GMO INTERNET GROUP

GMOインターンシップ

CTR予測

GMO INTERNET GROUP

STOCK CODE 9449

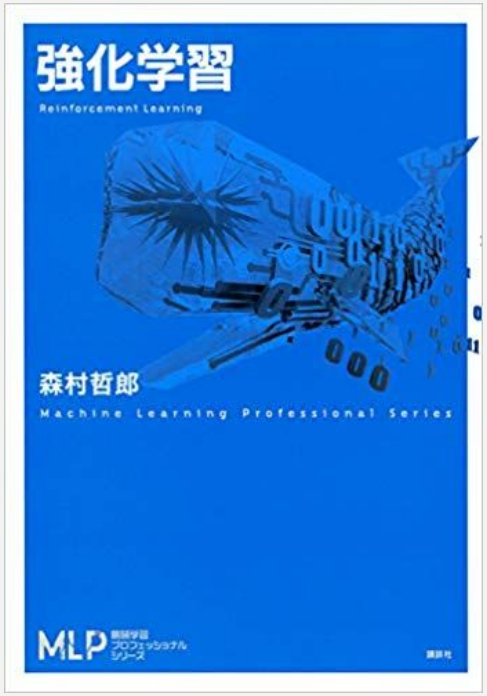
すべての人にインターネット

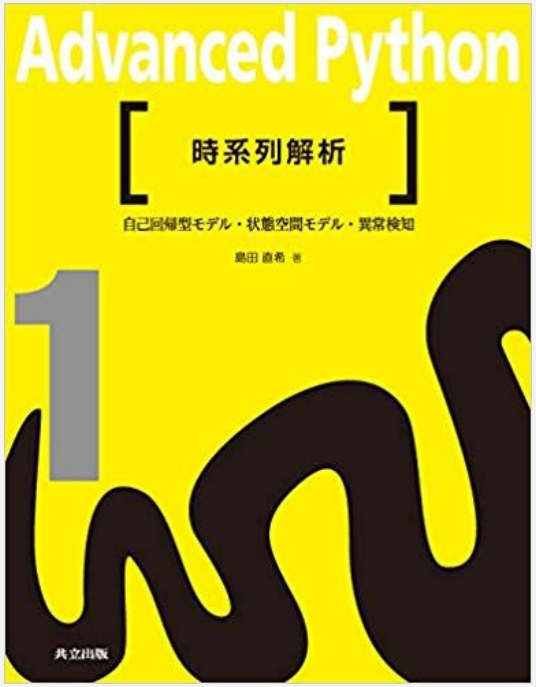
http://www.gmo.jp

目次

自己紹介

名前:長谷川太河
所属:東京大学経済学部3年
最近読んだ本:





目的・目標

目的・目標

「広告のクリック率(CTR)予測」

1. 課題に即した評価指標で最適なモデルの策定
2. 推論時間を短くしたい
3. feature importanceの可視化
4. スケールできるようなモデル

データ・特徴量の作成

データの特徴

log データセット

・ユーザー*i*と広告*j*の属性
・広告*j*に対するユーザー*i*の行動

変数名	説明
広告の識別ID	campaign idやsponsor idなど広告の種類を識別
ssp-uid	ユーザーの識別ID
os, browser, carrier name, region	ユーザーの属性
fq, inview fq	ユーザーが広告を表示・視認した回数
recency, inview recency	前回の広告表示・視認からの時間
request time, click time, cv time	リクエスト、クリック、cvの時間
click_flg, cv_flg, is_rt	クリック、cvしたかどうか、rtタグを踏んだユーザーかどうか

今回の目的変数

データの特徴

rt データセット　　ユーザー*i* の今までのrtに関する時系列データ

変数名	説明
広告の識別ID	campaign idやsponsor idなど広告の種類を識別
ssp-uid	ユーザーの識別ID
os, browser, carrier name, region	ユーザーの属性
rt_status	有効なrtかどうか
request time	リクエストされた時間

データの特徴

1. **不均衡データ**
   全体のCTRは10%
   （総クリック数20,000、総データ数200,000）
   →**Over Sampling, Under Sampling, Probability Calibration, Focal Loss, Affinity Loss**

2. **ほぼ1ユーザー1データ**
   （総ユーザー数194,525、総データ数200,000）
   →**ユーザーベースの手法よりコンテンツベースの手法**
   ssp uidをone hot encodingするのはあまりよくなさそう

3. **カテゴリ変数が多い**
   →**交互作用、One-hot encoding、Target Encoding**

特徴量作成 (1)

1. **request timeから月・日・時間・曜日を抽出**

*click timeやcv timeに関しては特徴量作成ダメ！！

月　　時間

2019-07-06 21:33:35 UTC

日・曜日

2. **User *i* に対する広告 *j* の累計rt回数（rt_times）**

ssp uid	sponsor_id	is_rt
i	j	1
i	j	1
i	k	1
i	j	1

User *i* に対する広告 *j* の累計rt回数
rt_times = 3

特徴量作成 (2)

3 Target Mean Encoding

変数〇〇ごとのクリック率

（例）slot_idごとのクリック率

```
df.groupby(["slot_id"]).mean()["click_flg"]
```

slot_id	click_flg	Target Encoding
34	1	0.66
25	0	0
25	0	0
34	1	0.66
34	0	0.66

4 Target Smooth Mean Encoding

カテゴリの数が少数の場合、過学習になってしまう恐れがあるので、データ全体での”click_flg”率との加重平均をとる。

5 One Hot Encoding

決定木では不要。回帰では重要。

特徴量作成 (2)

6 欠損値補完

region → -1

recencyとinview recency → 100000

*Nanはないがゼロが存在。初めて広告を見たユーザーの値がゼロに設定されている。

7 log変換

recencyとinview recencyは分散が大きい

max : 1
min : 100000

⇨

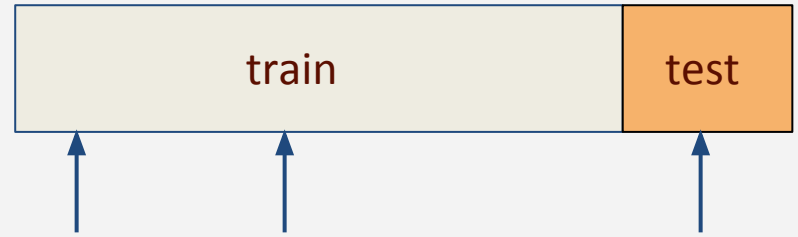
max : 0
min : 11.512

決定木モデルはScaleの影響がないが、回帰とDeepは大きく影響を受ける

データの作成

request timeでソートし8(Train):1(Validation):1(Test)に分割

（理由）時間に依存した変数の存在



rt_timesやTarget Encodingを行う際、未来の結果も利用して特徴量を作成するとleakageになってしまう。

時点Aまでにわかる情報のみを使ってテストデータの特徴量を作成する

モデル紹介

評価指標

1 ROC_AUC

ROC : 偽陽性率(=FP/(FP+TN))を横軸、
真陽性率(=TP/(TP+FN))を縦軸
にしてplotしたもの

ROC_AUC : ROCカーブの下側の面積

2 PR_AUC

PR : Recall(=TP/(TP+FN))を横軸、
Precision(=TP/(TP+FP))を縦軸にして
plotしたもの

PR_AUC : PRカーブの下側の面積

今回用いたモデル

***実際に採用したモデル**

- Logistic Regression
- Random Forest
- Light GBM
- Multi Layer Perceptron
- xDeepFM

***試そうとしたモデル**

- Support Vector Machine **計算量**
- Deep Interest Evolution Network **ユーザーベース**

xDeepFM(1)

1. **低次元・高次元の明示的な交互作用**特徴量を自動的に求めたい

   (例) region × creative id × request_hour

2. 不要な交互作用は使わないようにしたい

   (理由) 精度が下がる可能性あるから

3. **ベクトルワイズ**に交互作用を計算したい

＊　https://arxiv.org/pdf/1803.05170.pdf

xDeepFM(2)

近年、DNNを活用してFMをより表現力をもつようにしようとしている。

	表現力	低次元交互作用	ベクトルワイズ	高次元交互作用が明示的
FM	×	○	○	○
FNN, PNN	○	×	×	×
Wide&Deeep, Deep FM	○	○	×	×
xDeepFM	○	○	○	○

xDeepFMはこれらがすべて可能なモデル

工夫点（1）

1 **Focal Loss**

$$\mathrm{FL}(p_t) = -(1-p_t)^{\gamma}\log(p_t).$$

簡単に分類できるデータの損失を $(1-p_t)^{\gamma}$ により小さくする。

2 **特徴量選択**

Light GBM, Random ForestのFeature ImportanceやValidation Scoreをもとに使用する特徴量を選択

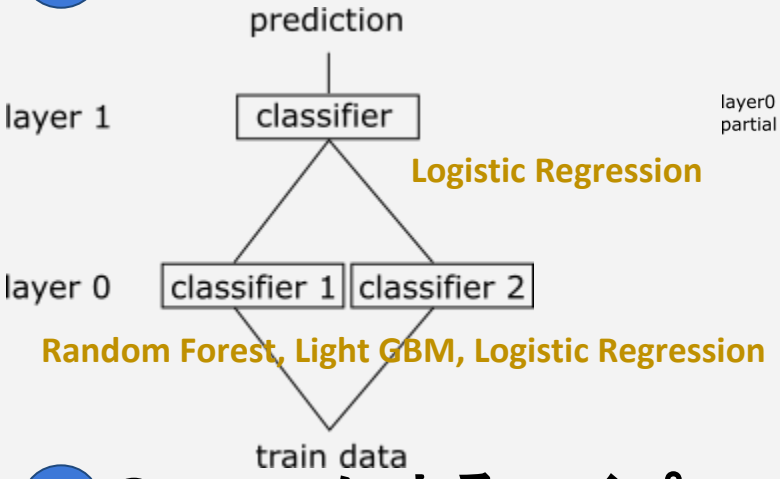
モデルによって使用する特徴量違う

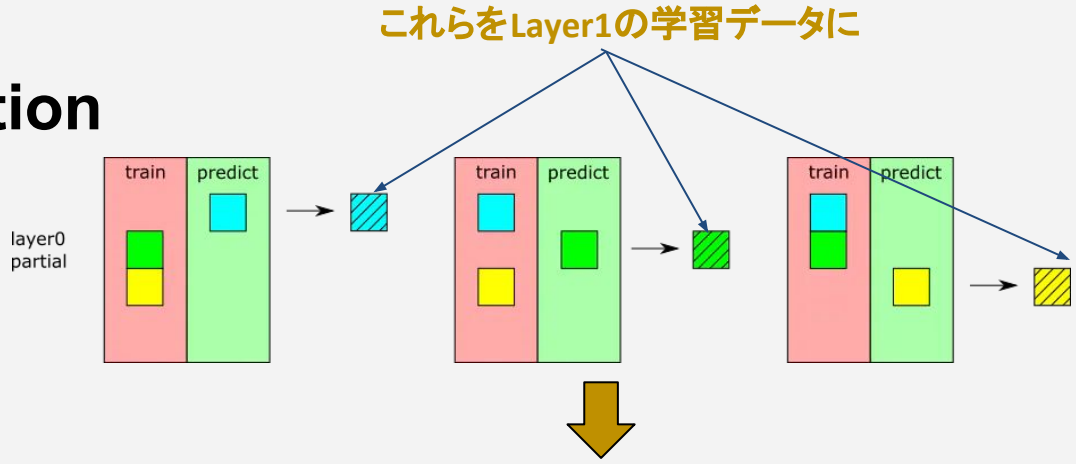
&

rt_timesは使われず

工夫点（2）

3 Stacked Generalization





4 Optunaによるハイパーパラメータの最適化

Light GBMはハイパーパラメータが多いのでoptunaにより最適化。

Random ForestはGrid Search。

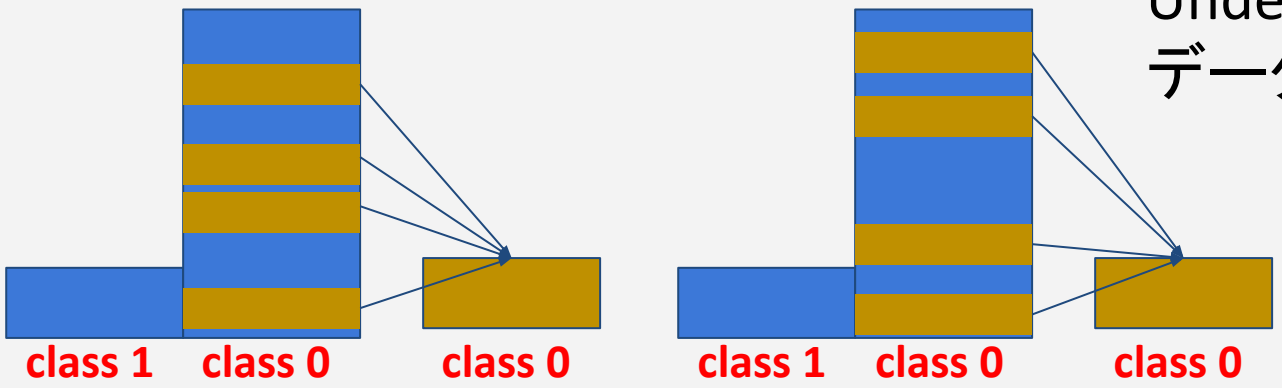
工夫点（2）

5 Under Sampling, Over Sampling

Light GBM : AUC_ROCは上がったが、AUC_PRは下がってしまう。
Logistic Regression : ともに下がる
Probability Calibration : 効果なし

6 Under Sampling + Bagging

毎回違うseedで
UnderSamplingされた
データを用い学習



結果

結果

モデル名	ROC_AUC	PR_AUC
Without features made by me		
Logistic Regression	0.7927	0.3264
Random Forest	0.7444	0.2677
Light GBM	0.7742	0.3003
With features made by me		
Logistic Regression	0.7932	0.3277
Random Forest	0.7646	0.2879
Light GBM (Focal Lossなし)	0.7832	0.3111
Light GBM(Focal Lossあり)	0.7834	0.3141

特徴量エンジニアリングを行うことで**精度UP**!!

Focal Lossで精度少しUP

Logistic Regressionが一番良い結果に

結果

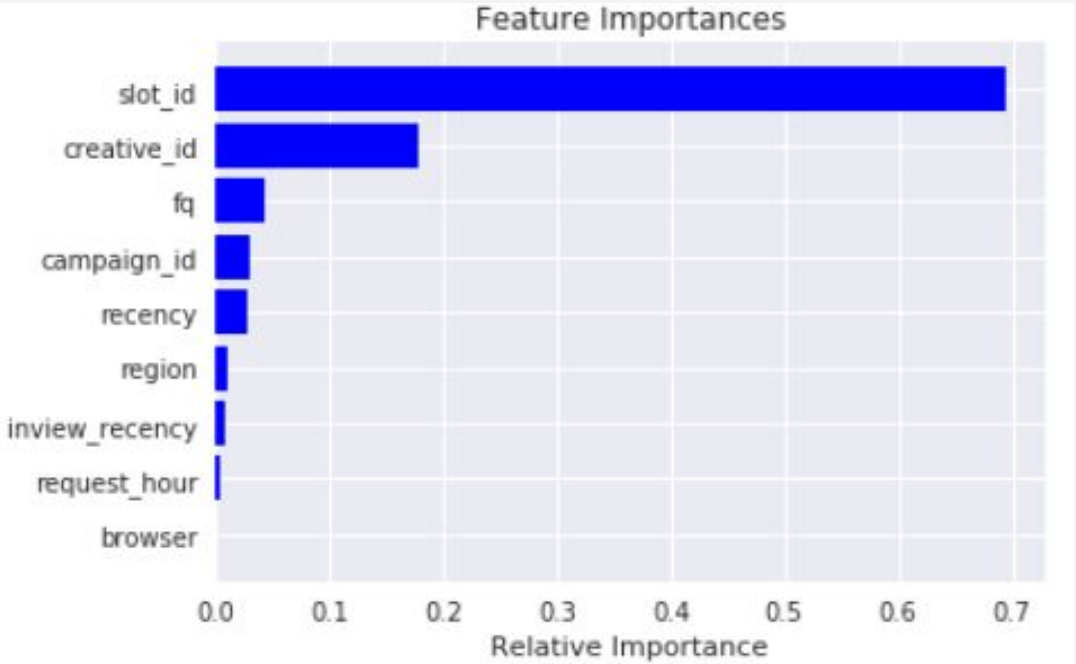
モデル名	ROC_AUC	PR_AUC
Logistic Regression (Undersampling)	0.7873 Down..	0.3159 Down..
Light GBM (Under Sampling)	0.7859 UP!	0.3126 Down..
Light GBM (Under Sampling + Bagging)	0.7848 UP!	0.3175 UP!
Multi Layer Perceptron	0.7476	0.2563
Stacked Generalization	0.7820	03151
xDeepFM	**0.7870**	**0.3103**
Ensemble	**0.7950**	**0.3284**

Best Score

結果

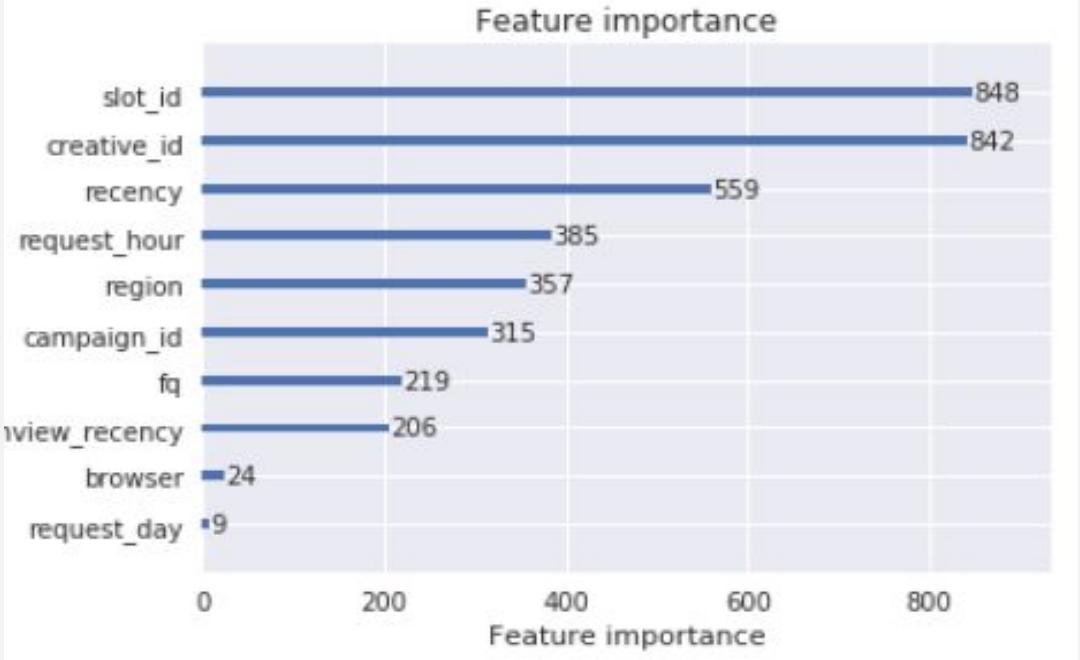
1 Random Forest

slot id, creative id, fq, campaign id, recency の順で重要



2 Light GBM

slot id, creative id, recency, campaign id, request hour の順で重要



結果

予測ラベル

正解ラベル	0	1
0	17740	249
1	1746	265

確実にクリックされたい。

クリックされない広告をクリックされると予測するのは減らしたい（=**Precisionを高めたい**）

Precision : **51.5%**　クリックされると予測したものの半分は実際にクリックされる（Randomだと10%程度）

まとめ

1. **課題に即した評価指標で最適なモデルの策定**
   → ROC_AUC : 0.7950, ROC_PR : 0.3284, Precision 0.515

2. **推論時間を短くしたい**
   → あまり、気にかけることができなかった。。

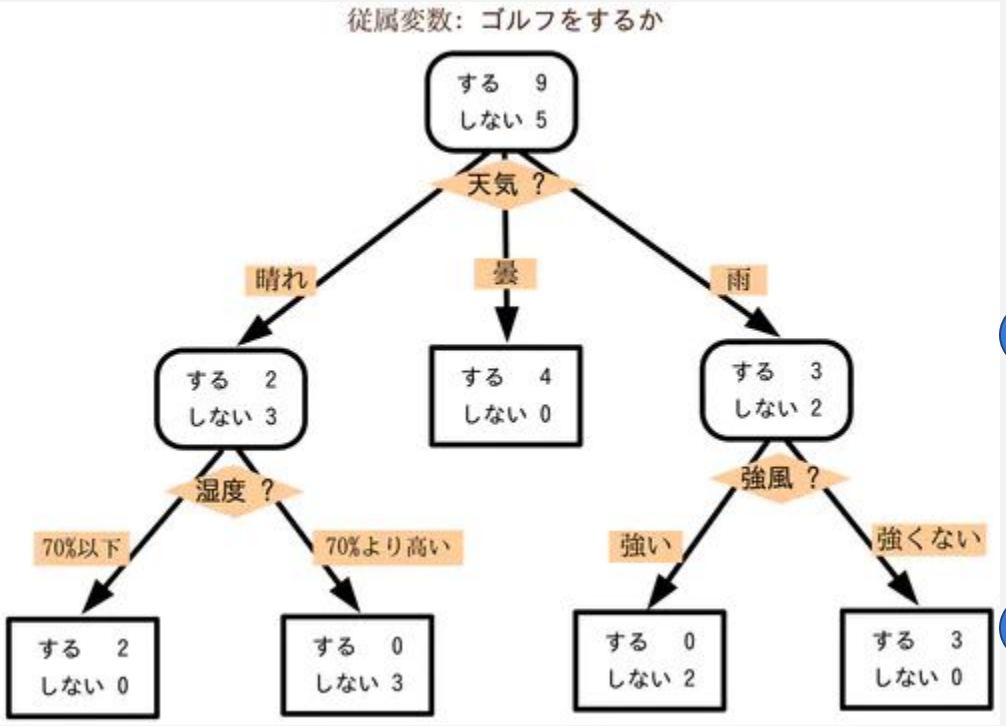
3. **feature importanceの可視化**
   → slot id, creative id, recencyが重要

4. **スケールできるようなモデル**
   → コンテンツベースなので、新しいユーザーが増えてもスケールできる。新しいコンテンツに関してはデータが集まらないと予測がうまくできない。

ご清聴ありがとうございました

Apendix

Random Forest



1. gini impurityで境界とnodeを決定

$$p(i|t) = \frac{n_i}{n}$$

$$I_G(t) = 1 - \sum_{i=1} p(i|t)^2$$

2. bootstrappingでデータをサンプル(replacementありサンプリング)

3. バギング

4. ダミー変数・交互作用作る必要なし

Light GBM(1)

特徴1. Gradient Boosting Decision Tree

2. For $m = 1$ to M:

psedo-residual

(a) For $i = 1, 2, \ldots, N$ compute

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$

(b) Fit a regression tree to the targets r_{im} giving terminal regions R_{jm}, $j = 1, 2, \ldots, J_m$.

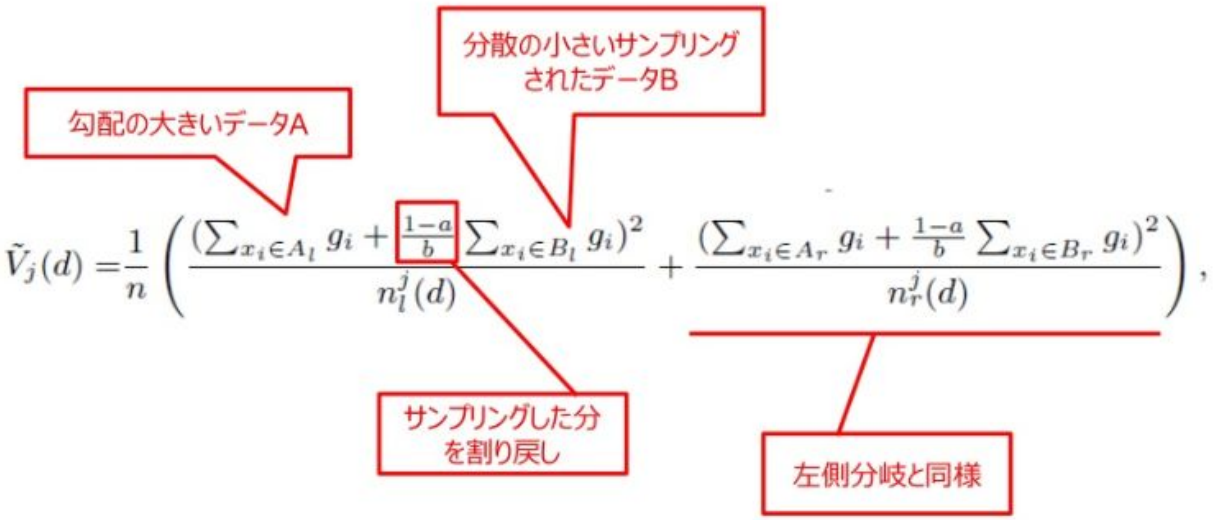
(c) For $j = 1, 2, \ldots, J_m$ compute

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$

(d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.

Light GBM(2)

特徴2. ヒストグラムをもとに分岐点を計算
特徴3. Gradient-based One-side Samplingでデータ数を減らす



- **勾配の絶対値の上位$a \times 100$ %と、残りのデータの$b \times 100\%$をサンプリングして各反復で使用**
  サンプリングした側の勾配は$\frac{1-a}{b}$倍して使用

Light GBM(3)

特徴4. Exclusive Feature Bundlingで特徴量を減らす

Sparseなデータでは非ゼロ要素に被りがないときが多い。そのような特徴同士はまとめて(bundling)、一つの特徴としてしまう。
計算量がO(#data*#feature)→O(#data*#bundle)になる。

xDeepFM(1)

- One hot encoding

↓

- Embedding Layer

$$e=[e_1, e_2, \cdot, \cdot, e_m]$$

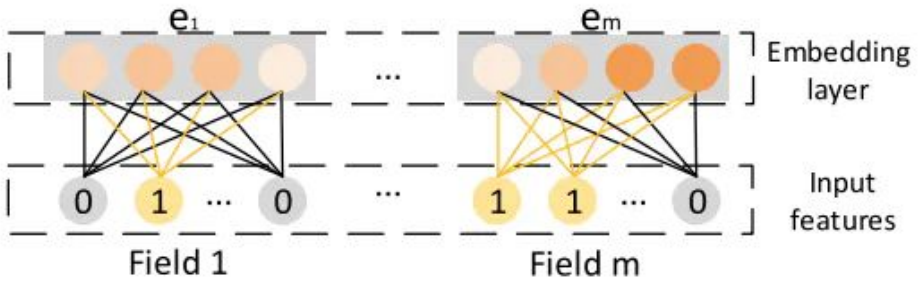


Figure 1: The field embedding layer. The dimension of embedding in this example is 4.

1 暗示的な高次元の交互作用

$$\mathbf{x}^1 = \sigma(\mathbf{W}^{(1)}\mathbf{e} + \mathbf{b}^1)$$

$$\mathbf{x}^k = \sigma(\mathbf{W}^{(k)}\mathbf{x}^{(k-1)} + \mathbf{b}^k)$$

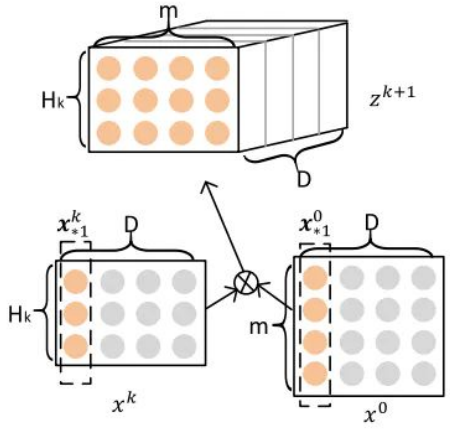
xDeepFM(2)

2 明示的な交互作用

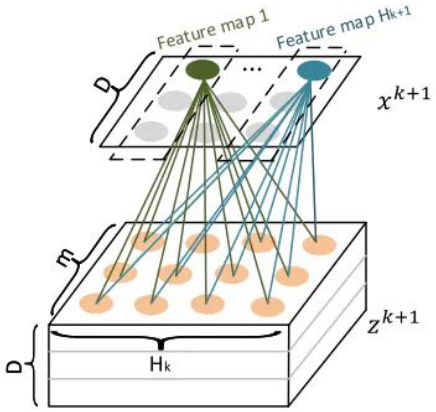
Compressed Interaction Network

明示的な交互作用

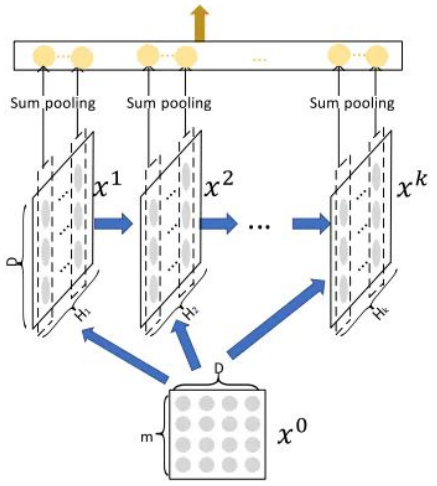
$$\mathbf{X}_{h,*}^{k} = \sum_{i=1}^{H_{k-1}} \sum_{j=1}^{m} \mathbf{W}_{ij}^{k,h} (\mathbf{X}_{i,*}^{k-1} \circ \mathbf{X}_{j,*}^{0}) \qquad (6)$$



(a) Outer products along each dimension for feature interactions. The tensor Z^{k+1} is an intermediate result for further learning.



(b) The k-th layer of CIN. It compresses the intermediate tensor Z^{k+1} to H_{k+1} embedding vectors (aslo known as *feature maps*).



(c) An overview of the CIN architecture.

xDeepFM(2)

3 Networkを組み合わせる

$$\hat{y} = \sigma(\mathbf{w}_{linear}^T \mathbf{a} + \mathbf{w}_{dnn}^T \mathbf{x}_{dnn}^k + \mathbf{w}_{cin}^T \mathbf{p}^+ + b) \quad (15)$$

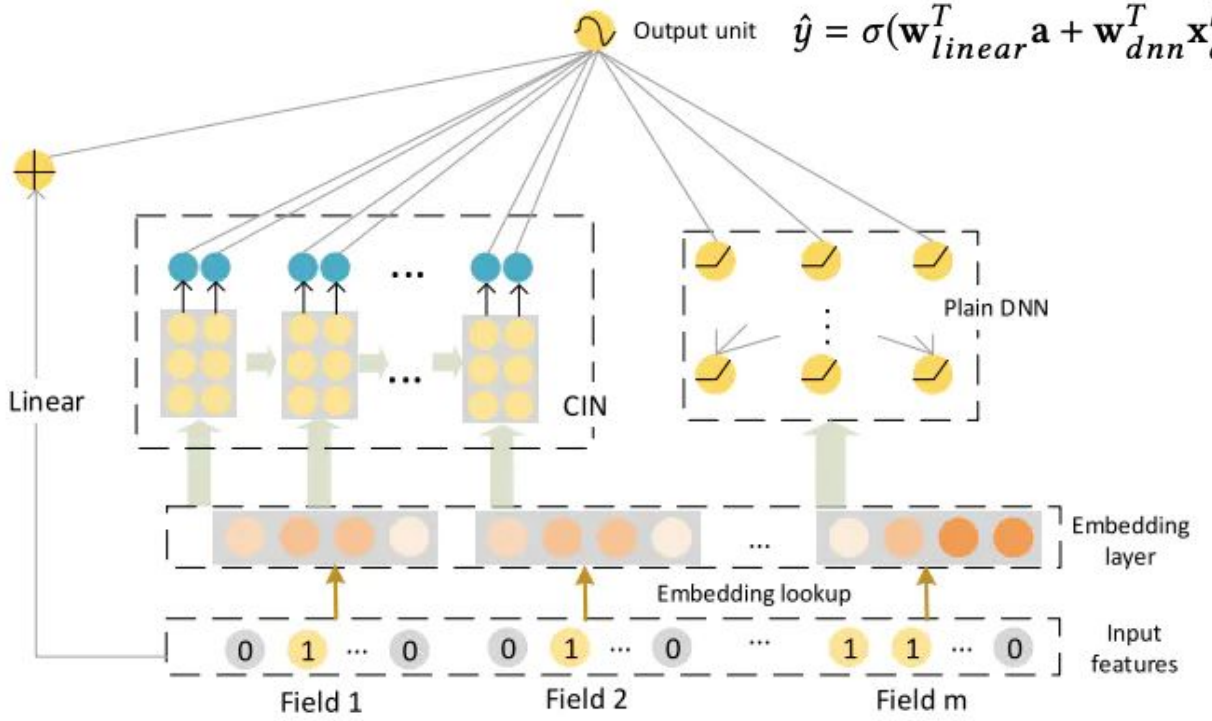


Figure 5: The architecture of xDeepFM.