げカーブを推定するためにトライしていたこととして  
比較的当てやすい総量とカーブのパラメータを機械学習で予測し分解するような取り組みも行っていました

## 出版物の予測

- 最初プロジェクトに入った時には、週ごとの売り上げ予測モデルが作られており12週先まで予測するために12個のモデルが運用されていた
- モデルの精度が落ちてきた際に特定のモデルだけ再画集をかけるのか、運用ルールが複雑になりがちでデータ分析に明るくない組織では運用が立ち行かなくなる懸念がありました



- 売り上げカーブはガンマ分布に則ると仮定
- 総量とガンマ分布のパラメータを推定すれば  
週次ごとの売り上げ予測に分解可能

## キーメッセージ

- ・ AIは現場でみんなで育てていく
- ・ 綺麗なデータも正確なラベルも評価指標も最初から揃っていることはありません
- ・ 今あるもので仮説を立てて、工夫して補い、試行錯誤を繰り返すことで現場に定着させる



- ・ AIを育てるとは、データの取得/理解を自ら行いモデルの精度を上げていくことです。
- ・ 泥臭く面倒なことにこそ価値があります。

