Kaoru Honda / 2020.09.03

PythonとKerasによるディープラーニング

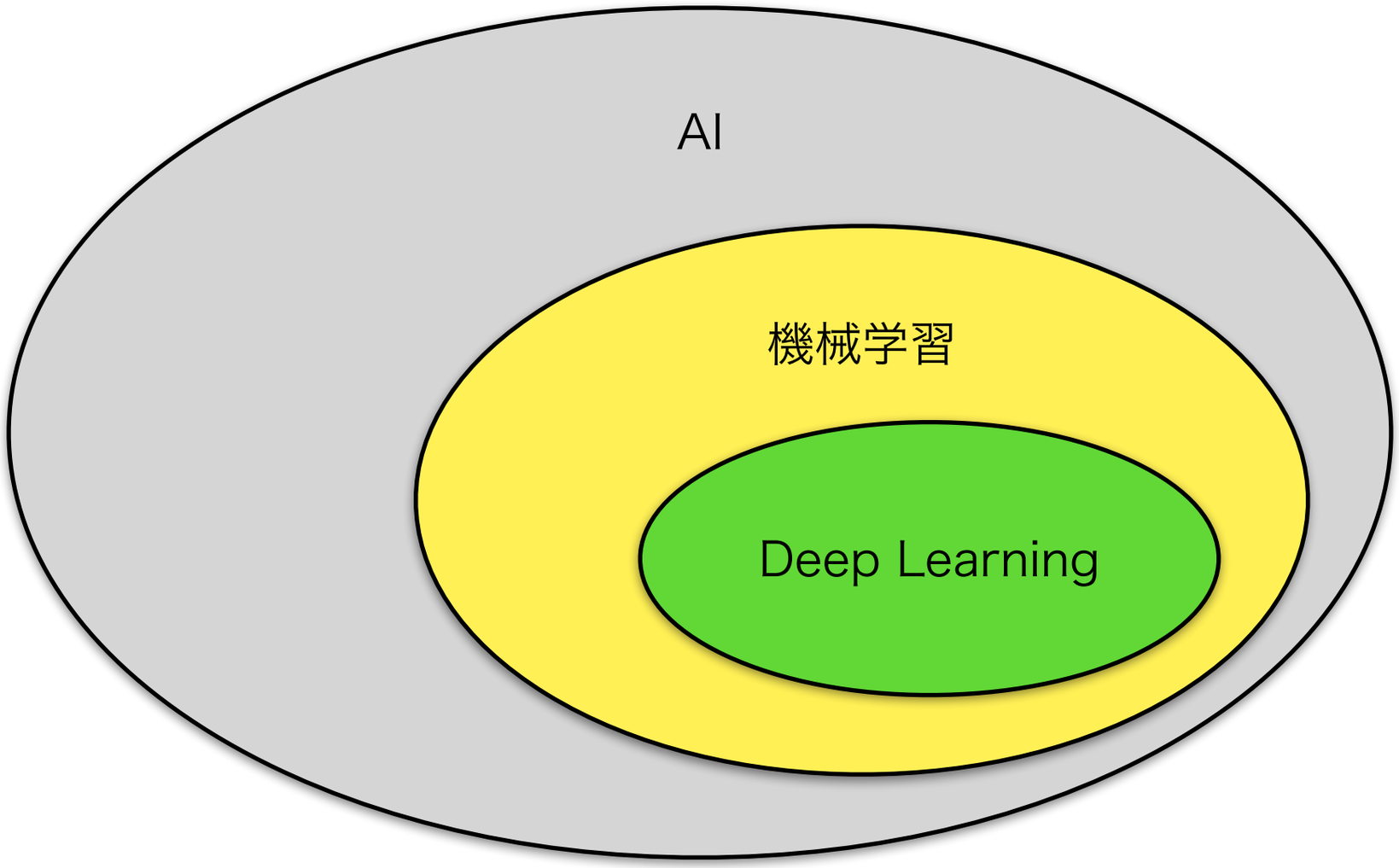
学習用の図版・資料集

PythonとKerasによるディープラーニング

原著名：Deep Learning with Python

https://book.mynavi.jp/ec/products/detail/id=90124

この資料は、「PythonとKerasによるディープラーニング」（以下底本）をベースに、理解を進めるために作成・引用した図版集である。断りがないかぎり、オリジナルのアイデア・著作権は原著者・翻訳者にある。

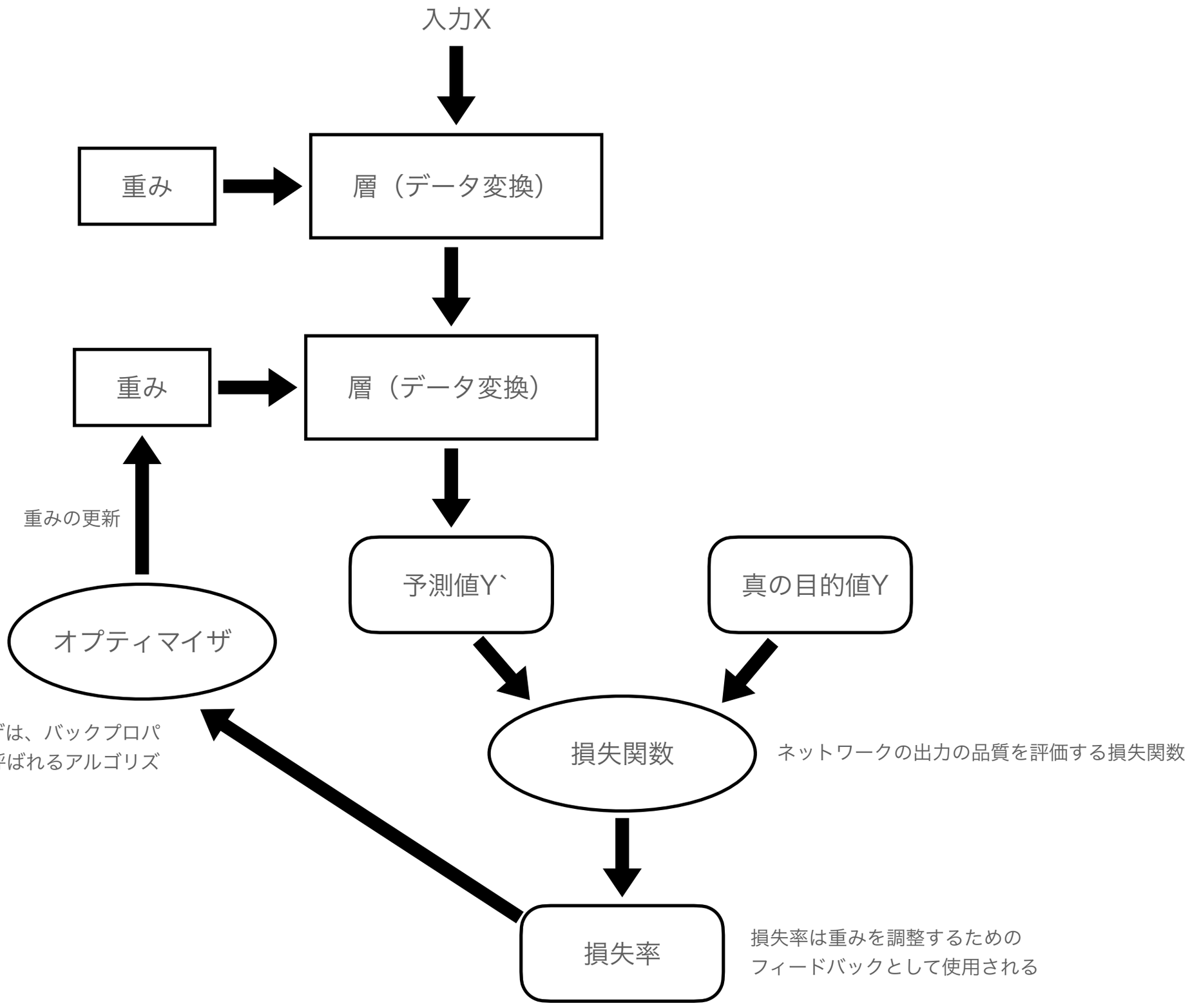


AI, 機械学習, Deep Learning

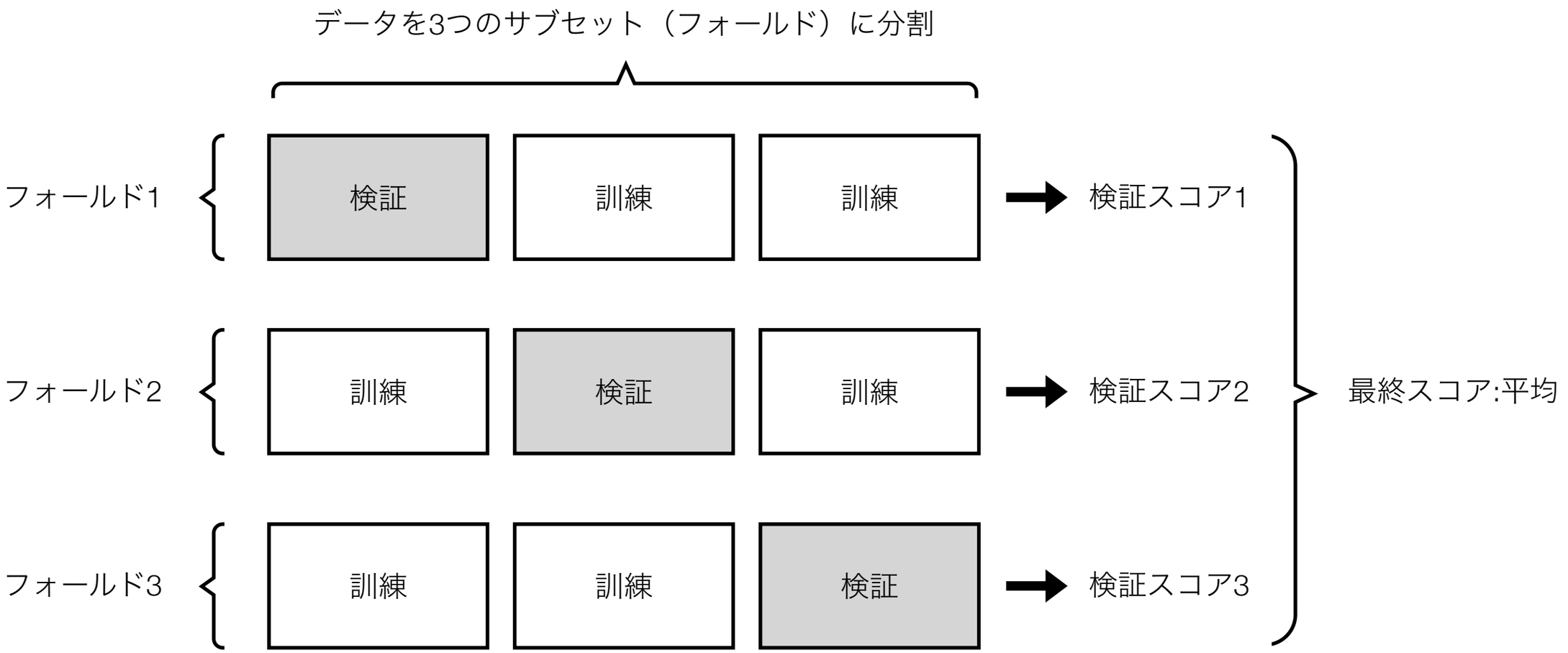
機械学習（Machine Learning）

新しいプログラミングパラダイム - 機械学習

Deep Learning の仕組み
ネットワーク、層、損失関数、オプティマイザの関係



k分割交差検証（3分割交差検証の例）



ニューラルネットワークのデータ前処理

データ前処理の目的は、生のデータをニューラルネットワークにより適したものにすることです。これには、ベクトル化、正規化、欠損値の処理、特徴抽出が含まれる。

• ベクトル化

ニューラルネットワークの入力値と目的地はすべて、浮動小数点データのテンソルでなければならない。
この手順を「データのベクトル化」(data vectorization)と呼びます。
例えば、整数のリストとして表されたテキストを、one-hotエンコーディングを使ってfloat型のテンソルに変換しました。

• 値の正規化

各特徴量を個別に正規化し、標準偏差が１、平均が0になるようにする必要がある。

NumPy配列を利用する。

xは形状が()の2次元配列

```
x -= x.mean(axis=0
x /= x.std(axis=0)
```

• 欠損値の処理

一般にニューラルネットワークでは、欠損値は0にするのが安全である。

過学習を防ぐ方法

- 訓練データを増やす
- ネットワークのキャパシティを減らす
- 重みを正則化する
- ドロップアウトを追加する

機械学習の一般的フロー

1. 問題を定義し、データセットを作成する
2. 成功の指標を選択する
3. 評価プロトコルを決定する
4. データを準備する
5. ベースラインを超える性能のモデルを開発する
6. 過学習するモデルの開発
7. モデルの正則化とハイパーパラメータのチューニング

モデルの最後の層の活性化関数と損失関数の選択

問題の種類	最後の層の活性化関数	損失関数
二値分類	sigmoid	binary_crossentropy
多クラス単一ラベル分類	softmax	categorical_crossentropy
多クラス多ラベル分類	sigmoid	binary_crossentropy
任意の値に対する回帰	なし	mse
0〜1の値に対する回帰	sigmoid	mseまたはbinary_crossentropy