PythonとKerasによるディープラーニング

4.3 データ前処理、特徴エンジニアリング、表現学習

17T4014Y 伊藤 陽樹

ニューラルネットワークでのデータの前処理

- ベクトル化
  - ✓ニューラルネットワークの入力値と目的値はすべて、浮動小数点データのテンソルでなければならない. (データのベクトル化)
- 値の正規化
  - ✓一般に、比較的大きな値をとるデータや種類の異なる値をとるデータをニューラルネットワークに供給するのは安全ではない.
  - ✓そうしたデータを使用すると、勾配を更新するための値が大きくなり、ネットワークが収束しなくなる.

ニューラルネットワークでのデータの前処理

- 値の正規化
  - ✓データに必要な特性
    - ➢小さな値をとる
      一般に、ほとんどの値は0~1の範囲の値.
    - ➢種類が同じである
      すべての特徴量をほぼ同じ範囲の値.
  - ✓厳格な正規化(場合による)
    - ➢平均が0になるように各特徴量を個別に正規化.
    - ➢標準偏差が1になるように各特徴量を個別に正規化.

ニューラルネットワークでのデータの前処理

- 欠測値の処理
  - ✓データに欠測値(missing value)が含まれることがある.
  - ✓一般に、NNでは、入力データの欠測値は0にするのが安全. (ただし、0が意味のある値としてすでに使用されていないことが前提)
  - ✓欠測値が含まれていないデータでネットワークが訓練されている場合、訓練サンプルで欠測値を人工的に作成する.

特徴エンジニアリング

- 使用しているデータと機械学習アルゴリズム(この場合はNN)に関する知識に基づいて、そのアルゴリズムの性能を向上させるプロセス.
- 特徴エンジニアリングでは、データをモデルに供給する前に、ハードコーディングされた(学習されたものではない)変換を適用する.
- モデルの作業が容易になるような方法でデータを提供する必要がある.

直感的な例

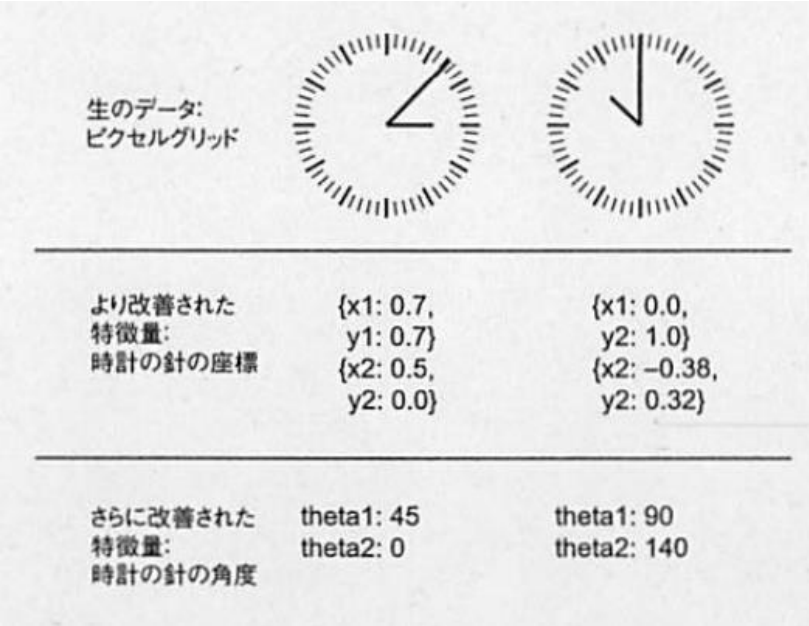

図: 時計の時刻を読み取るための特徴エンジニアリング

- 入力: 時計の画像
- 出力: 時刻
- 時計の針を表す黒いピクセルを追跡し、針の位置を(x, y)座標として出力.
- さらに座標変換
  - ✓針の位置を表す(x, y)座標を、画像の中心を基準とした極座標として実現する.
  - ✓各時計の針の角度θが入力

特徴エンジニアリング

- 特徴エンジニアリングの本質:
  特徴量をより単純な方法で表現することで、問題を容易にする.

- NNは生のデータから有益な特徴量を自動的に抽出できるため、最近のDLでは、殆ど特徴エンジニアリングが不必要.

- とはいっても、良い特徴量があると、
  ✓リソースの消費を抑えた上で、問題をより的確に解決できる.
  ✓問題をずっと少ないデータで解決できる.