

データ分析コンテストの技術と 最近の進展



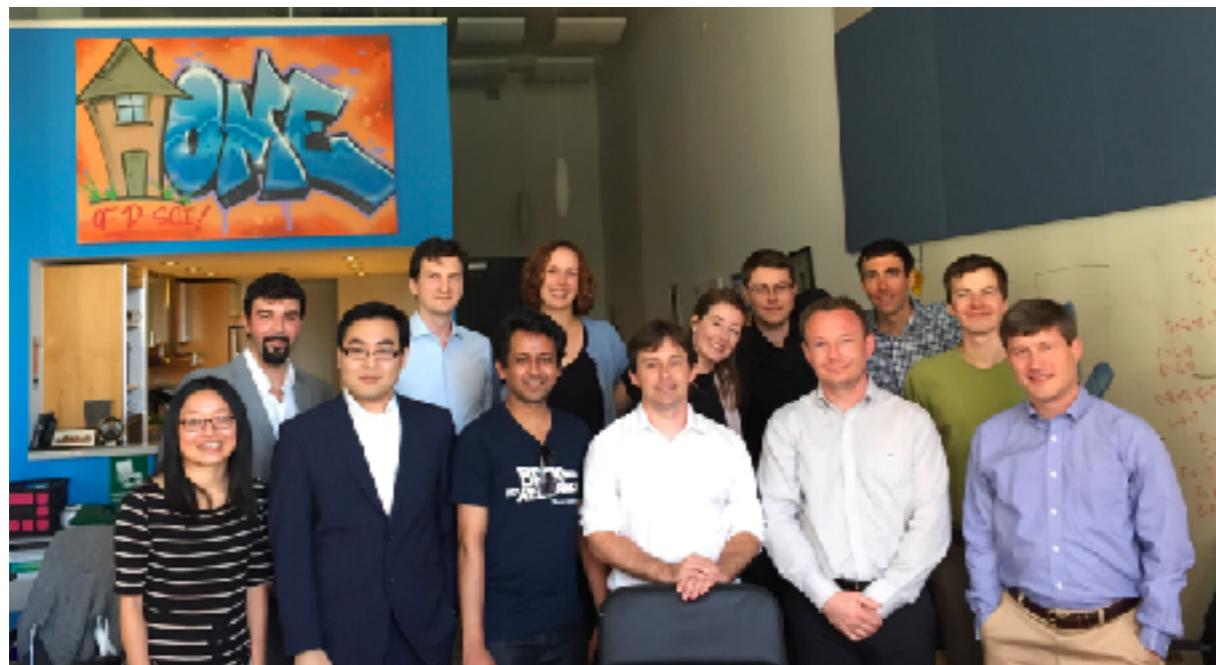
Kohei Ozaki (@smly)
Recruit Technologies
Advanced Technology Lab (ATL)

自己紹介

リクルートテクノロジーズ ATL の Sr. Software Engineer です。

- Kaggle 歴 7 年 (Grandmaster, Highest rank: 4th)
- Kaggle (Top10 finishes x10, Prize x3)
- ACM/KDD, KDD Cup 2015 1st prize winner
- TopCoder Marathon Match (3 WINS)

▼ Kaggle オフィスでの集合写真 (サンフランシスコ)



▼ リクルートテクノロジーズ ATL (広尾)



今日の話

- コンテストと Leaderboard のメカニズム (20)
- メタ学習 (Stacking, Netflix Blending, etc⋯⋯) (20)
- 深層学習 (Classification, Segmentation) (20)

今日の話

- コンテストと Leaderboard のメカニズム (20)
- メタ学習 (Stacking, Netflix Blending, etc⋯⋯) (20)
- 深層学習 (Classification, Segmentation) (20)

データ分析コンテストのプラットフォーム

Kaggle はコンテストの最も大きなプラットフォーム兼コミュニティ

- 120万人以上の登録ユーザー (アクティブは60万人以上)
- 様々な分野の企業や団体がコンテストを開催している

普段アクセスできない様々な業界のデータや問題設定に触れ、
データ分析の経験を積むことが出来る。



参加者の多様性

機械学習ライブラリの開発者や研究者も参加している。

XGBoost, RGF, LibFM, Lasagne, Keras, MXNet などなど…

Rie Johnson (RJ Research Consulting)
an prize winner in Heritage Health Prize

Our original motivation for entering the contest
was to try out our new tree ensemble
regularized greedy forest (RGF)
in a competitive setting.

(quote from <http://www.heritagecaliforniaaco.com/?p=hpn-today&article=45>)

コンテストで利用したアルゴリズムや工夫が論文として公開されるこ
とも珍しくない [Wang & Yang '17]

タスクの多様性

応用を意識したタスク設計が多く、参考になる。

期待値を見積もるための経験が積める。

▼ 例: Yelp Restaurant Photo Classification

Multi-instances, Multi-labeling



お店に対応する
ラベルを複数選択する

屋外席あり

ランチ可

アルコールあり

ディナー可

値段が高い

ビュッフェ形式

入力: お店に関する複数の画像

出力: お店に関する複数のラベル

* 写真 1 枚ずつにラベルが付いていない。
複数画像を単一のベクトルで表現するなど工夫が必要

一般的なコンテストの構造

訓練データ, テストデータが与えられる.

テストデータは更に「Validation」 「Holdout」 の2種に分かれている.



一般的なコンテストの構造

訓練データ, テストデータが与えられる.

テストデータは更に「Validation」 「Holdout」 の2種に分かれている.



① 予測結果

1. 参加者は訓練データとラベルから、テストデータのラベルを予測
2. 予測結果をサーバーにアップロードすると、サーバーが即座に「Validation」の評価を計算して暫定順位表 (Public LB) に反映。
3. コンテスト終了後に提出済みのファイルにおける「Holdout」での評価結果にもとづいて最終順位表 (Private LB) が公開される。

一般的なコンテストの構造

訓練データ, テストデータが与えられる.

テストデータは更に「Validation」 「Holdout」 の2種に分かれている.



1. 参加者は訓練データとラベルから、テストデータのラベルを予測
2. 予測結果をサーバーにアップロードすると、サーバーが即座に「Validation」の評価を計算して**暫定順位表 (Public LB)**に反映。
3. コンテスト終了後に提出済みのファイルにおける「Holdout」での評価結果にもとづいて**最終順位表 (Private LB)**が公開される。

一般的なコンテストの構造

訓練データ, テストデータが与えられる.

テストデータは更に「Validation」 「Holdout」 の2種に分かれている.



1. 参加者は訓練データとラベルから、テストデータのラベルを予測
2. 予測結果をサーバーにアップロードすると、サーバーが即座に「Validation」の評価を計算して暫定順位表 (Public LB) に反映.
3. コンテスト終了後に提出済みのファイルにおける「Holdout」での評価結果にもとづいて最終順位表 (Private LB) が決まる.

予測結果を投稿する

Overview Data Kernels Discussion Leaderboard Rules Team

My Submissions

Submit Predictions

Make a submission for **Kohei**

You have 5 submissions remaining today. This resets 3 hours from now (00:00 UTC).

Step 1

Upload submission file



Upload Submission File

File Format

Your submission should be a single file.
You can upload this in a zip archive, if you prefer.

サーバーに予測結果をアップロードする

Overview Data Kernels Discussion Leaderboard Rules Team My Submissions Submit Predictions

#	△1w	Team Name	Kernel	Team Members	Score ⓘ	Entries	Last
1	—	minu			0.76905	25	7h
2	▲ 1	AirRobot			0.74552	13	2d
3	new	iFighting			0.73789	11	5h
4	▼ 2	Optimus Prime			0.73149	9	2d
5	▼ 1	anokas			0.72541	13	5h
6	▲ 29	10011000			0.72490	14	1d
7	new	No Smoking			0.71930	4	4d
8	▼ 3	dudemeister			0.71639	10	5d
9	new	Kohei			0.71334	8	1d

Your Best Entry ↑

Your submission scored 0.71287, which is not an improvement of your best score. Keep trying!

Validation set でスコアが評価され、
Public LB に即座に反映される

過学習 (Overfitting) に注意

Public LB で表示される数字はあくまでも「Validation」上の評価。
この数字だけを頼りにチューニングしすぎると、
データの事例数やデータセットの特性によっては過学習を引き起こす。



条件が揃えば「データを一切見ること無く」

機械的に Public LB のスコアだけを上げることが容易にできる。

→ 敵対的ベンチマーク

敵対的ベンチマーク [Blum & Hardt '15]

単純化した N 個のテスト事例の二値分類問題を考える：

N 次ベクトル $y \in \{0, 1\}^N$ を当てる。評価指標はエラー率とする。

ある予測 $y_i \in \{0, 1\}^N$ の Public LB スコアを $s_H(y_i)$ とする。

正解	予測 1	予測 2
$y = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ Val	$y_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}$ Public LB Score 1 $s_H(y_1) = 0.75$	$y_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$ Public LB Score 2 $s_H(y_2) = 0.25$

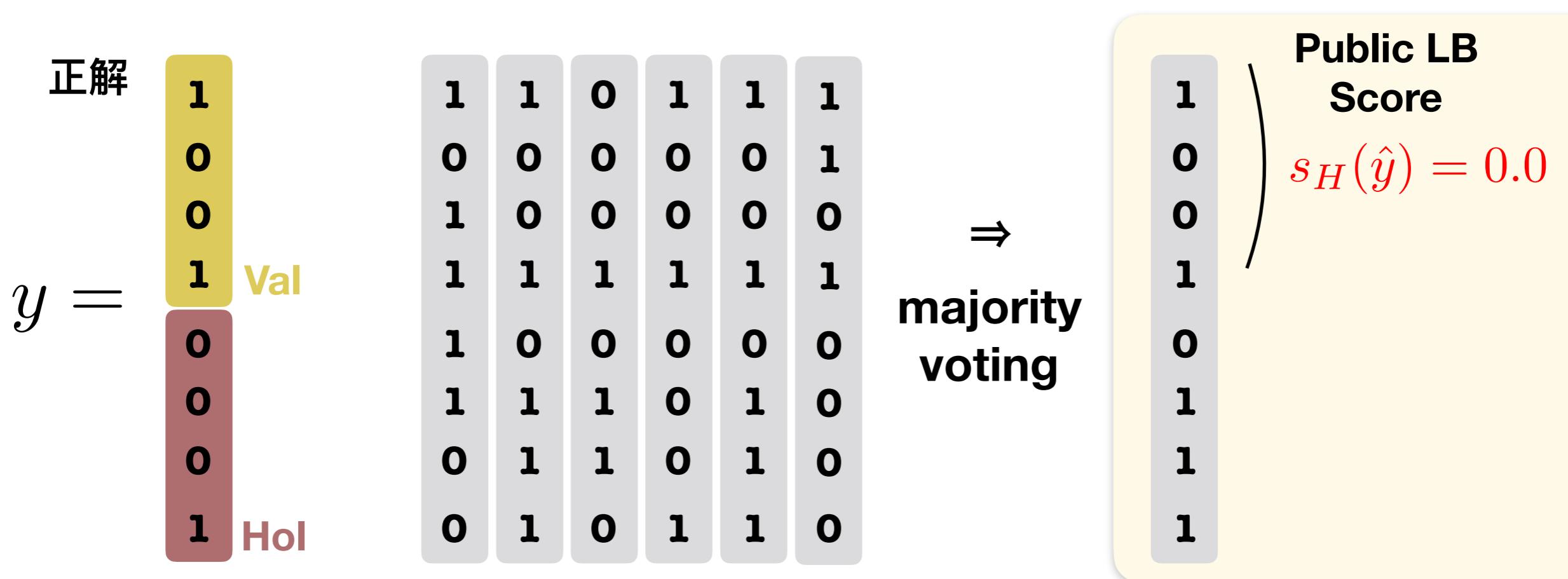
敵対的ベンチマーク [Blum & Hardt '15]

Algorithm (Boosting Attack):

ランダムに予測のベクトルを作成する

敵対的ベンチマーク [Blum & Hardt '15]

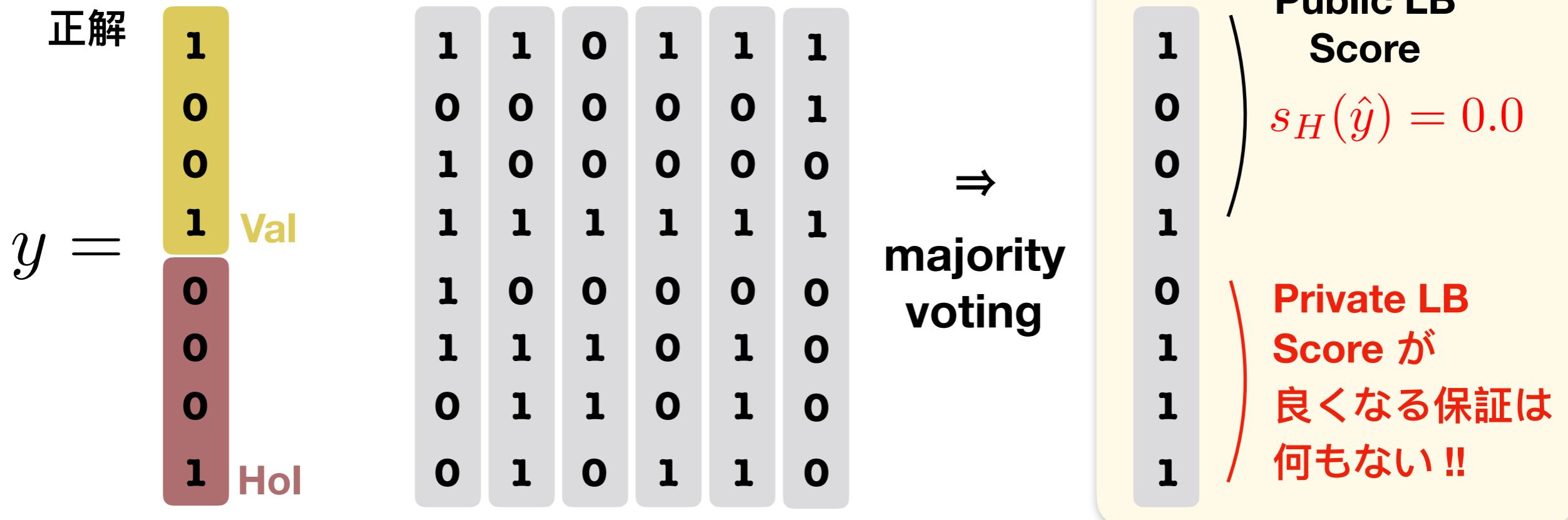
Algorithm (Boosting Attack):



ランダムな予測のベクトルから
Public LB スコアの良いベクトル
 $s_H(y_i) < 0.5$ だけを選ぶ

敵対的ベンチマーク [Blum & Hardt '15]

Algorithm (Boosting Attack):



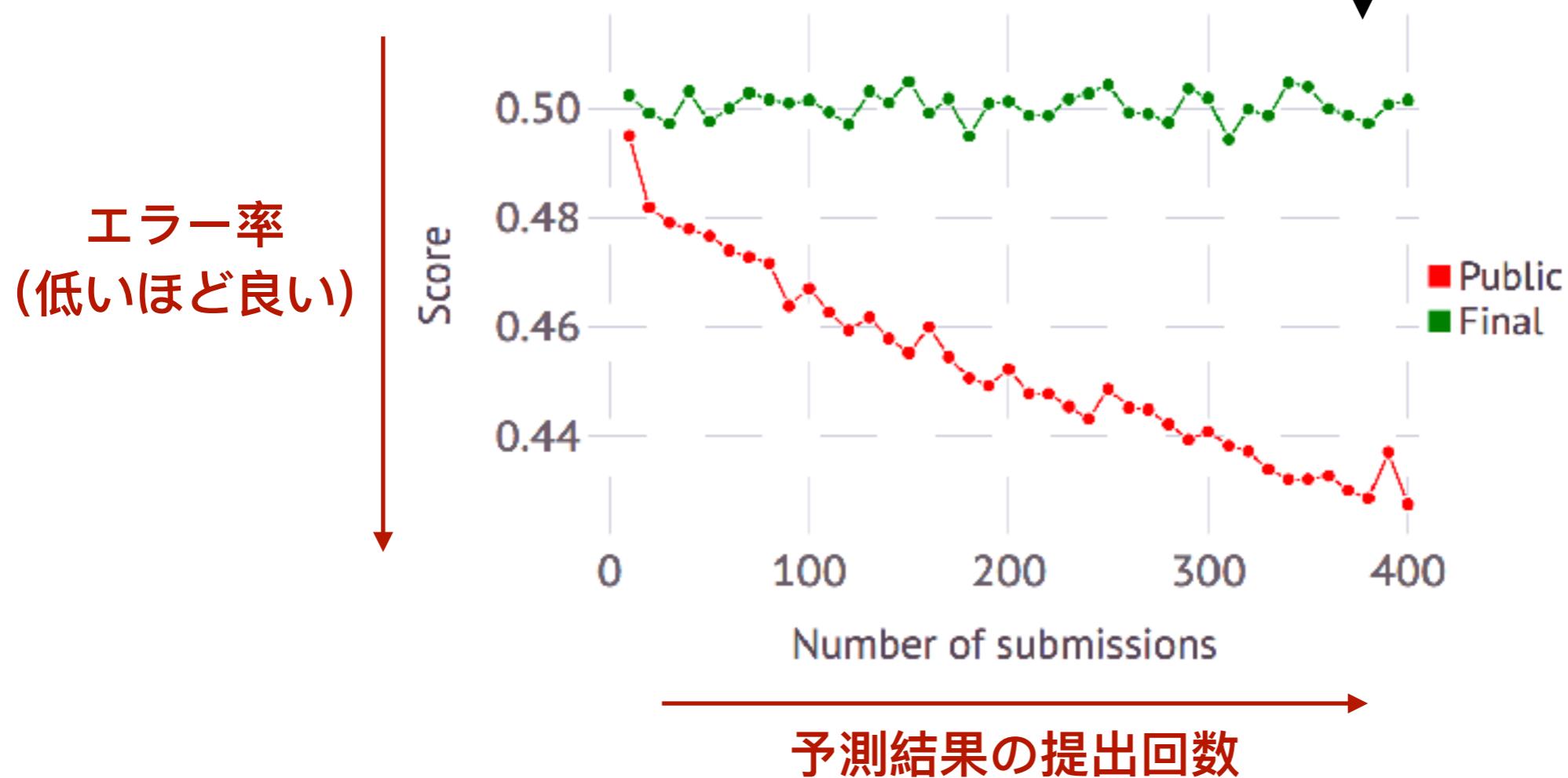
ランダムに予測のベクトルから
Public LB スコアの良いベクトル
 $s_H(y_i) < 0.5$ だけを選ぶ

敵対的ベンチマーク [Blum & Hardt '15]

最終的な順位を決定する Private LB スコアは改善しない。

Public の評価に使われるデータに overfit している。

最終順位の評価対象である Holdout (Final) は改善しない



Boosting attack は LB スコアの桁数に依存

Boosting attack で得られる Public LB score の
Upper bound を証明することができる。

$$\text{Error rate} \leq \frac{1}{2} - \Omega\left(\sqrt{\frac{k}{n}}\right)$$

k=投稿回数, n=テストセットの事例数

LB スコアの小数値の精度 α と
Validation セットのサイズ n の間に以下の条件がある。

$$\alpha \leq \frac{1}{\sqrt{n}}$$

α を大きく（表示桁数を小さく）， n を大きくすることで
敵対的なベンチマークに対する耐性を持たせることが出来る。

信頼性のある順位表

最近の Kaggle ではテストセットのサイズが小さい場合、コンテスト期間中だけ表示行数を減らしていることがある

#	△1w	Team Name	Kernel	Team Members	Score ⓘ	Entries	Last
1	new	Michael Jahrer			0.287	10	2h
2	new	Tehelka		 	0.283	25	3h
3	new	New Model Army			0.283	5	9h
4	new	No Drivers Installed		 	0.283	9	18h
5	new	ShallowBrain Machineless Na...			0.282	30	7h
6	new	self driving mode turned off			0.282	7	14h

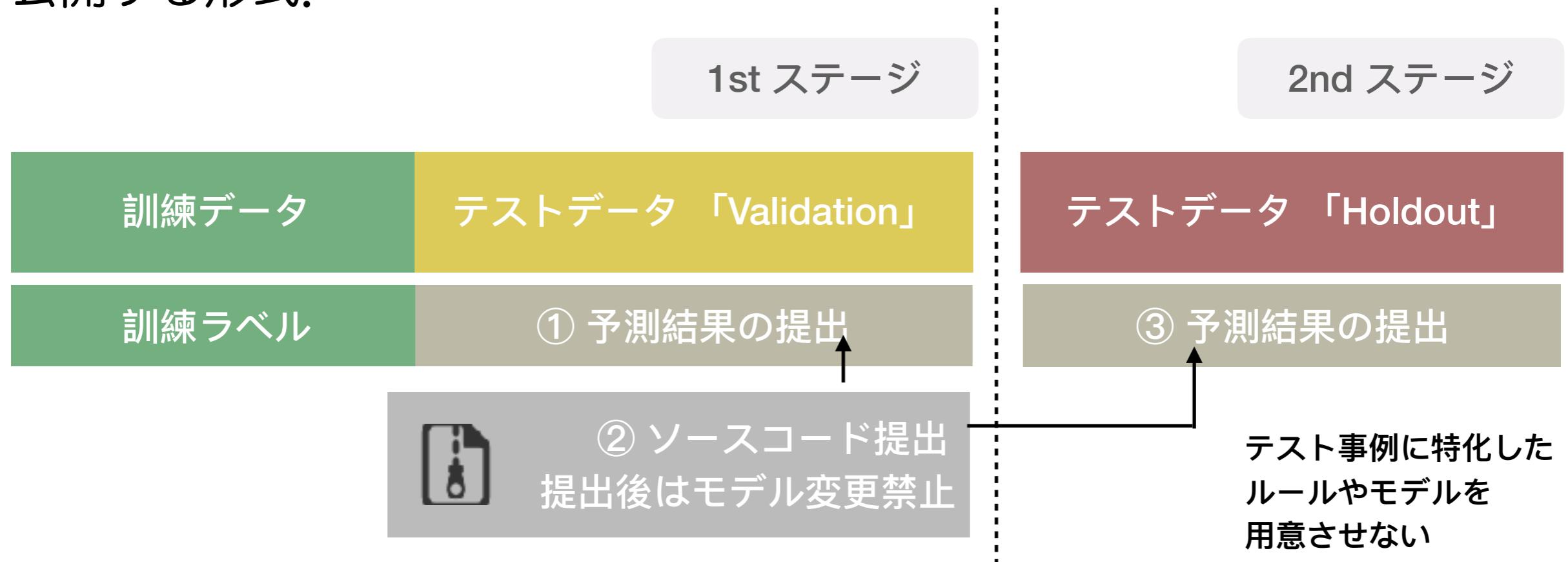
Kaggle のメカニズムと異なるフィードバックの与え方によって、信用できる順位表を提供する研究もある。

[Blum & Hardt '15, Hardt '17]

チート対策: マルチステージな評価

人手でラベル付けて if 文 10000 行の解答は欲しくない

与えられたデータに特化しすぎたモデルを作らせない。
プログラムの提出期限を設けて変更不可とした後で holdout データを
公開する形式。



時系列予測のようなテストデータを使わせたくない場合、
テスト事例が少なくアノテーションが容易である場合などに適している。

今日の話

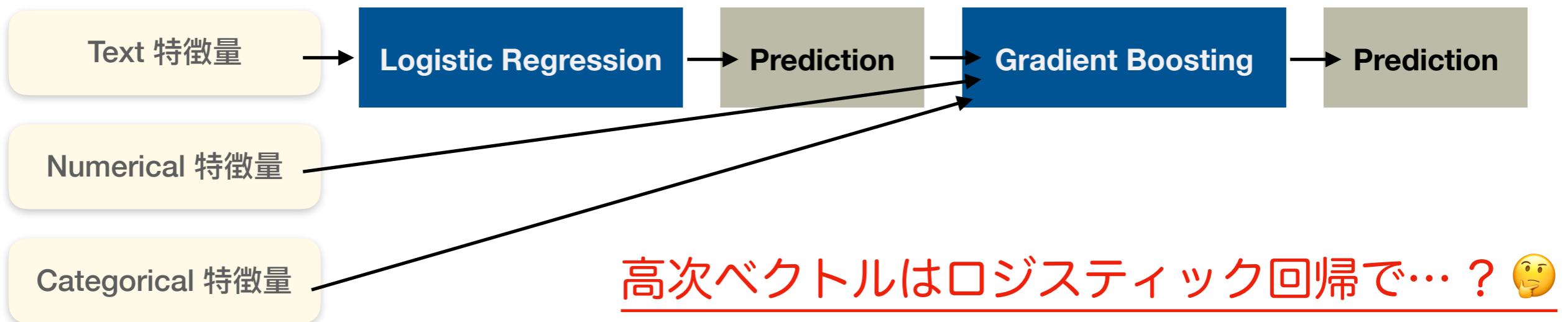
- コンテストと Leaderboard のメカニズム (20)
- メタ学習 (Stacking, Netflix Blending, etc…) (20)
- 深層学習 (Classification, Segmentation) (20)

注)

ここでは neuroscience や強化学習における銅谷らの meta learning ではなく、古典的な帰納学習の文脈における meta learning を指します。

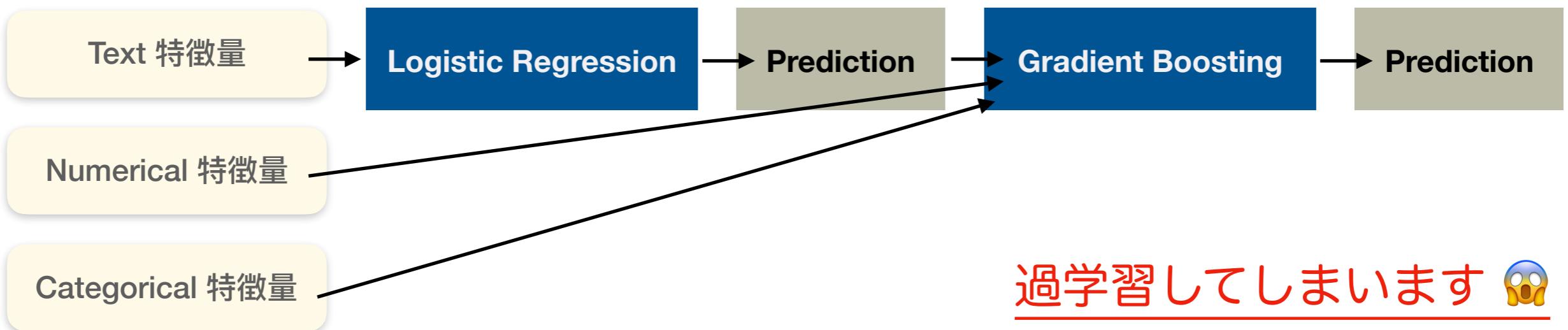
出発点: モデルの出力を特徴量として使う

モデルの出力結果を特徴量として使うことを考える。



出発点: モデルの出力を特徴量として使う

モデルの出力結果を特徴量として使うことを考える。

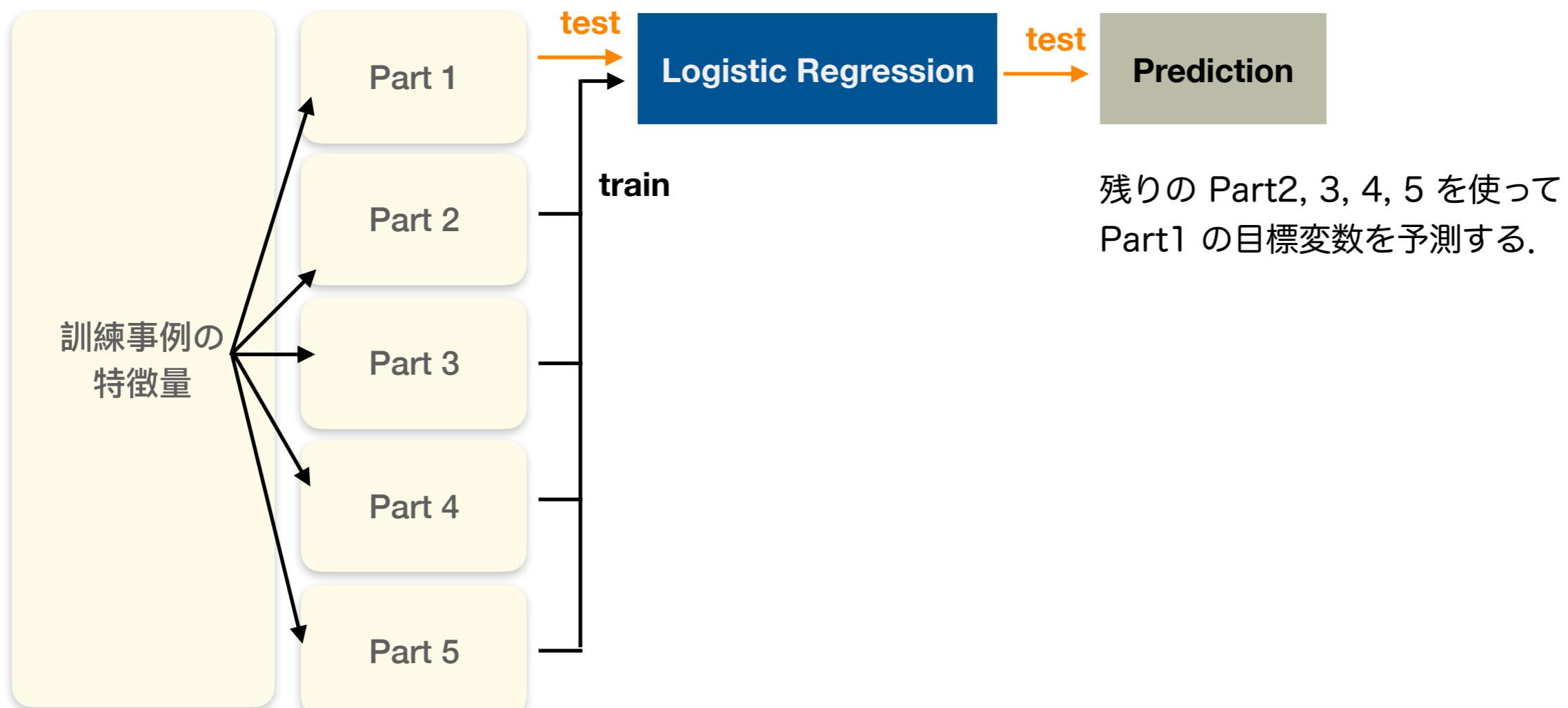


訓練事例に対して過学習を起こしてしまう。
前段のモデルが少しでも訓練データに対して過剰な識別境界を作っているならば、その結果を次段のモデルが更に過剰に学習してしまう。

⇒ 1つ1つの事例に対応するラベルを直接使わないよう注意を払う

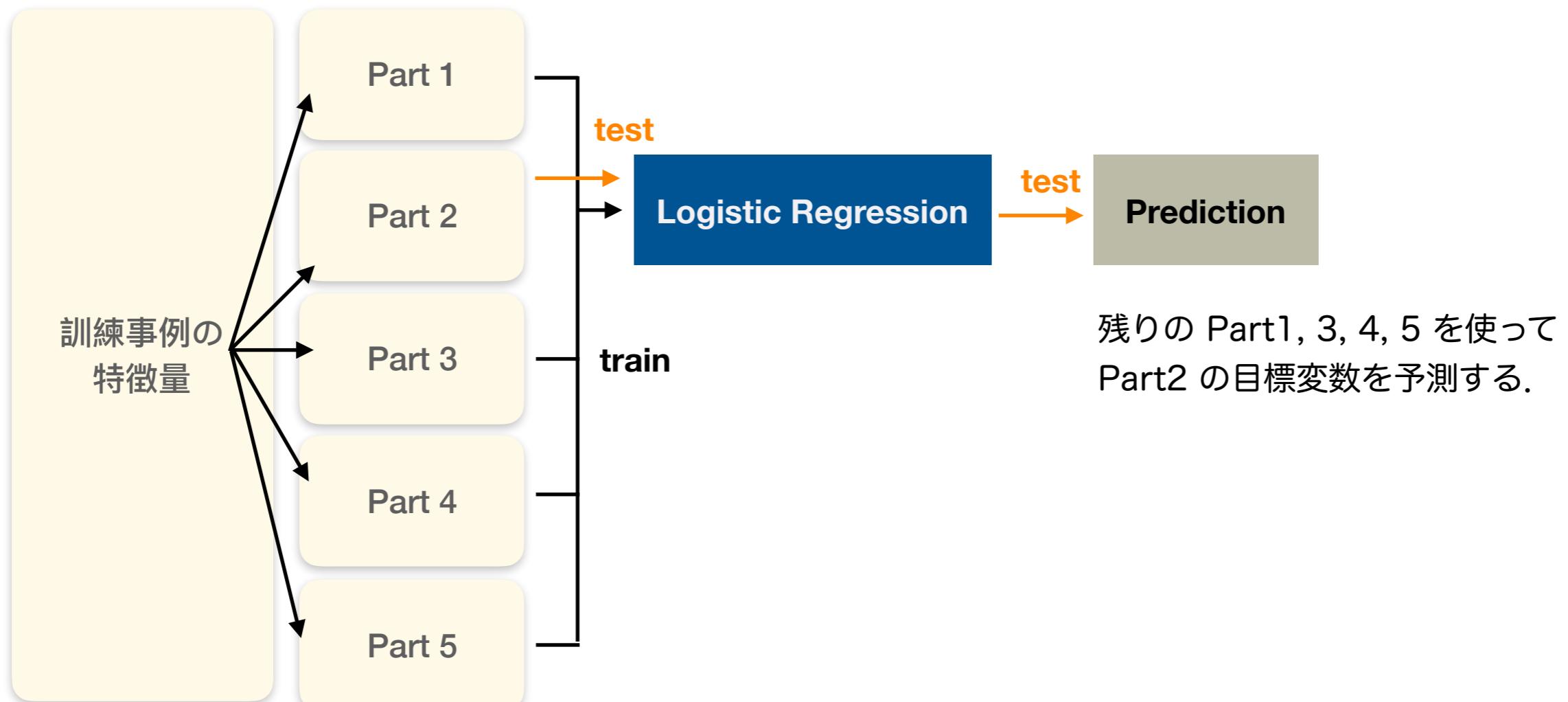
おさらい: k-分割交差確認でモデルを評価

訓練事例を k 分割する。1つの部分を「残りの $k-1$ の部分で予測」する。



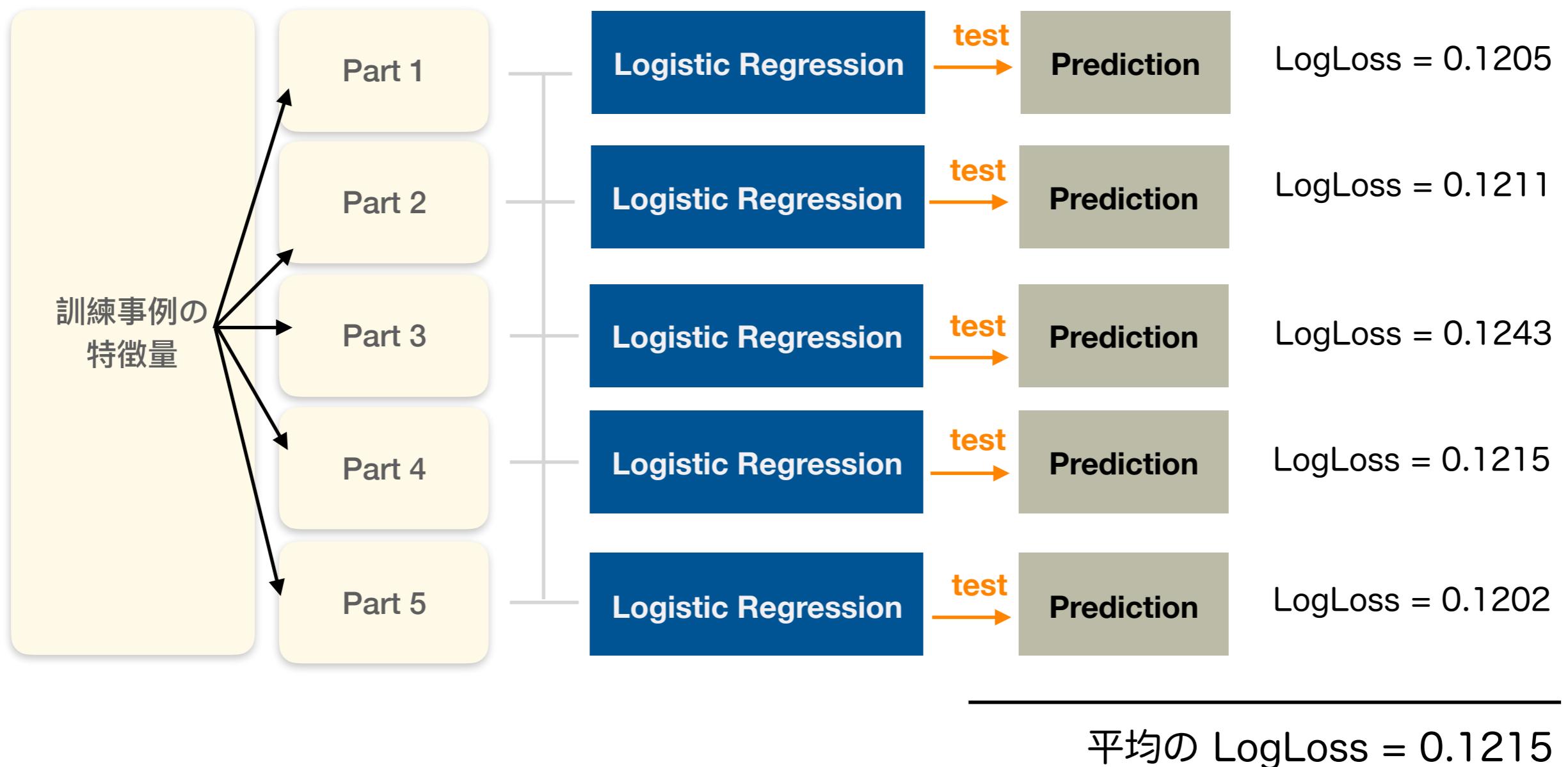
おさらい: k-分割交差確認でモデルを評価

訓練事例を k 分割する。1つの部分を「残りの $k-1$ の部分で予測」する。

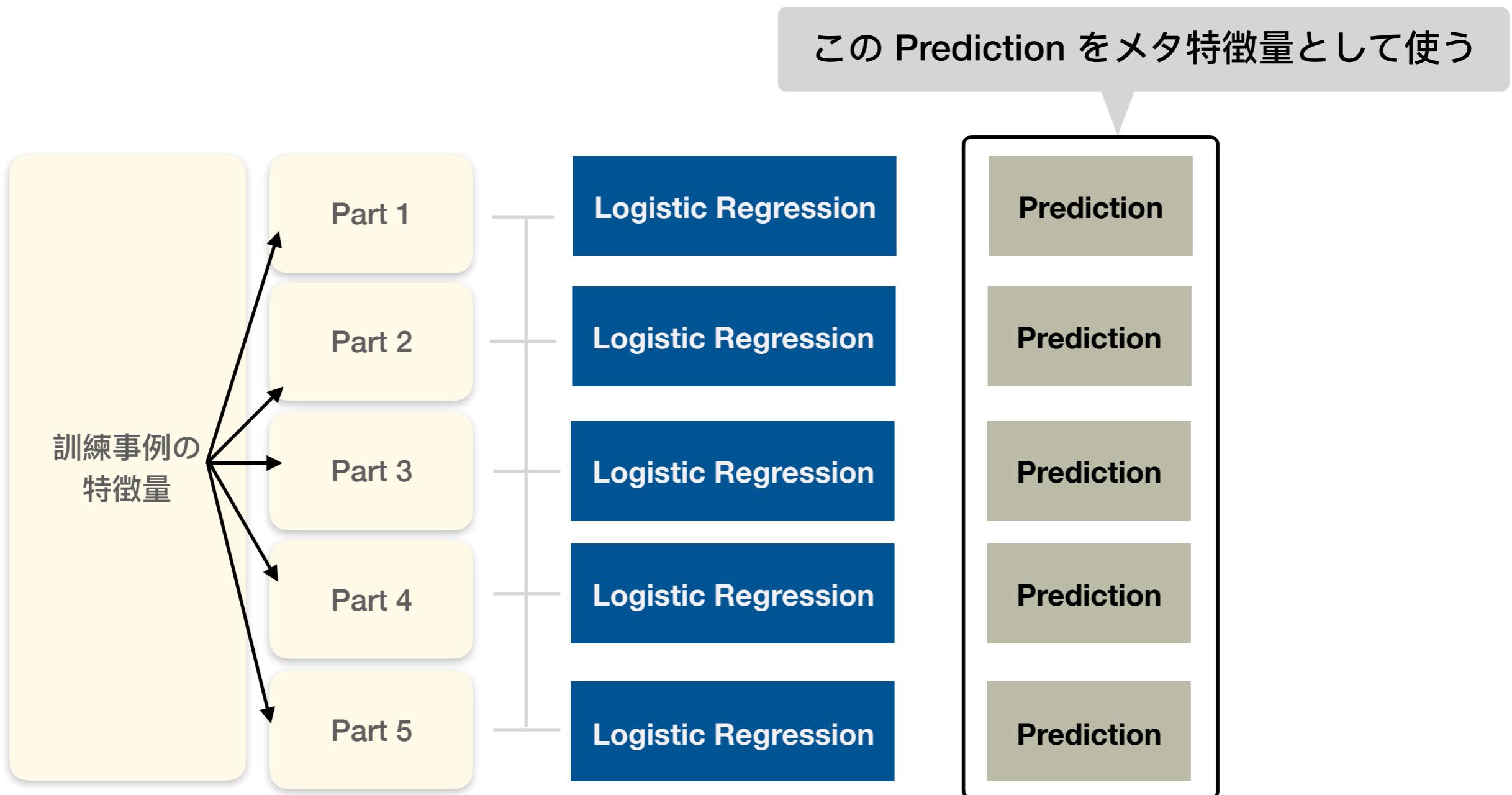


おさらい: k-分割交差確認でモデルを評価

k 個の評価の平均をとり、分割に偏りのない評価の見積もりを得る。



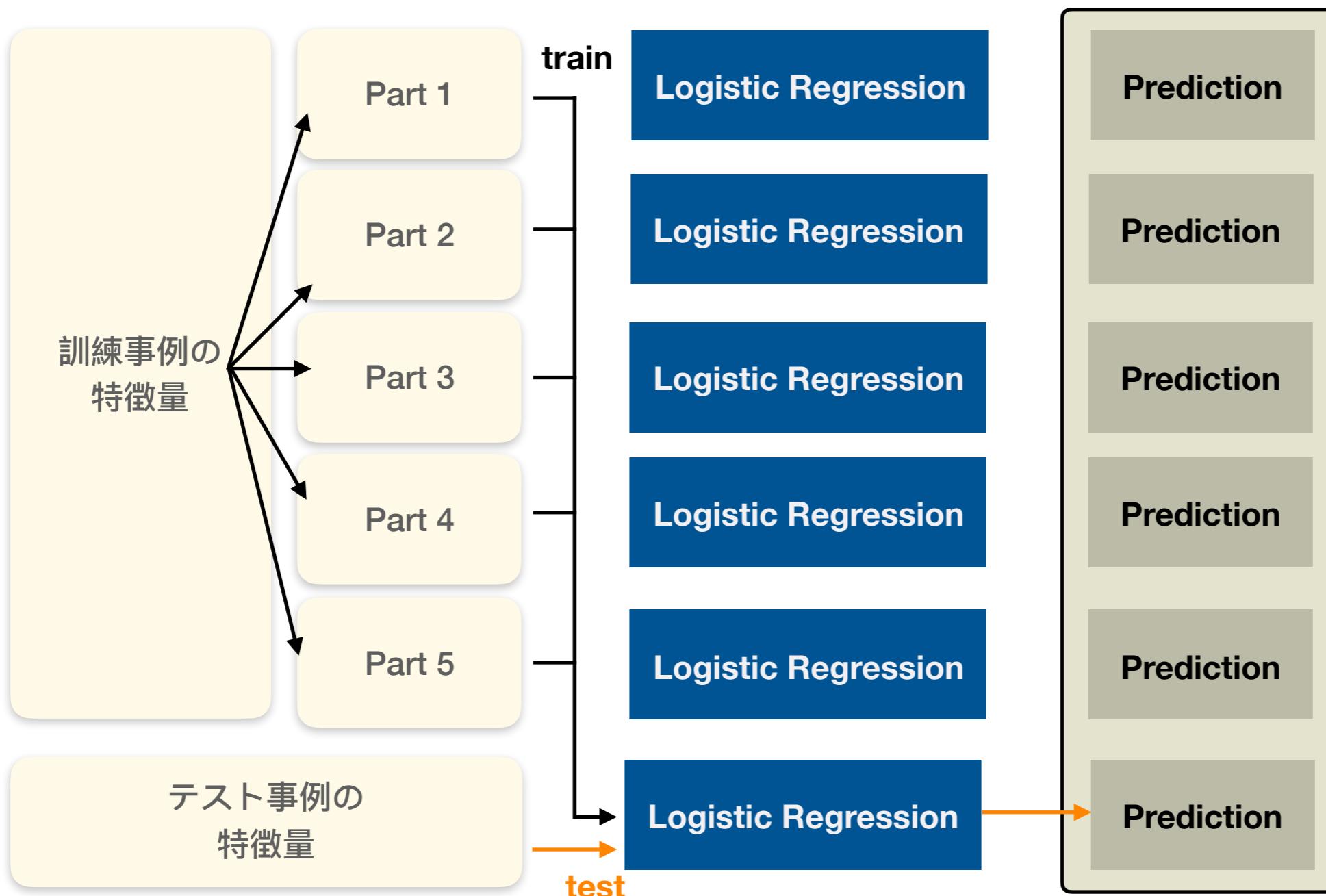
k-分割交差確認の予測を特徴量とする



できたメタ特徴量は個々の事例に対応するラベルを直接使っていない。
(過学習を起こさないと期待できる)

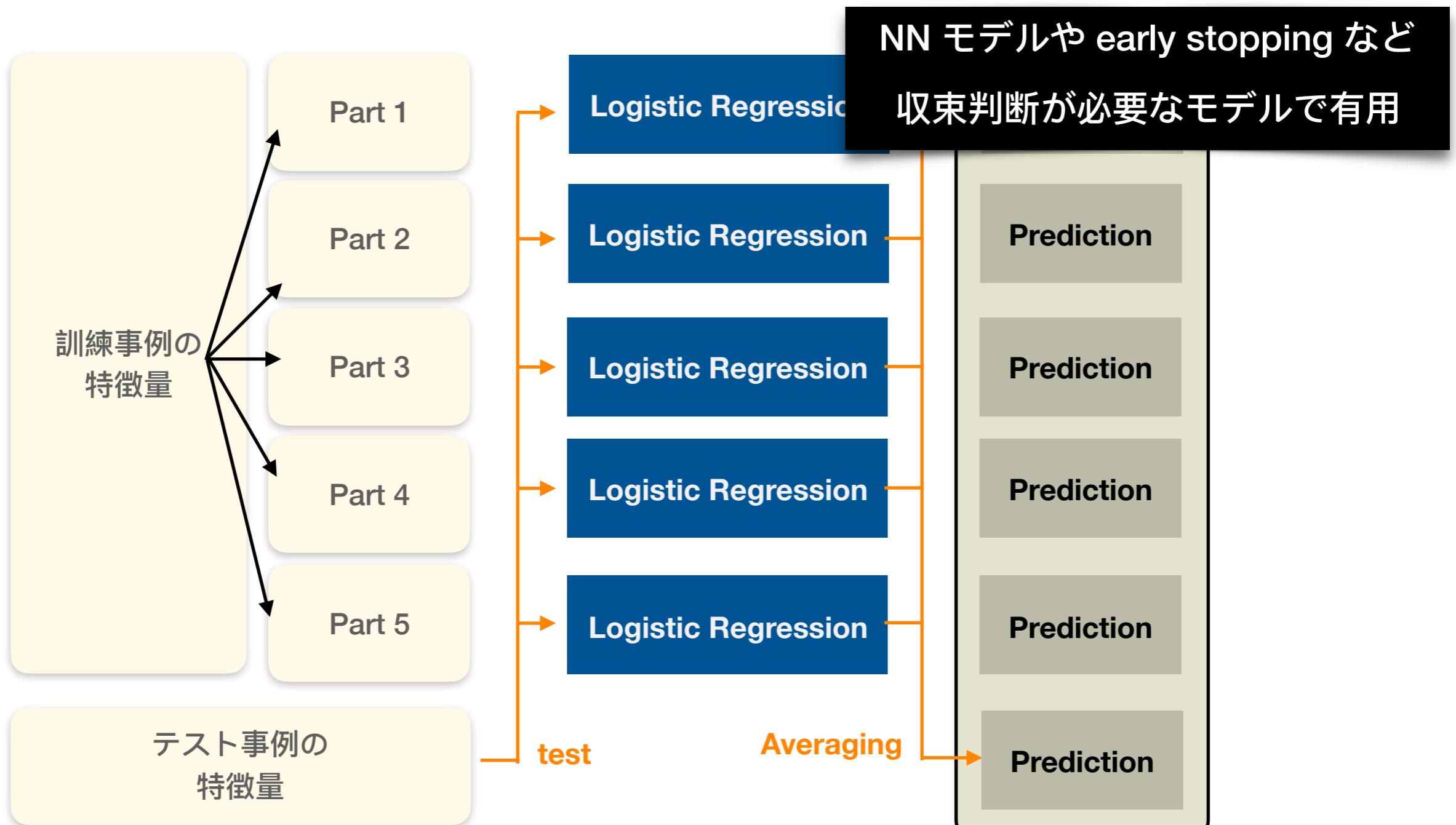
テスト事例のメタ特徴量

テスト事例についても同様にメタ特徴量を用意する。



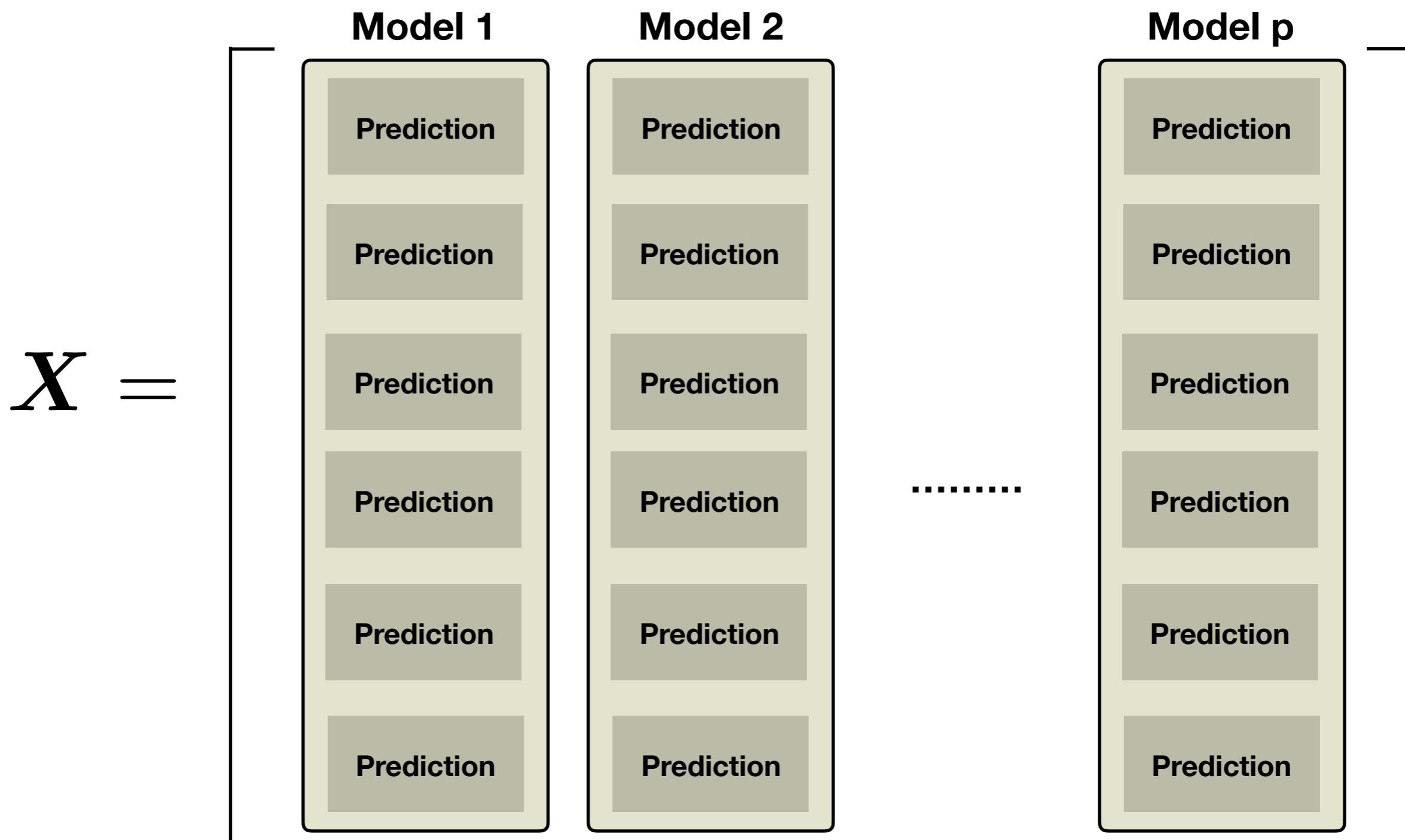
テスト事例のメタ特徴量（工夫）

各分割で作成したモデルの Prediction の平均を使うこともできる。



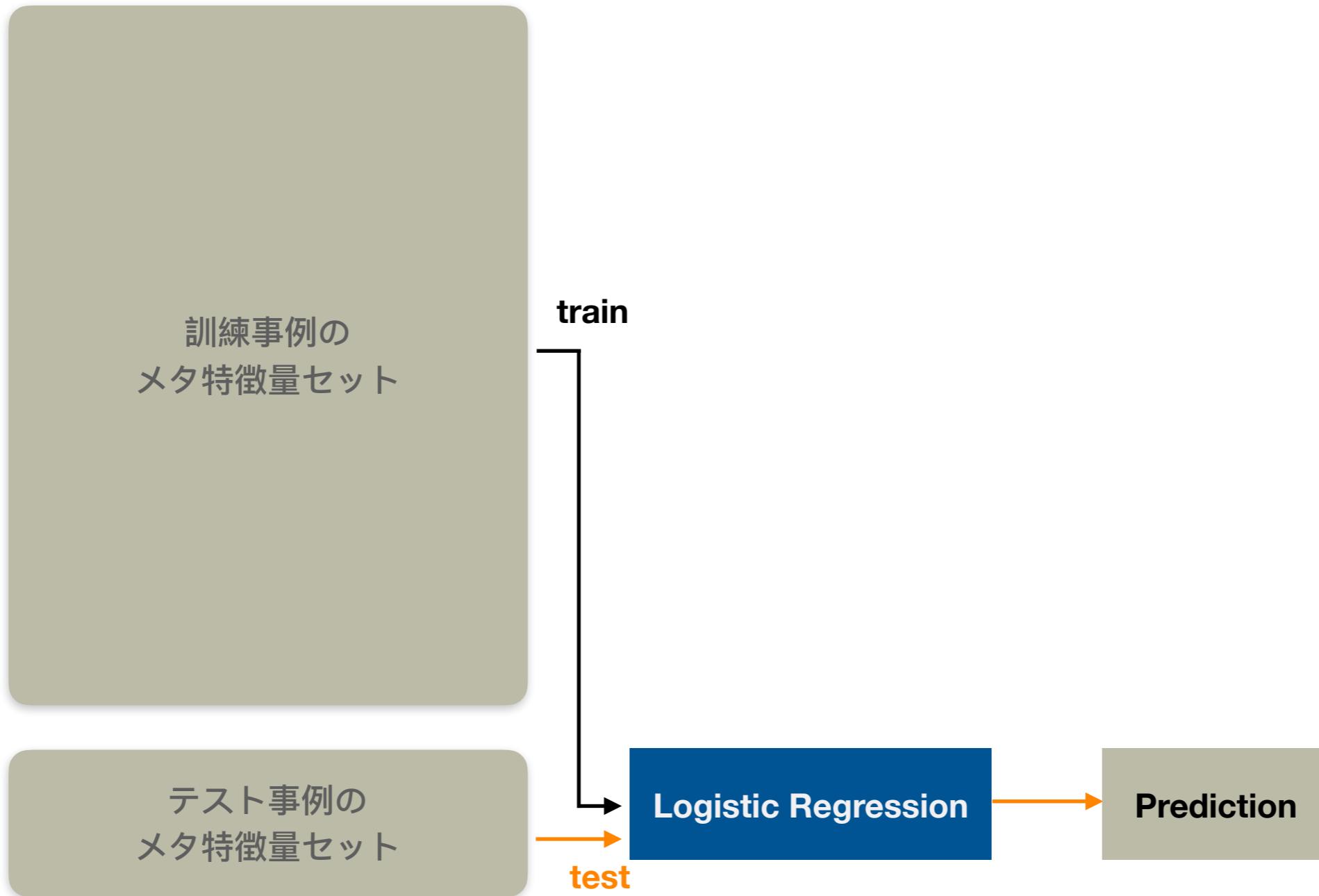
Stacking (Stacked Generalization)

様々なモデルで作成したメタ特徴量を集め、特徴量セットを構成する。



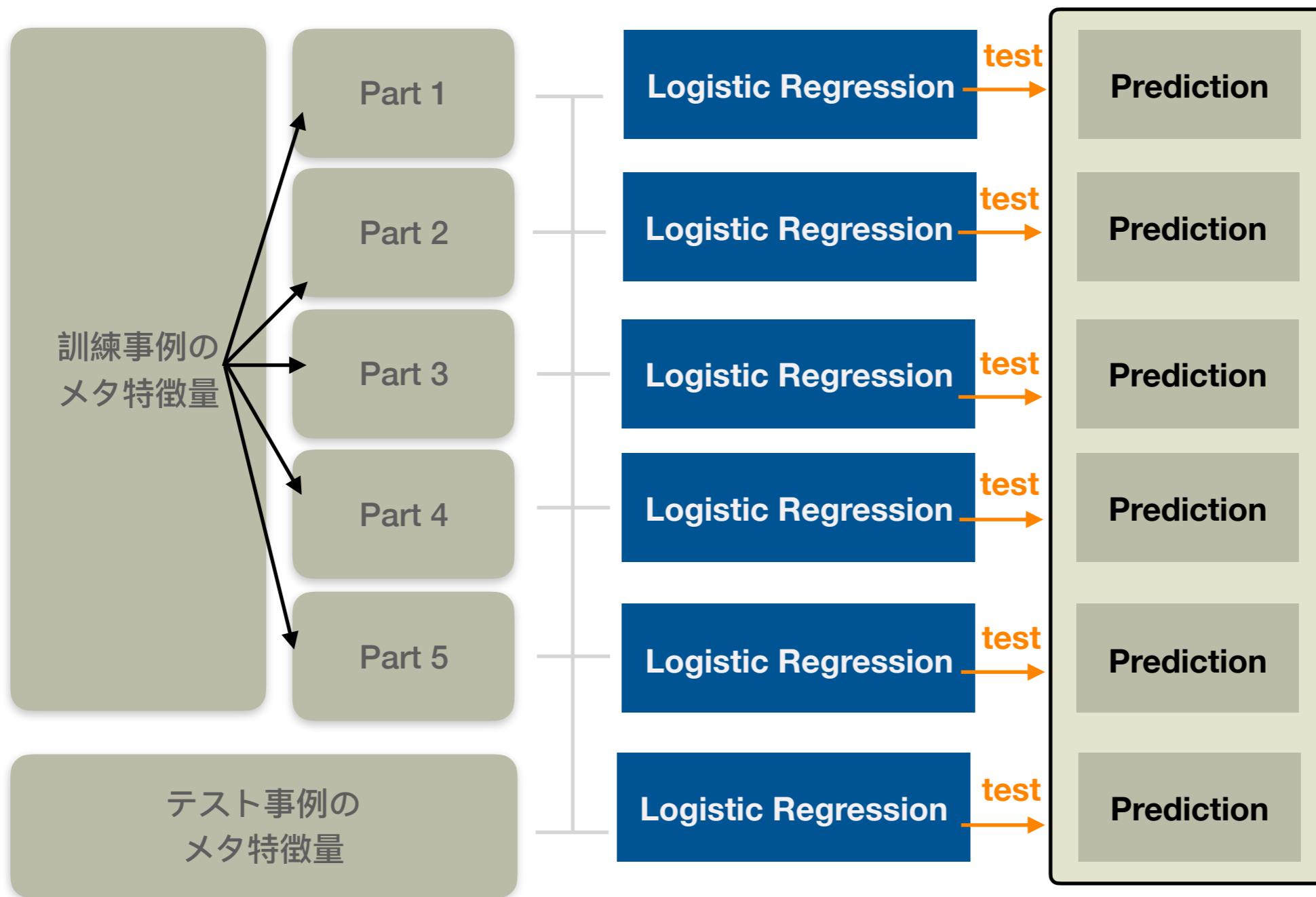
Stacking (Stacked Generalization)

メタ特徴量を使いモデルを学習し、テスト事例の目標変数を予測する。



Stacking (Stacked Generalization)

メタ特徴量から更にメタな特徴量を作成し、多層化することもできる。



Stacking を使う時の注意点

- Stacking に使うモデルの diversity を増やすと効果的。
 - Gradient boosting や Neural network モデルの他にも Logistic regression や k-Nearest Neighbor なども試す
 - 異なる特徴量セット、異なるパラメータ、異なる事例集合などなど…
- Public LB に overfit しないよう気をつける。
 - 特にテスト事例が少ない場合は交差確認 (CV) のスコアを信じる
 - 方法に問題がなければ CV score と Public LB は線形の関係になる。ならなければ原因がどこにあるのか考える。

目標変数を使った特徴量のエンコーディング

Stacking に関する話題として、目標変数を使った特徴量のエンコーディング方法がある。Stacking と同様に 1 つ 1 つの事例に対応するラベルを（分割の中で）直接使わないよう注意を払う。

分割 ID	カテゴリカル変数	目標変数	エンコード
1	エコノミー	3	$(1+4) / 2 = 2.5$
1	ビジネス	10	$(9 + 8) / 2 = 8.5$
1	ビジネス	11	$(9 + 8) / 2 = 8.5$
1	エコノミー	4	$(1+4) / 2 = 2.5$
2	エコノミー	1	$(3+4) / 2 = 3.5$
2	エコノミー	4	$(3+4) / 2 = 3.5$
2	ビジネス	9	$(10+11) / 2 = 10.5$
2	ビジネス	8	$(10+11) / 2 = 10.5$

分割1以外の部分から
カテゴリカル変数が同値である事例の
目標変数の平均でエンコード

Netflix Blending [Töscher & Jahrer]

特殊な事例：Public LB のスコアをヒントとしてアンサンブルする手法

賞金 \$1M (約 9200万円) の Netflix Prize における優勝者の手法。
「アンサンブルを Leaderboard スコアを使った線形回帰に帰着させた」



Netflix Blending [Töscher & Jahrer]

問題：RMSE を評価指標とした予測問題のアンサンブルを考える。

ゴールは個々のモデルの出力する予測値 $y_i \in \mathcal{R}^N$ の線形回帰により
真の値 $y \in \mathcal{R}^N$ に近いベクトルを得ること。

$N \times p$ 行列

N テスト事例, p 個のモデルの予測を結合した行列

$$X = \left(\begin{array}{cccccccccccc} \text{Prediction} & \text{Prediction} \end{array} \right)$$

$y \in \mathcal{R}^N$ 真の値 (未観測) $\curvearrowleft y_i \in \mathcal{R}^N$

Netflix Blending [Töscher & Jahrer]

ゴールは個々のモデルの出力する予測値 $y_i \in \mathcal{R}^N$ の線形回帰により
真の値 $y \in \mathcal{R}^N$ に近いベクトルを得ること。

\mathbf{X}	$N \times p$ 行列 N テスト事例, p 個のモデルの予測を結合した行列
--------------	---

$y \in \mathcal{R}^N$	真の値 (未観測)
-----------------------	-----------

もし y が既知であれば, 正規方程式を用いて得られる最小二乗法の
解によって回帰係数 β が計算できる。

$$\boxed{\mathbf{X}\hat{\beta}} \text{ where } \hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} (\mathbf{X}^T \mathbf{y})$$

線形回帰で真の値に近づける

Netflix Blending [Töscher & Jahrer]

もし y が既知であれば、正規方程式を用いて得られる最小二乗法の解によって回帰係数 β が計算できる。

$$\mathbf{X}\hat{\beta} \text{ where } \hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \underline{(\mathbf{X}^T \mathbf{y})}$$

$$\underline{(\mathbf{X}^T \mathbf{y})} = \begin{pmatrix} : \\ \text{j-th element} \\ : \end{pmatrix} = \begin{pmatrix} : \\ \sum_u x_{ju} y_u \\ : \end{pmatrix} \quad \text{モデル j の予測と y の内積}$$

$$\sum_u x_{ju} y_u = \frac{1}{2} \left[\sum_u y_u^2 + \sum_u x_{ju}^2 - \sum_u (y_u - x_{ju})^2 \right]$$

モデル j の予測と y の内積
近似できる
(すべて 0 と予測した
ときの LB score)

正確に計算できる

LB score で近似できる
(MSE に N を掛ける)

すべて計算可能 & 線形回帰が可能となる ■

今日の話

- コンテストと Leaderboard のメカニズム (20)
- メタ学習 (Stacking, Netflix Blending, etc⋯⋯) (20)
- 深層学習 (Classification, Segmentation) (20)

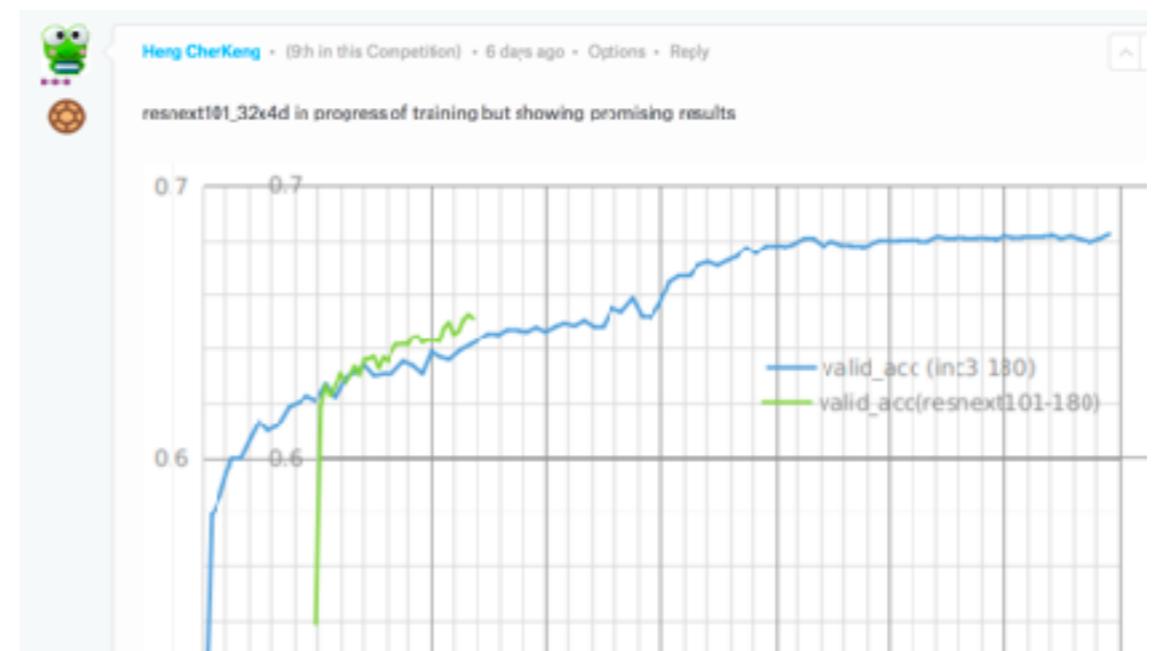
1. 画像の分類タスクのコンテストでよく使われる基本的な技術,
2. 画像の Segmentation タスクのコンテストで効果的であった
トリッキーな方法の具体例を紹介

最近のデータ分析コンテストの傾向

- ・ コンテスト上位に深層学習を使った解法が増えている。
- ・ ただし単に SoTA を使えば勝てるという話にはならない。基本的なことはすべてできることを前提として、データ/タスク/評価指標に適合する手法や特別なアイディアを選択することが決め手になることが多い。



▲ 霧のかかった衛星画像が含まれるデータセットに
対し霧を除去する手法 [He+ '09] で正規化して学習



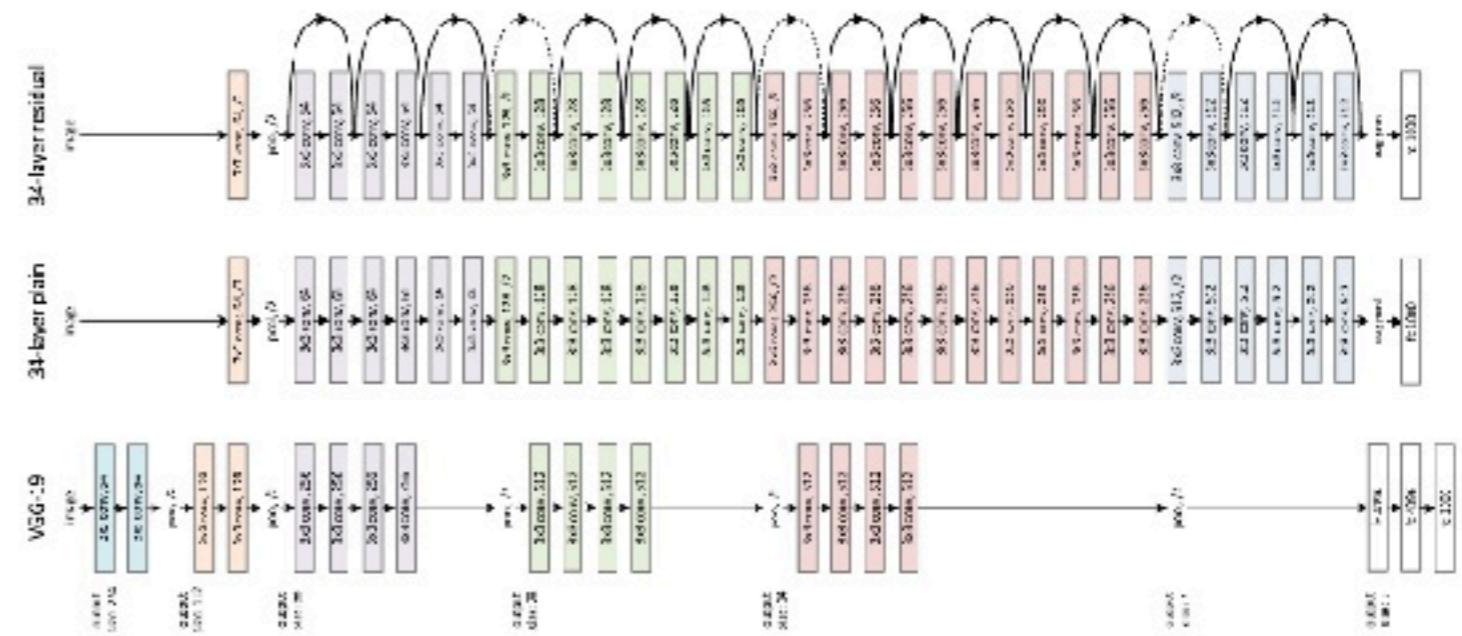
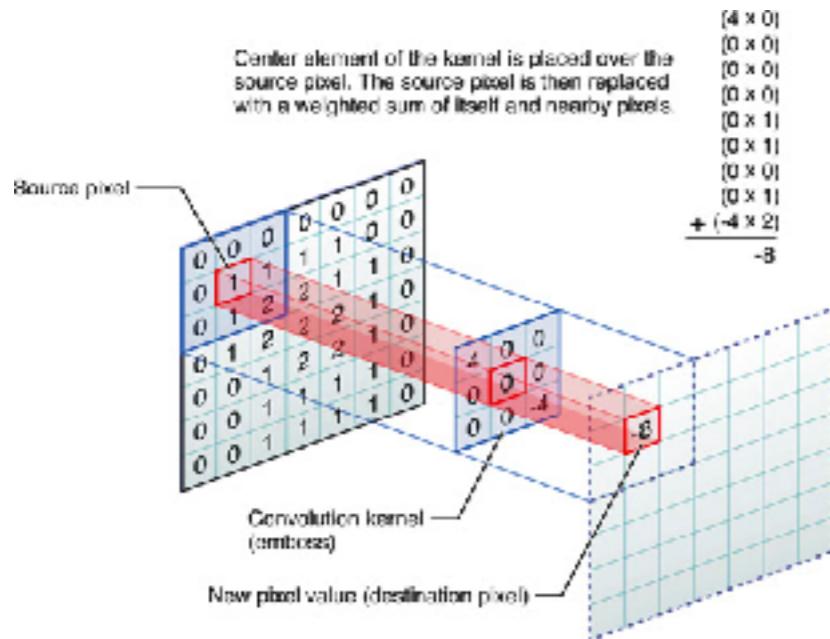
▲ Squeeze-and-Excitation [Hu+ '17] を応用した
SE-ResNeXT での実験結果や実装の共有

画像の分類タスク (基礎的な技術を紹介)

画像のセグメンテーション (具体例を紹介)

画像のクラス分類モデル

Convolution を用いた Neural network (NN) モデルが定番の手法。
VGG, ResNet, DenseNet など様々なバリエーションが提案されて
いる。



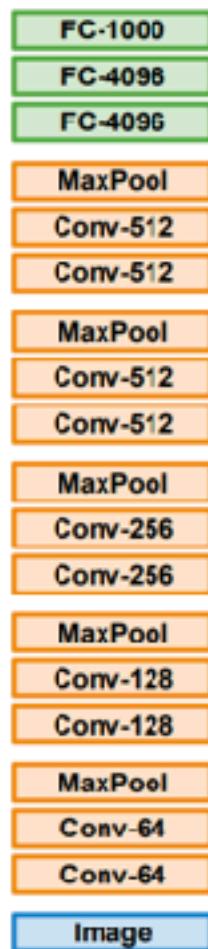
大きな NN モデルはそれだけ表現力も高く、モデルパラメータは数百万の
レベルで大きくなる。パラメータの数より学習データセットのほうが小さいと
簡単に overfit してしまうことが知られている。

Transfer Learning (fine-tuning)

学習データが小さいなら別の大規模データセットで学習済みモデル (pre-trained model) の特徴パターンを流用する。

解く対象のデータセットが小規模であれば効果的な方法 (※ ただし同様の特徴パターンを有する)

1. Train on Imagenet



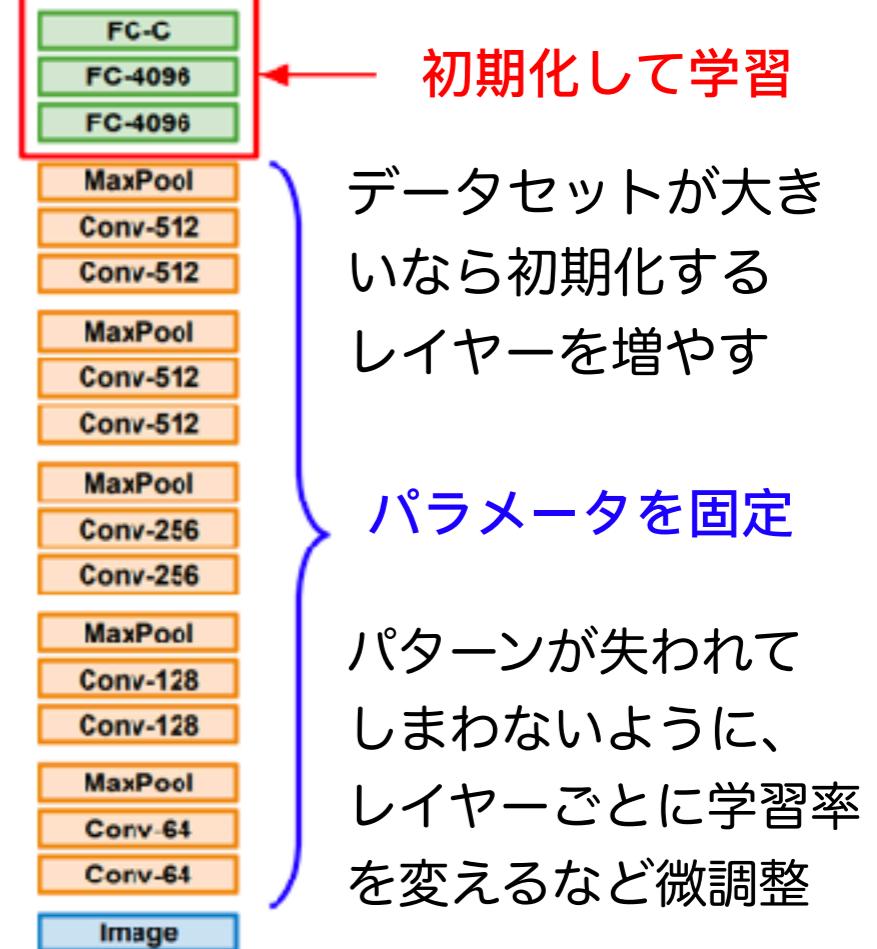
学習済みのモデルは
様々な開発者が公開
している。

2. Small Dataset (C classes)



ここを置き換えて
クラス数を合わせる

3. Bigger dataset



初期化して学習

データセットが大き
いなら初期化する
レイヤーを増やす

パラメータを固定

パターンが失われて
しまわないように、
レイヤーごとに学習率
を変えるなど微調整

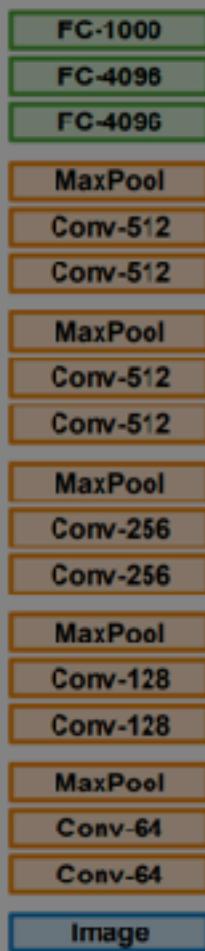
Transfer Learning (fine-tuning)

➡ エッジなど既に獲得できているパターンが失われないように
小さな学習率を設定する

trained model) の特徴パターンを流用する。

解く対象のデータセットが小規模であれば効果的な方法（※ ただし同様の特徴パターンを有する）

1. Train on Imagenet



学習済みのモデルは
様々な開発者が公開
している。

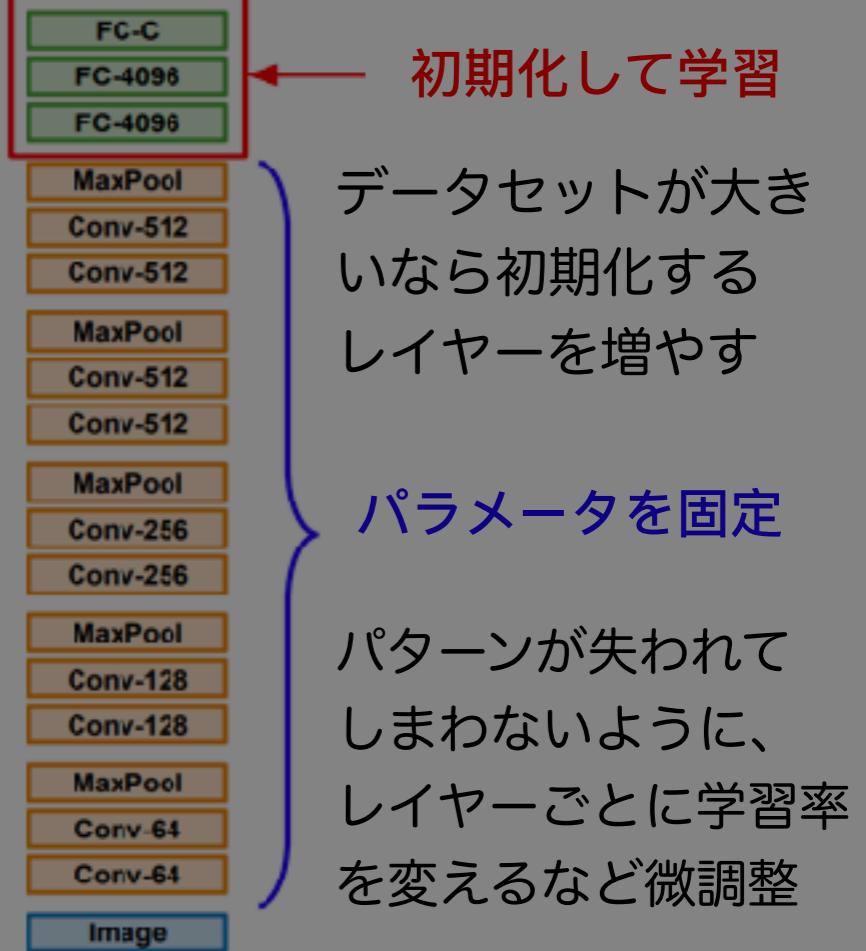
2. Small Dataset (C classes)



ここを置き換えて
クラス数を合わせる

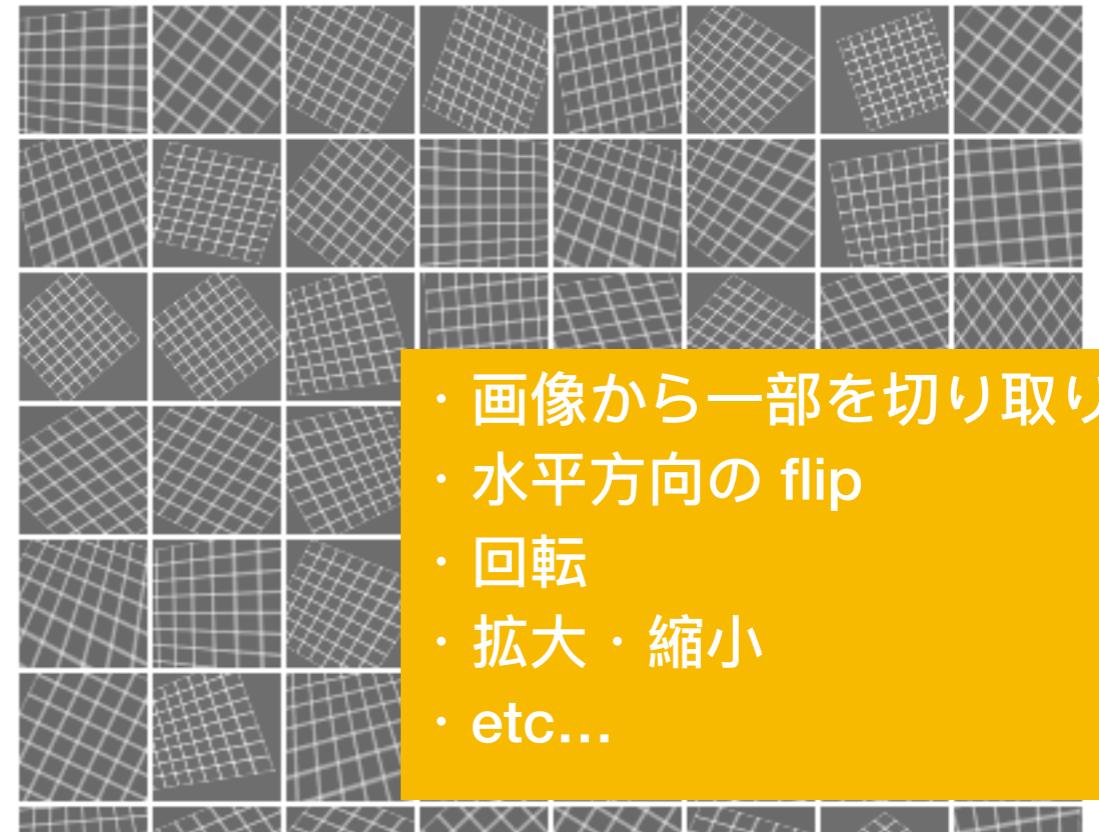
パラメータを固定

3. Bigger dataset



Data Augmentation

Overfitting を避けるための別 の方法として、入力画像に人工的な変形をほどこして訓練データセットを大きくするという方法が一般的に使われている。

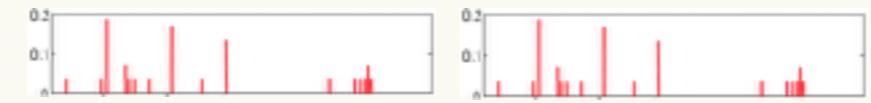


- ・画像から一部を切り取り
- ・水平方向の flip
- ・回転
- ・拡大・縮小
- ・etc...

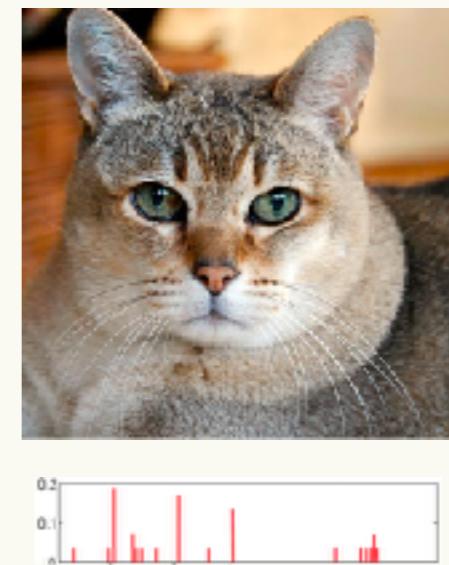
データサイズが膨大になるので、バッチ作成時に CPU でランダムをいれながら変形することが効率的 (Real-time Data Augmentation)。今は特に “Real-time” と書かなくても “Real-time” であることが多い。

Test-Time Augmentation (TTA)

訓練データを拡張するだけでなく、テスト画像も拡張することができる。
テスト事例を拡張して得た予測結果の平均を使う。



テスト画像を増幅する
それぞれの予測結果を計算する



元が同じ画像の予測結果の
平均を答えとする

アンサンブルに近い発想なので、アンサンブルと改善部分を食い合うか
もしれない。ただし経験的にはアンサンブルと両方やったほうが良い。

すーど

Pseudo-Labeling (Self-training)

半教師あり学習の一つ。Kaggle では Pseudo-labeling とよく呼ばれている。「予測したテストデータの一部」を学習データとして追加する。

Algorithm (Pseudo-Labeling):

1. ラベルありデータ (X_l, y_l) から f を学習
2. ラベルなし事例 $x \in X_u$ を f で予測
3. $(x, f(x))$ をラベルありデータに追加。
これは一度に部分集合を追加しても良い。
4. くり返し

やりすぎると「ラベルノイズを増幅してしまう」ため、
High confidence な事例から一部だけを利用するのが一般的な方針。

画像の分類タスク

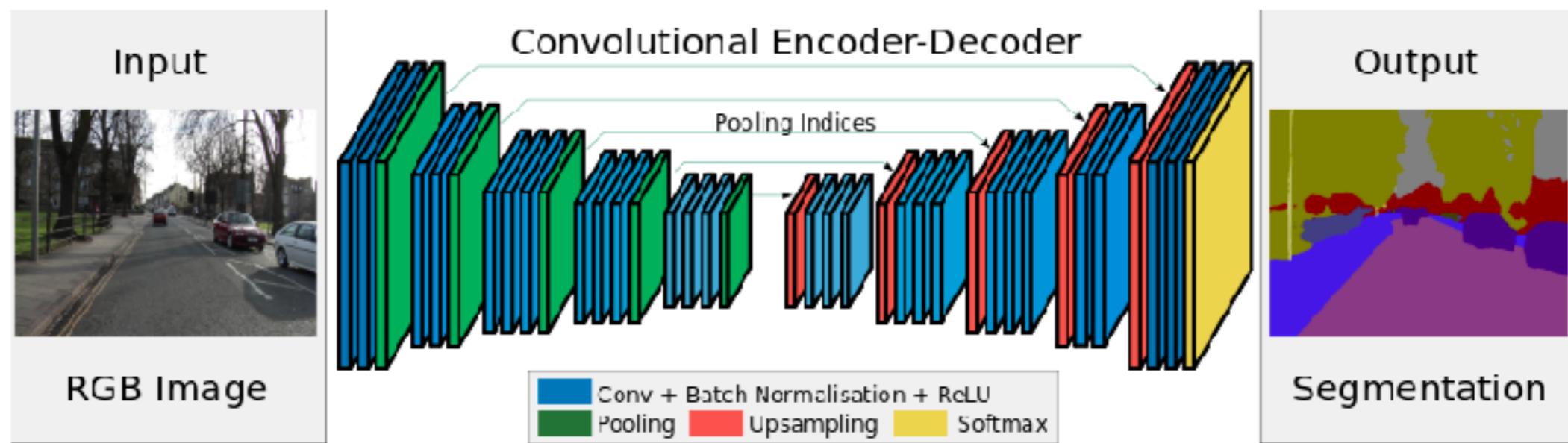
(基礎的な技術を紹介)

画像のセグメンテーション

(具体例を紹介)

Semantic Segmentation モデル

画像からピクセル単位での分類を行うモデル。Fully-Connected Layer を持たない Fully-Convolutional Network (FCN) でモデルを構成することが一般的。



定番として SegNet [Badrinarayanan+ '15]、U-Net [Ronneberger+ '15] がある。両者は by-pass の方法が異なるが、ほぼ同じもの。セグメンテーションは少量のラベルデータでも End-to-end な FCN によって優れた結果が得られることが知られている。

Segmentation を道具として使う

NOAA Right Whale Classification (※ 2年前の SegNet 論文以前)
→ 447クラス分類。クジラ頭部検出 CNN → クジラ分類 CNN



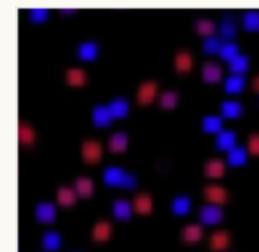
NOAA SeaLion Counting (2017)

トドの数え上げ。

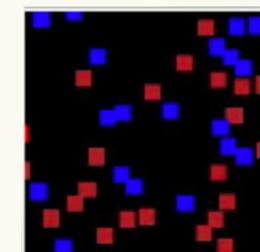
トド検出 FCN (U-Net) →
数え上げ Ridge Regression



Input



Prediction

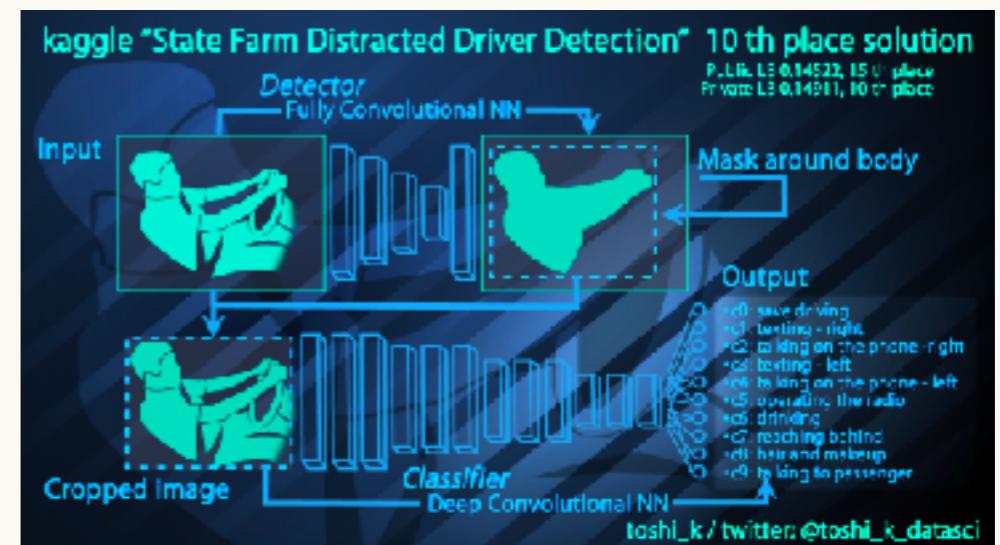


Target

StateFarm Distracted Driver Detection (2016)

多クラス分類。

運転手検出 FCN (SegNet) → 危険運転分類 CNN



(Crazy) Data Augmentation

あらゆる試行錯誤で効果検証を行い、最高のモデルを追求する。時にはクレイジーなことも試す。

((アイディアとして面白いので紹介))



標札を自動で貼り付けて学習データを拡張



標札・自転車を追加して拡張する



SpaceNet Challenge #2

背景：衛星画像から地図作成（建物の検出）の自動化を行う

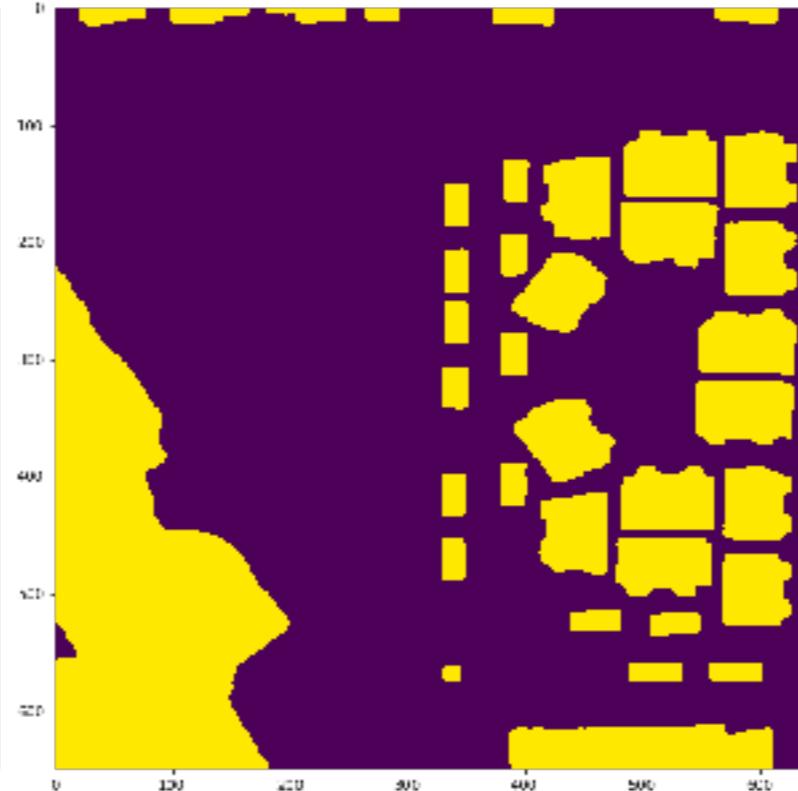
タスク：**Semantic Instance Segmentation**

評価指標：建物ポリゴンの被覆率50%以上を TP とした F-score

入力



出力



建物ごとの評価

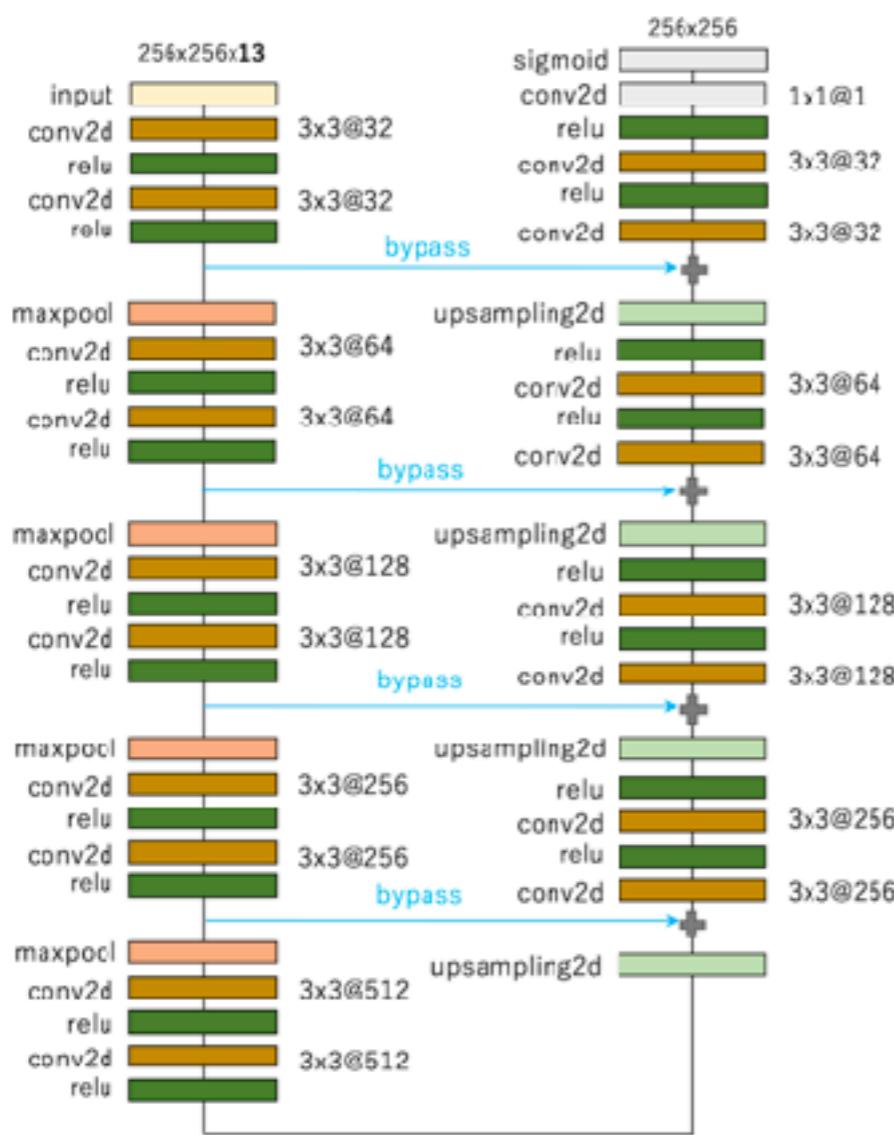
被覆率 50% 以上を TP:
 $(A \cap B) / (A \cup B) > 0.5$

Prediction

Ground truth

「小さな建物」の False-Positive

U-Net で Segmentation して連結成分をインスタンスとした。
データと予測結果をしばらく観察していると、
「小さな建物」の FP によって Precision が極端に悪い。



少しのエラーで被覆率が 0.5 を切るので困難



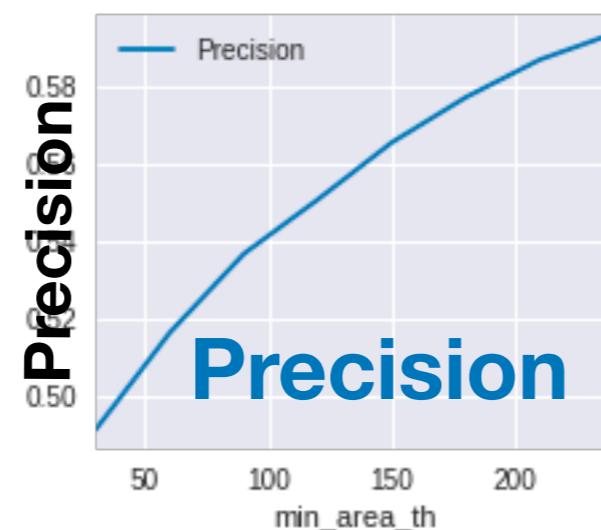
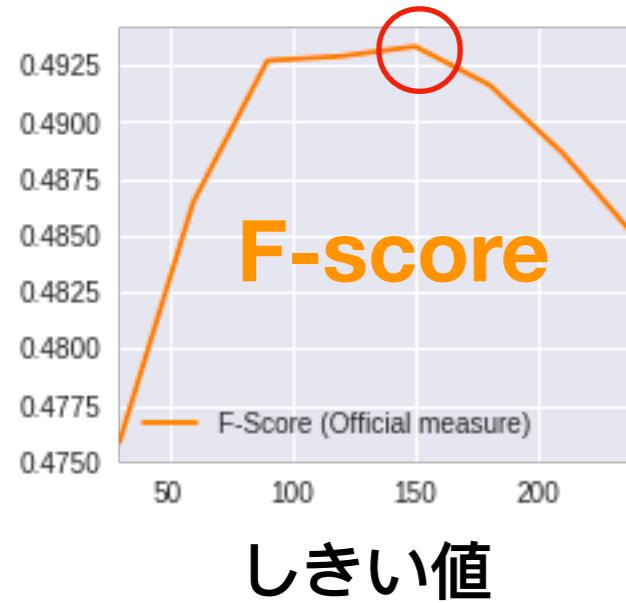
小さいオブジェクトを切り捨てる

「小さいオブジェクトを切り捨てる」ヒューリスティクスを実装。
面積をしきい値として recall-precision のトレードオフを調整する。
閾値は validation set から決定する。

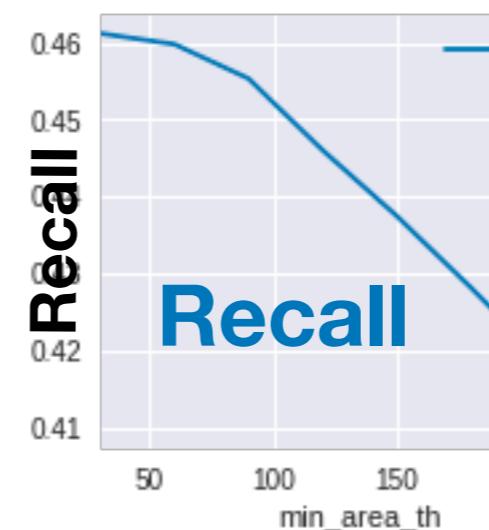
→ F score 0.02 ほどの大きな改善

F-Score を最大とする PR トレードオフを見つける

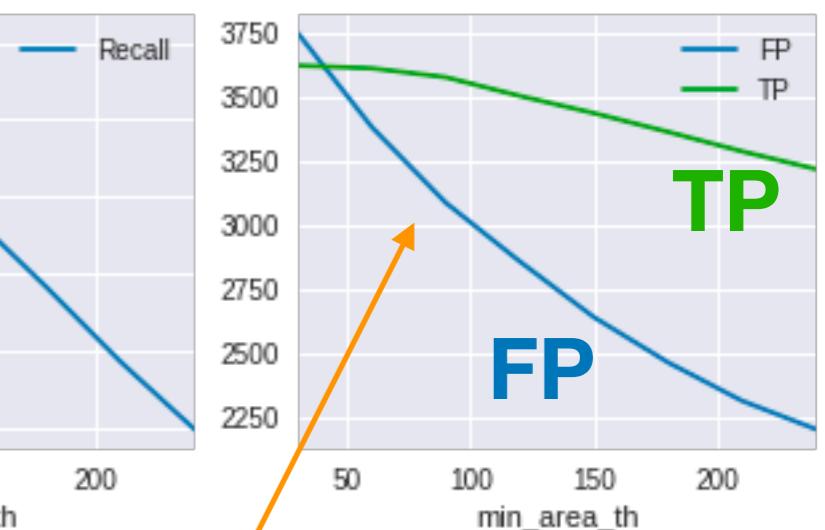
F-スコア



難しい事例がなくなるので
Precision は上がる



TP が減るので
Recall は下がる



しきい値で FP が大きく減る

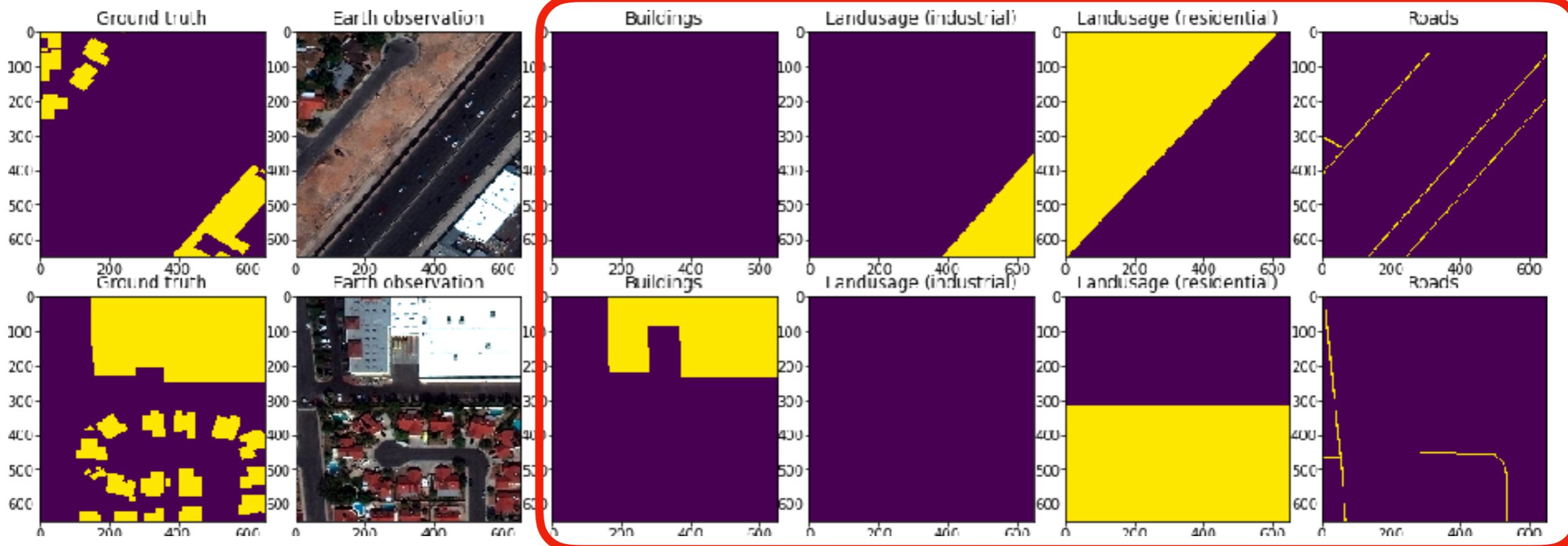
外部リソースを使った改善

OpenStreetMap (OSM) から都市全体のポリゴンを取り出す。

衛星画像 8 channels + OSM 7 channels の合計 15 channels を U-Net の入力とする。建物の形は住宅区画と工業区画で異なるので有用であると期待できる。

→ F score 0.02 ほどの大きな改善

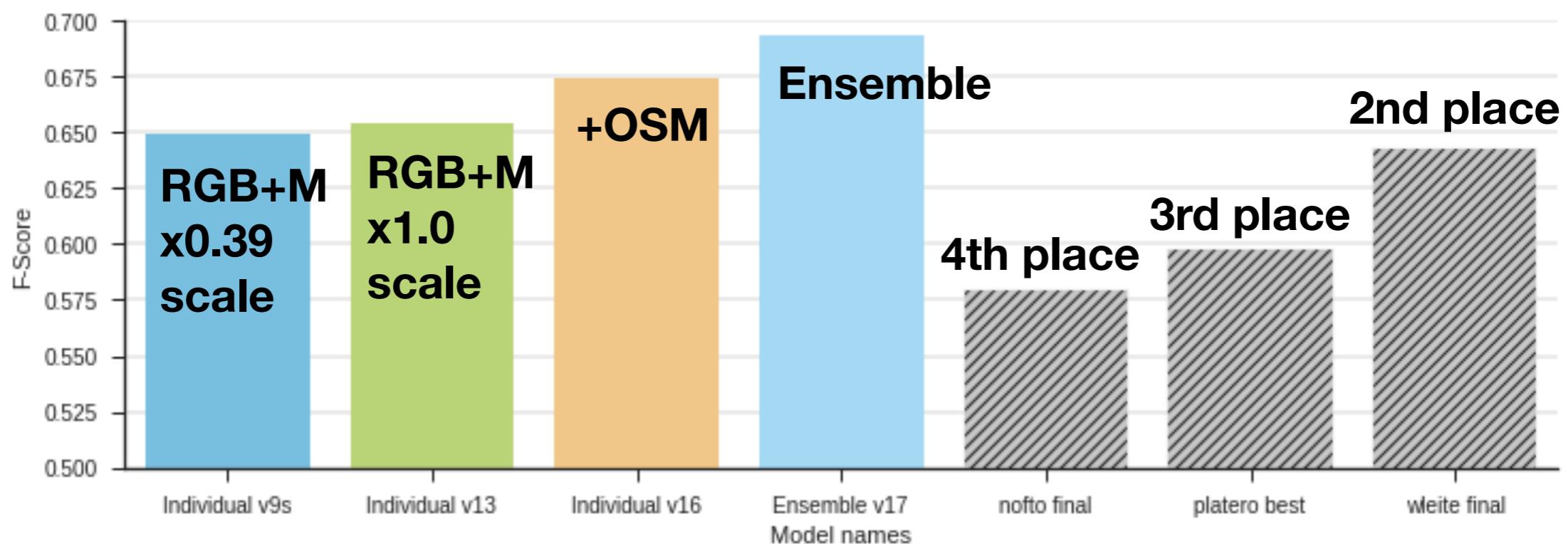
このチャンネルを U-Net の入力に追加



アンサンブル

セグメンテーションモデルは非常にパラメータが不安定なので、経験的にもアンサンブルや Bagging が効果的であることが多い。

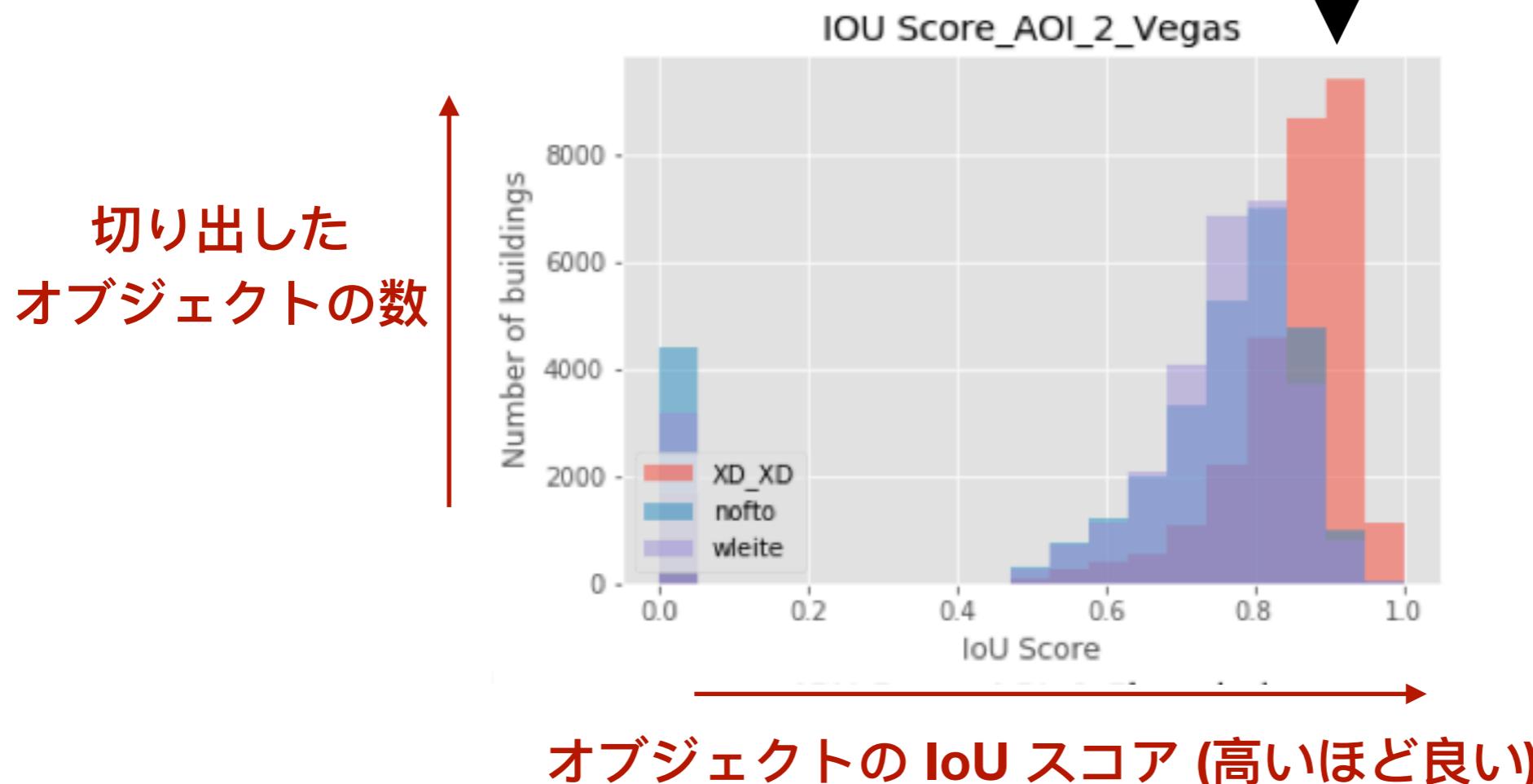
小さいオブジェクトに対する閾値による改善が大きく、
単体のモデルで 1位相当のスコア。
加えて OSM とアンサンブルで 2 位との差を 0.05 まで広げた。



結果分析

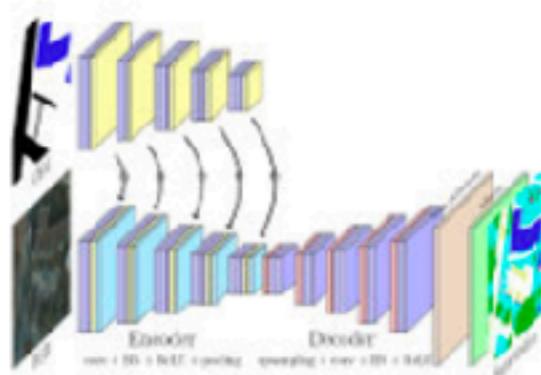
深層学習 (1位) vs エッジ検出ベースのアルゴリズム (2位, 3位)
IoU スコアの分布を比較。

IoU が全体的に 10% ほど高い。個々の Instance の形を正確に切り出せている。



関連研究

コンテスト終了直前に ONERA (フランス国立航空宇宙研究所) の研究員から同じアイディアを元にした論文が出た。FuseNet を使うので私の方法よりもモデルパラメータが少なく学習効率が良さそう。

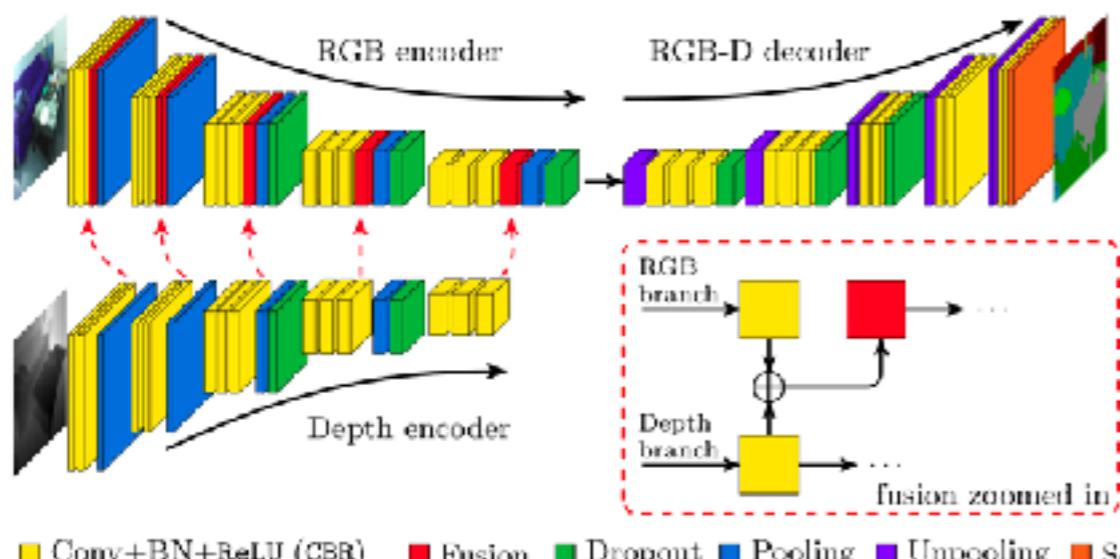


Cartography and especially crowd-sourced geographic information like [OpenStreetMap](#) is a great way to drive a neural network towards a correct classification. With Nicolas Audebert and Sébastien Lefèvre, we built fusion networks able handle efficiently this new input.

The SpaceNet Challenge round 2 winner is using a similar solution: see his [blog post](#) which mentions our paper. OSM as input is promising !

[CVPR'17 paper / arxiv]

出典: <http://www.onera.fr/en/staff/bertrand-le-saux>



FuseNet (Hazirbas+, '16) は element-wise summation で 実装される Fusion Layer を Pooling の前に挿入して 複数の branch における feature map を合成する方法。

最後のスライド：コメント

今日の発表では、
「コンテストと Leaderboard のメカニズム」、
コンテストで広く使われる「メタ学習」「深層学習」の紹介をした。

データ分析コンテストには課題もあるが、
主催サイドも悪いところは改善し進化し続けている。

コンテストは様々なアルゴリズムや最新の研究の reproducibility を
調べ、学ぶためのとても良い舞台。是非参加して楽しんでほしい。

付録：参考文献 (1/2)

1. [Blum & Hardt '15] "The Ladder: A Reliable Leaderboard for Machine Learning Competitions", In Proc. of the ICML '15. <https://arxiv.org/abs/1502.04585>
2. [Hardt '17] "Climbing a shaky ladder: Better adaptive risk estimation", <https://arxiv.org/abs/1706.02733>
3. [Töscher & Jahrer '09] "The BigChaos Solution to the Netflix Grand Prize", http://www.netflixprize.com/assets/GrandPrize2009_BPC_BigChaos.pdf
4. [Wang & Yang '17] "Diabetic Retinopathy Detection via Deep Convolutional Networks for Discriminative Localization and Visual Explanation", <https://arxiv.org/abs/1703.10757>,
5. [Audebert+ '17] "Joint Learning from Earth Observation and OpenStreetMap Data to Get Faster Better Semantic Maps" In Proc. of EARTHVISION 2017 IEEE/ISPRS CVPR Workshop, <https://arxiv.org/abs/1705.06057>
6. [Hazirbas+ '16] "FuseNet: incorporating depth into semantic segmentation via fusion-based CNN architecture", In Proc of Asian Conference on Computer Vision 2016.

付録：参考文献 (2/2)

7. [Hu+ '17] “Squeeze-and-Excitation Networks”, In Proc. of the CVPR '17.
<https://arxiv.org/abs/1709.01507>
8. [He+ '09] “Single Image Haze Removal”, In Proc. of the CVPR '09.
<http://kaiminghe.com/cvpr09/>