平成30年度文部科学省委託
「専修学校による地域産業中核的人材養成事業」

機械学習III

Society5.0実現のためのIT技術者養成モデルカリキュラム開発と実証事業

平成30年度文部科学省委託
「専修学校による地域産業中核的人材養成事業」

機械学習III

Society5.0 実現のための IT 技術者養成モデルカリキュラム開発と実証事業

目 次

第1回：機械学習の基本

講義概要と講義全体の説明

アジェンダ

- 講義全体の説明
- 機械学習とは？
- 機械学習の手法の分類
- 機械学習の手法の選び方
- 機械学習の得意なこと、苦手なこと
- 機械学習プロジェクトの進み方
- 環境構築
- まとめ

講義全体の説明

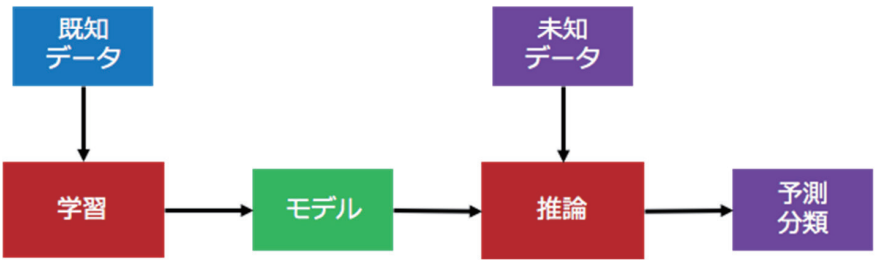
- データ分析において注目を集めている機械学習を網羅的に取り扱う
- 講義の実行環境としては以下の通り
  - プログラミング言語としてはPython
  - プログラミングの実行環境としてJupyter
  - 機械学習のライブラリとしてScikit-learn
  - ディープラーニングのライブラリとしてTensorFlow

全15回の講義について

- 主に機械学習の手法を中心に講義を実施する
  - プログラミング言語としてはPython
  - Pythonの各種ライブラリを利用してデータ分析に必要なスキルの習得を目指す
  - 第2回以降の講義で詳細を取り扱う

機械学習とは？

既知のデータからルールを見つけ出す「学習」と、見つけたルールを用いて未知のデータに適用して結論を得る「推論」から成ります



機械学習とは？

赤い点が猫、青い点が犬を表していて、鼻の大きさと尻尾の長さのデータをプロットしているとします

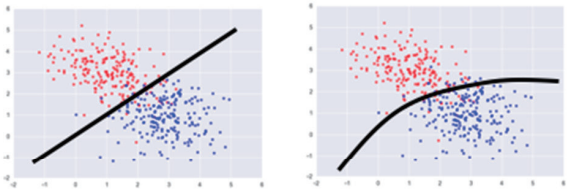
- 学習とは、猫と犬を見分けるためのルールを見つけ出すこと
- 推論とは、初めて見る動物でも、鼻の大きさと尻尾の長さから、猫なのか犬なのか判断すること

機械学習とは？

学習には「アルゴリズムを選ぶ」ことと「パラメータを決める」２つのステップが必要になります

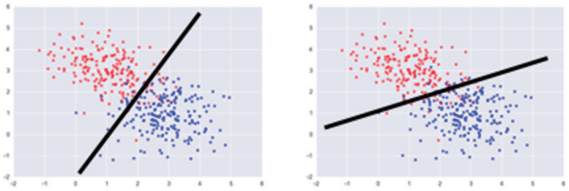
アルゴリズムを選ぶ

どのように判断するかを決める
・直線で分ける？
・曲線で分ける？
・その他の方法で分ける？



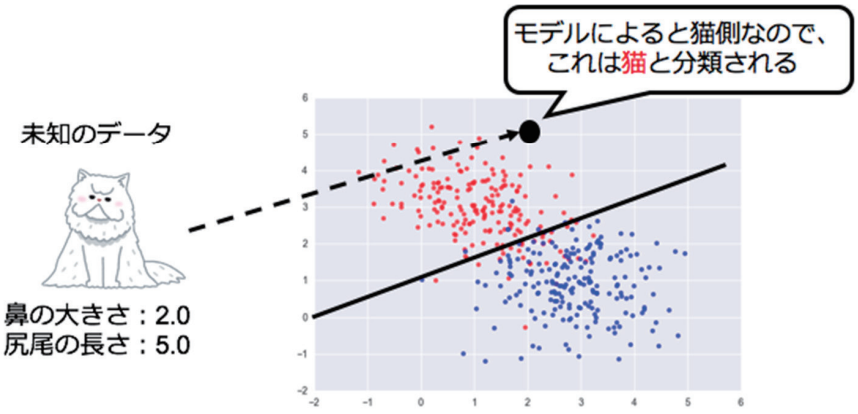
パラメータを決める

直線の場合は傾きを決める
何らかの評価基準に従って
「最も良い」傾きを決めます



機械学習とは？

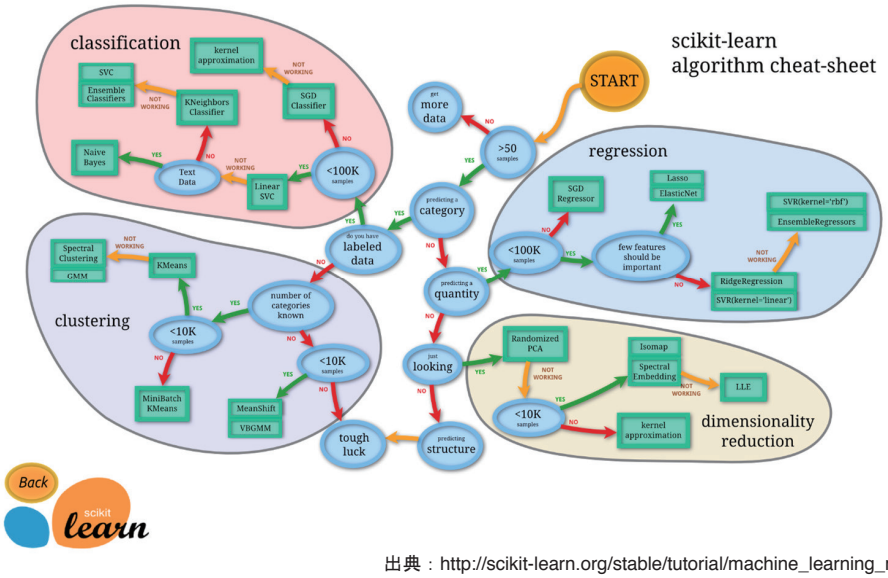
得られた学習結果（モデル）をもとに判断（猫か犬かなので分類）が行われます



機械学習の手法の分類

- 機械学習の手法にはいくつかの分類方法がありますが、代表的なものは以下の3つに分けるものです。
- 教師あり学習
  - 教師データで与えられた正解に近づくように学習し、未知のデータに対しても正しい答えを導き出せるようにする手法
- 教師なし学習
  - 教師データは与えられず、与えられたデータの構造から何らかのルールを導き出せるようにする手法
- 強化学習
  - 試行錯誤を繰り返しながら、より良い結果にたどり着くための方法を獲得していく方法

機械学習の手法の選び方



出典：http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

機械学習の得意なこと、苦手なこと

- 機械学習の得意なこと
  - 人間では見つけられないような複雑なルールを見つけ出すこと
  - 環境の変化（データの変化）にも柔軟に対応できること

- 機械学習の苦手なこと
  - すでに人間にとって自明となっているルールを改めて見つけること
  - 少ないデータしか得られていない状況

機械学習のプロジェクトの進み方

- いきなり機械学習を本番環境に適用しようとすると多くの困難が伴います
- 解決したい問題に対して機械学習が適用可能かを判断するための小さな実験を行います
- この実験のことをPOC（Proof of Concept：実証実験）と呼びます
- 機械学習を実社会で応用していく辺り、非常に重要な考え方となります
- 本講義は、POCを実施するための最低限の知識づくり、を目指します

※一方で、数学に関する知識は基本的に不要とし、ライブラリをうまく使い機械学習を扱えるようになることを目指します

分析環境の構築

- 前提としてはPythonは3系の最新版である3.6系
- 利用するライブラリは以下のもの
  - Numpy, Pands, Scipy, Scikit-learn, TensorFlow, matplotlib, jupyter
    - （＋一部の章で追加のライブラリを導入します）
- 構築方法としては下記2つ
  - 素のPythonに必要となるライブラリをインストールしていく方法
    - Pythonの環境を自力で構築できる方向け
  - Anaconda(miniconda)を利用する方法
    - Pythonの環境を自力で構築する自信がない方向け
    - インストーラを使って必要なソフトウェア一式のインストーラを行う

1. 素のPythonからインストール

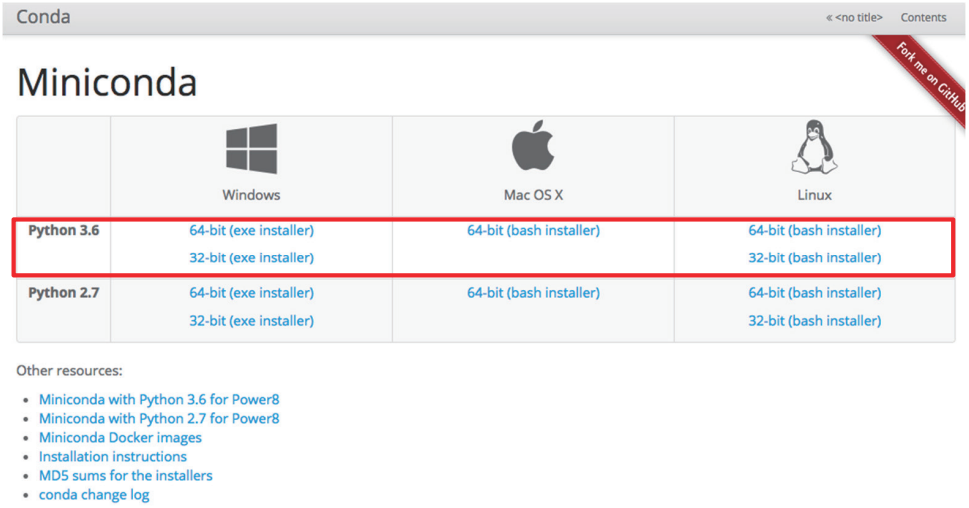
- Pythonにはパッケージ管理システムであるpipが標準で付属しているため、pipを利用して各種ライブラリをインストールする
  - Python3系そのもののインストールは割愛

```
$ pip install jupyter matplotlib scikit-learn numpy pandas nltk scipy
```

2. Anaconda(miniconda)を利用してインストール

- AnacondaはPython及び分析に必要なライブラリ一式を提供してくれるパッケージ
- minicondaはAnacondaの中で分析に必要な最小限のみ集めた軽量のディストリビューション
  - Anacondaのダウンロードページ（https://conda.io/miniconda.html）から使用しているOSに合わせたインストーラを取得してダウンロードする
  - 今回はPython3.6系が前提のため、Python3.6系のインストーラを利用する

2. Anacondaを利用してインストール（続き）



3. minicondaの使い方

- 環境を作る

```
$ conda create -n analytics
```

- 環境のアクティベート

```
$ activate analytics
```

- 環境内でのパッケージインストール

```
$ pip(conda) install jupyter matplotlib scikit-learn numpy pandas nltk scipy
```

- 環境の終了

```
$ deactivate analytics
```

以後の作業

- Jupyter notebook上での作業を推奨とします。
- コマンドを実行したディレクトリをルートとしてjupyterが起動します。

```
$ jupyter notebook
```

Jupyter

- Webブラウザから Pythonなどのプログラミング言語を実行させる環境
- もともとは、Pythonのインタラクティブシェルをより強力にしたIPythonが始まり
  - セル単位でのプログラム実行とタブ補完
  - オブジェクトの調査とシェルコマンドの実行
  - etc...
- IPythonをWeb経由に実行させられるものとして、IPython Notebookが登場
  - テーブルやグラフ、数式の埋め込み
  - Markdownの埋め込み
  - .ipynb形式により、分析過程自体を再実行性を担保して保存と共有が可能に
- IPython NotebookをPython以外の言語にも対応させたものがJupyter Notebook
  - Python
  - R
  - Julia

第2回：機械学習による分類問題への対応

分類問題とは？

アジェンダ

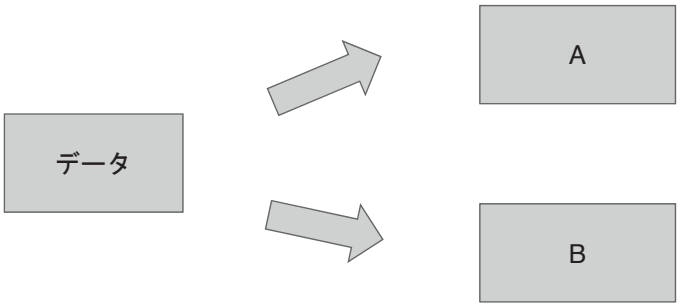
- 分類問題とは？
- 2値分類問題の概要
- 2値分類問題の典型的な問題
- 多値分類問題の概要
- 多値分類問題の典型的な問題
- 分類問題を評価する指標について説明
- まとめ

分類問題とは？

- 分類問題とは、与えられたデータがどのグループに属するものか、を判別する機械学習における典型的な問題
- 教師あり学習に属する問題領域
  - 事前にどんなデータがどのグループに属するかの情報が与えられる
  - その情報をもとに学習
  - 未知のデータに対して、どのグループに属するのかの判断を下す（推論）

2値分類問題の概要

- 2値分類問題は、AであるかBであるか、という2つのどちらかに属するかを判別する分類問題の最も基礎的な問題
- AとBという選択肢は言い換えると、AとA以外、というようにも表現できるため、非常に幅広く活躍できる問題



2値分類問題の典型的な問題設定

ある病院では患者の検査データを管理している。患者を病気か、病気ではないかを判断したい。これは機械学習の問題として取り扱うことができるか？できるならどのように行うか？

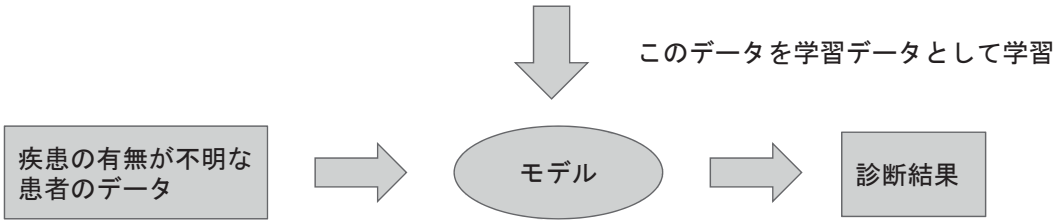
2値分類問題の典型的な問題設定

ある病院では患者の検査データを管理している。患者をある特定の病気か、病気ではないかを判断したい。これは機械学習の問題として取り扱うことができるか？できるならどのように行うか？

病気：クラス1、病気ではない：クラス2　とすれば2値分類問題となるため、機械学習を使った分類として取り扱うことができる。

2値分類問題として扱うイメージ

患者ID	血圧（下）	血圧（上）	その他の検査結果	疾患の有無
0001	80	120	...	有
0002	75	135	...	無
...				
0100	90	140	...	有

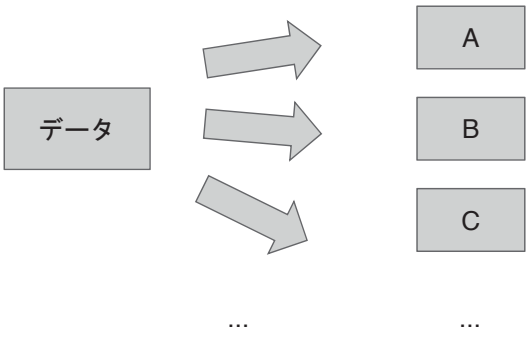


演習：2値分類問題の設定

2値分類問題に落とし込める問題を設定してみよ。まずはじめに問題を設定し、
どのようなデータを使い、どのように活用するのか、という観点でまとめること。

多値分類問題の概要

- 多値分類問題は、A,B,C....という複数のもののどれに属するかを判別する分類問題の中でも難易度が高い問題



演習：多値分類問題の難しさ

一般的に2値分類問題は多値分類問題よりも難易度の高い問題となる。それはなぜか理由をまとめてみよ。

演習：多値分類問題の難しさ

一般的に2値分類問題は多値分類問題よりも難易度の高い問題となる。それはなぜか理由をまとめてみよ。

- 1つのクラス（AとかBとか）に属するデータの量が少なくなってしまう。機械学習においてデータの量は非常に重要な要素
- 複数のクラスに分類させる方法は多く考えられる。どれが良いかは問題によって異なる。
  - 一度にどのクラスに所属するのかを計算してしまう方法
  - あるクラスに属しているか、属していないかを、すべてのクラスについて判断していく方法
  - etc...

多値分類問題の典型的な問題設定

ある病院では患者の検査データを管理している。患者がどのうような病気にかかっているのか（あるいはそもそも病気にはかかっていないのか）を判断したい。これは機械学習の問題として取り扱うことができるか？できるならどのように行うか？

多値分類問題の典型的な問題設定

ある病院では患者の検査データを管理している。患者がどのうような病気にかかっているのか（あるいはそもそも病気にはかかっていないのか）を判断したい。これは機械学習の問題として取り扱うことができるか？できるならどのように行うか？

病気A：クラスA、病気B：クラスB.... 病気ではない：クラスZ のように設定すれば多値分類問題として機械学習で取り扱うことができる。ただし、病気の種類分クラスが必要となる

モデルの評価

- 学習がどれくらいうまくいったかを評価するプロセス
- 単純に分類がどれくらいうまくいったかだけに着目していると、過学習に陥ってしまう可能性があるため注意が必要
- 評価するための代表的な指標や考え方は下記の通り
  - Accuracy（精度）
  - Confusion Matrix（混同行列）
  - Precision（適合率）、Recall（再現率）、F値
  - Precision-Recall Curve
  - ROC Curve（Receiver Operating Characteristic Curve：ROC曲線）
- いずれも教師あり学習の手法で適用されるもの
  - 教師なし学習は、そもそも正解のラベルがないため、評価がしにくい

Accuracy（精度）

- 予測されたラベルが、正解ラベルに対してどれくらいあっているかという指標
- 例えば、以下の場合を考える
  - 予測されたラベル：[0, 1, 1, 1, 0]
  - 正解のラベル：[0, 1, 1, 0, 0]
  - 5個中4個正解なので、精度は4 / 5 x 100 = 80% となる
- 精度は一般的な指標であるが、精度のみを見ていると見落とす情報も多い

Confusion Matrix（混同行列）

- 真のラベルと、予測したラベルについて行列で表現したもの
- 精度だけでは見えなかった「正と予測したが実は負だったラベル」「負と予測したが実は正だったラベル」が確認できるようになる
  - 「正と予測したが実は負だったラベル」は偽陽性とも呼ばれ、問題があることを見逃してしまうことに相当するので、問題によっては割けなければいけない間違い
  - 「負と予測したが実は正だったラベル」は偽陰性とも呼ばれ、問題がないものを問題であるとしてしまうことに相当する

混同行列の例

- データ件数5件
- 正しいラベル[0, 1, 0, 1, 0]とする（0は負、1は正のラベルとする）
- 予測したラベル[0, 1, 1, 0, 0]とする

		予測したラベル		
		正	負	合計
正しいラベル	正	1	1	2
	負	1	2	3
	合計	2	3	5

Precision, Recall, F値

- Precision（適合率）とは、正と予測したラベルのうち、実際に正だったものの割合
  - すなわち、正しく予測できたものの、全体に対する割合
- Recall（再現率）とは、実際に正であるラベルのうち、実際に正と予測されたものの割合
  - すなわち、正しく予測されたものの全体に対する割合
- F値とは、適合率と再現率の調和平均
  - 適合率と再現率は一般的にトレードオフの関係にあるため、双方を同時に評価するための指標

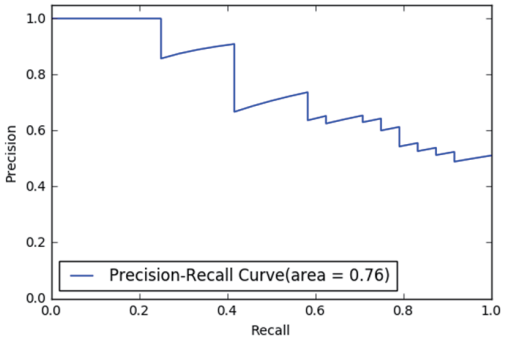
混同行列とPrecision/Recallの関係

		予測したラベル		
		正	負	合計
正しいラベル	正	1	1	2
	負	1	2	3
	合計	2	3	5

混同行列からPrecisionとRecallは算出することができる
・Precision = 1 / 2 = 0.5
・Recall = 1 / 2 = 0.5

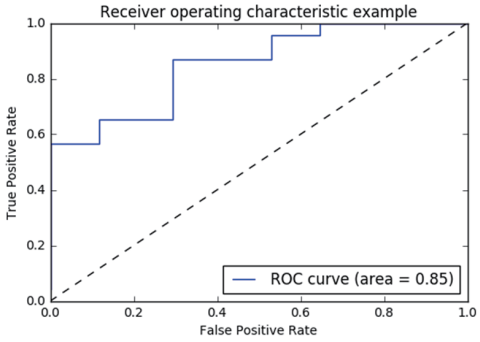
Precision-Recall Curve

- データの件数をランダムに1件から順に増やして行きながら、適合率と再現率を調べてプロットしたもの
- このグラフの下側部分をAUC（Area Under the Curve）と呼び、精度の指標として用いられる



ROC曲線

- 横軸にFalse Positive（偽陽性）の値、縦軸にTrue Positive（正しく分類できた）の値をプロットする
- Precisino-Recall Curveと同じく、データを1件からランダムに取り出し、上記の値を調べ、プロットしていったもの
- 同じく曲線下部の面積はAUCと呼ばれる



AUCについてもう少し

- Precision-Recall Curveでも、ROC曲線でもAUCは0から1までの値を取る
- 完全にきれいな分類ができているものは面積が1となる
- 一方、完全にランダムに分類した際も面積は0.5となる
  - つまり、一般的には0.5より大きな値となる
  - 0.5を下回る場合は、非常に良くない状態

演習問題

教師あり学習で分類を取り扱えるアルゴリズムは数多くある。それぞれのアルゴリズムを調べて、その特徴は何か、どのような原理で動作するのかを調べよ

第3回：機械学習による分類問題への対応（応用）

分類問題のアルゴリズムを体験する

アジェンダ

- 前コマの振り返り
- 最近傍法についての概要及びアルゴリズムの解説
- 最近傍法を用いた演習問題
- ナイーブベイズについての概要及びアルゴリズムの解説
- ナイーブベイズを用いた演習問題
- パーセプトロンについての概要及びアルゴリズムの解説
- パーセプトロンを用いた演習問題
- 本コマのまとめ

最近傍法の概要及びアルゴリズムの解説

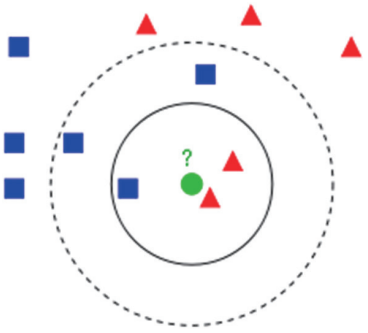
最近傍法はパターン認識などで最もよく用いられる手法の一つ

アルゴリズムの簡単な解説

- 教師データを特徴空間に分布させる
- 分類させたいデータを特徴空間にマッピングさせた時に最も近いクラスに分類させる
- 学習で処理時間がかかるということはないが、全てのデータとの距離を計算する必要があるため、データを与える毎に一定の処理時間がかかる

参考：k近傍法

- k近傍法は距離の近いk番目までの点から、どのクラスに分類されるかを判断する手法
- k近傍法のk=1の場合が最近傍法である



・緑のデータを分類したい
・近傍数kを3とする
・赤い三角の方が数が多いため、緑のデータは赤い三角側のクラスと分類される
・例えば近傍数kを5とすると結果は変わる

出典：https://upload.wikimedia.org/wikipedia/commons/thumb/e/e7/KnnClassification.svg/220px-KnnClassification.svg.png

最近傍法を用いた演習問題

Scikit-learnで提供されているデータセットiris（あやめのデータ）を使って最近傍学習を適用してみる

- まずはじめにデータセットを確認する
- 学習データとテストデータに分割する
- 次に学習データに対してscikit-learnで提供されている最近傍法のクラスを適用する
- 最後にテストデータに対して最近傍法を適用して結果を確認する

ナイーブベイズの概要及びアルゴリズムの解説

主に文書分類などで用いられる教師あり学習の手法。代表的な事例にスパムメールの判定が挙げられる

アルゴリズムの簡単な解説

※ナイーブベイズ自体はベイズ統計についての知識が必要になるため、その本質的な説明はここでは割愛する

- 文書分類という観点で言うなら、文書を単語に分解する
- 文書 - 単語 - 文書のカテゴリという3つ組のデータを構築する
- データをもとにベイズ統計の論理に従う（ライブラリ呼び出しに任せる）

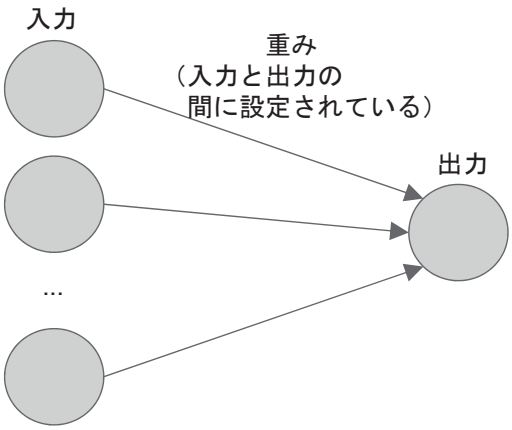
ナイーブベイズを用いた演習問題

Scikit-learnで提供されているデータセット20newsgroup（ニュースのデータ）を使ってナイーブベイズを適用してみる

- まずはじめにデータセットを取得して、どのようなデータか確認する
- データは学習データとテストデータに分割する
- 次に学習データに対してscikit-learnで提供されているナイーブベイズクラスを適用する
- 最後にテストデータに対してナイーブベイズを適用して結果を確認する

パーセプトロンの概要及びアルゴリズムの解説

パーセプトロンはニューラルネットワークの最もシンプルな形態のもので、入力と出力だけで決まるもの



①入力は出力へ向かう際に、設定されている重みを掛け合わせる
②それぞれの入力 x 重み の結果を足し合わせて出力を計算する
③出力には一定のしきい値が設定されており、しきい値を超えるかどうか分類されるクラスが決まる
④パーセプトロンの学習とは、入力と出力を見比べて、正しくなかった場合、重みを更新することに相当する

パーセプトロンを用いた演習問題

Scikit-learnで提供されているデータセットiris（あやめのデータ）を使ってパーセプトロンを適用してみる

- まずはじめにデータセットを確認する
- 学習データとテストデータに分割する
- 次に学習データに対してscikit-learnで提供されているパーセプトロンのクラスを適用する
- 最後にテストデータに対してパーセプトロンを適用して結果を確認する

精度向上のポイント：特徴量の標準化

特徴量毎にデータの分布やスケールに偏りがあると学習がうまくいかない

このような偏りを解消させるために、データ分布の平均と標準偏差を使いデータを0から1、もしくは-1から1にスケールし直すことを標準化という

標準化のやり方は様々であるが、一般的には以下のように行う

- 各列ごとの平均と標準偏差を計算する
- データの値から平均を引き、標準偏差で割る

まとめ

- 分類問題へ対応するためのアルゴリズムとして以下の3つのアルゴリズムを紹介した
  - 最近傍法
  - ナイーブベイズ
  - パーセプトロン
- いずれも基本的なアルゴリズムではあるが、2値分類だけでなく多値分類にも適用できるアルゴリズムとなっている
- 精度向上のアプローチとして特徴量の標準化を紹介した

第4回：機械学習による回帰問題への対応（基礎）

回帰問題とは？

アジェンダ

- 回帰問題とは？
- 線形回帰問題の概要
- 線形回帰問題の典型的な問題
- 非線形回帰問題の概要
- 非線形回帰問題の典型的な問題
- まとめ

回帰問題とは？

- 回帰問題とは説明変数と呼ばれるデータから目的変数と呼ばれるデータを予測する、分類問題と並ぶ典型的な問題の一つ
- 分類問題との違い
  - 分類問題はカテゴリカルなデータを取り扱う
  - 回帰問題は連続値を取り扱う
- AかBかCか、といったものではなく、数値（1, 0.5など）を結果として得られるもの

線形回帰問題の概要

- 線形の関係を前提として回帰を行うこと
- 線形の関係とは、簡単に言うと加算と乗算の関係のみだけで表される関係

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon\,.$$

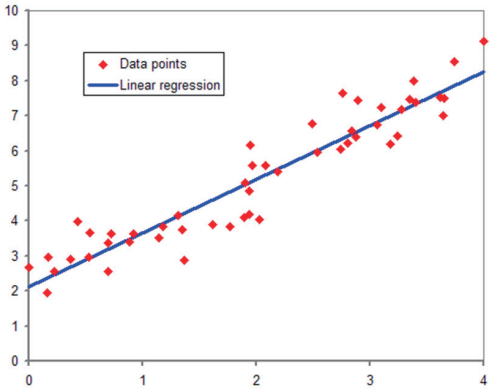
- この関係を使って目的変数を説明するようなモデルを構築する

数式はWikipediaより転載
https://ja.wikipedia.org/wiki/%E7%B7%9A%E5%BD%A2%E5%9B%9E%E5%B8%B0

線形回帰問題の典型的な問題

気温を横軸、売上を縦軸にプロットして気温と売上の関係を求めよ、など

応用範囲は大きい



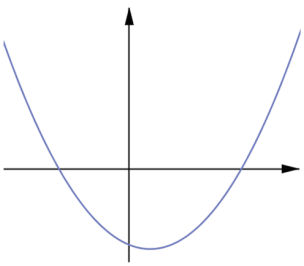
赤い点が与えられた時、関係性を表すために、
一番シンプルにする場合は、青い直線を
引けば良い。これが回帰分析である

例えば、直線は１次関数　y = ax + b　にて
描くことができる。
すなわち、aとbを求めることに相当する

出典：https://upload.wikimedia.org/wikipedia/commons/b/be/Normdist_regression.png

非線形回帰問題の概要

- 非線形回帰は線形回帰で扱いきれないより複雑な問題を取り扱うことのできる枠組み
- xの累乗を説明変数に持つため、複雑な関係性を記述できる
  - 簡単に言うと、直線の関係ではない、ということ
- 数学的にはかなり高度になるため、アルゴリズム部分の説明は割愛



参考：http://www.avocado-fes-thought.com/WhatIsNonlinearity.html

非線形回帰の典型的問題設定

基本的には線形回帰で取り扱う問題と相違はない。あくまで回帰問題に対して、非線形な関係を使って関係性を記述しているだけ

まとめ

- 回帰問題について説明した
- 線形回帰問題について概要を説明した

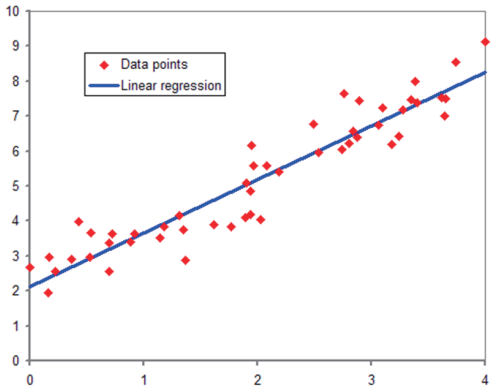
第5回：機械学習による回帰問題への対応（応用）

回帰問題に取り組む

アジェンダ

- 前コマの振返り
- 線形回帰についての概要及びアルゴリズムの解説
- 線形回帰を用いた演習問題
- ロジスティック回帰についての概要及びアルゴリズムの解説
- ロジスティックを用いた演習問題
- 一般線形モデルについての概要及びアルゴリズムの解説
- 一般線形モデルを用いた演習問題
- 本コマのまとめ

線形回帰問題の概要



赤い点が与えられた時、関係性を表すために、
一番シンプルにする場合は、青い直線を
引けば良い。これが回帰分析である

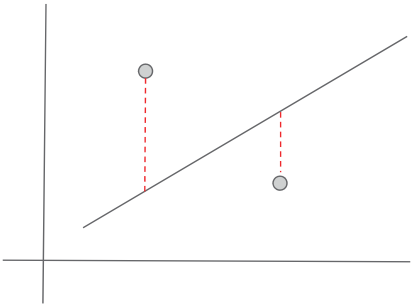
例えば、直線は１次関数　y = ax + b　にて
描くことができる。
すなわち、aとbを求めることに相当する

青い直線を引くためのアルゴリズムが最小二乗法

出典：https://upload.wikimedia.org/wikipedia/commons/b/be/Normdist_regression.png

最小二乗法

- 最小二乗法は偏微分などのやや高度な数学的な話が出てくるため、ここではイメージの説明のみとする
- 端的に言うと、直線とデータ点の差が最も小さくなるような傾きと切片を求めることに相当する
  - 各データ点と直線の差を足し合わせ、それが最小となるような傾きを決める



赤い点線の合計が最
も小さくなるように
直線を引く

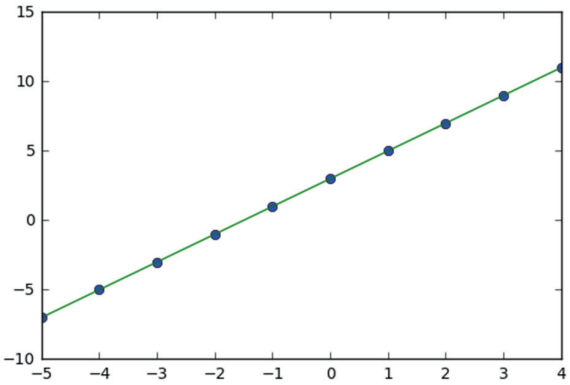
Pythonによる最小二乗法

- Numpyに最小二乗法を行ってくれる関数が用意されているため、それを利用する
- numpy.polyfit
  - また、その後のグラフ描画に便利なpoly1dも提供されているため、非常に便利

- まずはじめに関数が分かっている状態で最小二乗法を試してみる
  - y = 2x + 3
  - 最小二乗法を適用して、2と3に近しい値が求められることを確認する

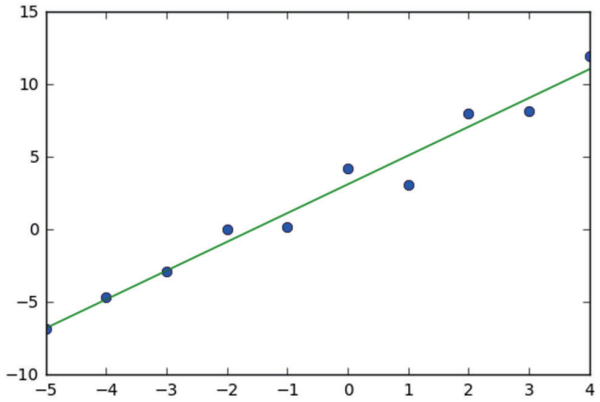
最小二乗法による回帰：１次関数（1/2）

- 線形関数を元にして作成したデータ点から、最小二乗法で直線を引くと確かに直線が引かれることが確認できる



最小二乗法による回帰：１次関数（2/2）

- 線形関数を元にして作成したデータ点にノイズを乗せ、より実践的な状態で最小二乗法を使う
- 確からしい直線が引かれることが確認できる



演習：irisデータに対しての最小二乗

- irisデータの「がく片の長さ」と「花片の長さ」の関係に対して最小二乗法を用いて回帰を行い直線を引いてみよ

単回帰と重回帰

- 変数を一つだけ使う回帰を単回帰と呼ぶ
- 変数を複数使う回帰を重回帰と呼ぶ

一般的に重回帰の方が複数の変数を使うため、複数の変数の関係性を扱うことができる

演習：bostonデータに対する線形回帰

- 回帰分析用のデータセットである boston に対して線形回帰を行ってみる
- 単回帰と重回帰の双方を行う

決定係数

- 回帰の当てはまりの良さとして用いられる尺度
- 様々な定義があるが一般的には以下の式に従うR^2が決定係数となる
  - Scikit-learnでは簡単に求めることができる

$$R^2 \equiv 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \bar{y})^2}$$

出典：https://ja.wikipedia.org/wiki/%E6%B1%BA%E5%AE%9A%E4%BF%82%E6%95%B0

ロジスティック回帰についての概要及びアルゴリズムの解説

- ロジスティック回帰はデータ分布がベルヌーイ分布に従う回帰モデルの一種
  - ベルヌーイ分布とは正規分布のように確率分布の一つの形態
- 幅広く利用されている手法

$$\operatorname{logit}(p_i) = \ln\left(\frac{p_i}{1-p_i}\right) = \alpha + \beta_1 x_{1,i} + \cdots + \beta_k x_{k,i},$$

$$i = 1, \ldots, n,$$

数式で表すと上記のような形式
※本講義では数学的には深く入り込まない

出典：
https://ja.wikipedia.org/wiki/%E3%83%AD%E3%82%B8%E3%82%B9%E3%83%86%E3%82%A3%E3%83%83%E3%82%AF%E5%9B%9E%E5%B8%B0

ロジスティック回帰を用いた演習問題

- 回帰と名前がついているが事実上分類問題に適用する手法
- irisデータにロジスティック回帰を適用してみる

一般化線形モデルについての概要解説

- 線形回帰は正規分布を前提としている
- これを正規分布以外にも利用できるように拡張した手法
  - つまり、様々なデータ分布を前提におくことができる
- 一般化線形モデルは高度な数学的な知識が必要になるため、深入りはしない

- ちなみに、ロジスティック回帰も一般化線形モデルとして扱われるモデルである

総合演習

- Bostonデータに対して様々な変数を使って回帰を行ってみよ
- その際、変数間の関係や決定係数などを確認し、精度について考察せよ

まとめ

- 最小二乗法について学んだ
- ロジスティック回帰について学んだ
- 一般化線形モデルについて学んだ

第6回：決定木による問題解決

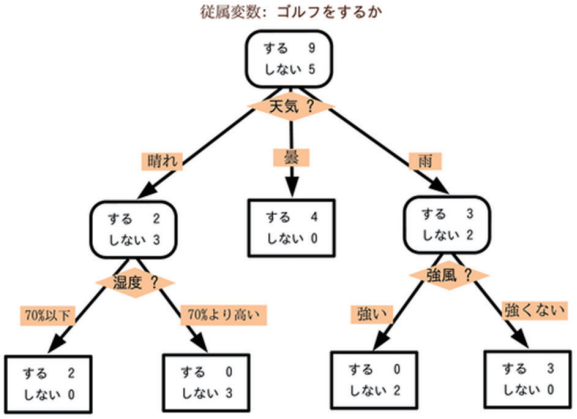
決定木を理解する

アジェンダ

- 決定木とは
- 決定木の代表的アルゴリズム
- 決定木による分類
- 決定木による回帰
- 本コマのまとめ

決定木とは？

- 目的となる値を導くための決定を行うためのグラフ
- 木のような形をしているため、決定木と呼ばれる



出典：https://ja.wikipedia.org/wiki/%E6%B1%BA%E5%AE%9A%E6%9C%A8

決定木のメリット

- 機械学習がどのように意思決定したのかを後から確認することができる
- ブラックボックス型の機械学習ともっとも違う点
- 「なぜ」そのような結果になったのか、に対して答えることができる手法
- 後述する、ランダムフォレストなど、強力な手法の基本ともなっている

決定木の代表的アルゴリズム

- 基本的な考え方としては、ある項目で情報を分類した時により「情報量」が小さくなるように分類を行っていく
  - 情報量とは「整理の進行具合」のようなもの
- 情報量の基準としてはいくつかの指標がある
  - ジニ不純度（ジニ係数）
  - 情報エントロピー

ジニ不純度（ジニ係数）

もともとは経済学で考えられた、あるグループにおける所得の格差の大小を定量的に評価するための指標

- 0か1の間の値を取る
- ジニ係数の値が大きいほどグループの不平等が大きい、ことを表す

$$\mathrm{Gini} = \frac{1/2 - \int_0^1 L(F)dF}{1/2} = 1 - 2\int_0^1 L(F)dF.$$

出典：https://ja.wikipedia.org/wiki/%E3%82%B8%E3%83%8B%E4%BF%82%E6%95%B0

情報エントロピー

情報理論という学問分野で取り扱われるある出来事の起こりにくさを表す指標

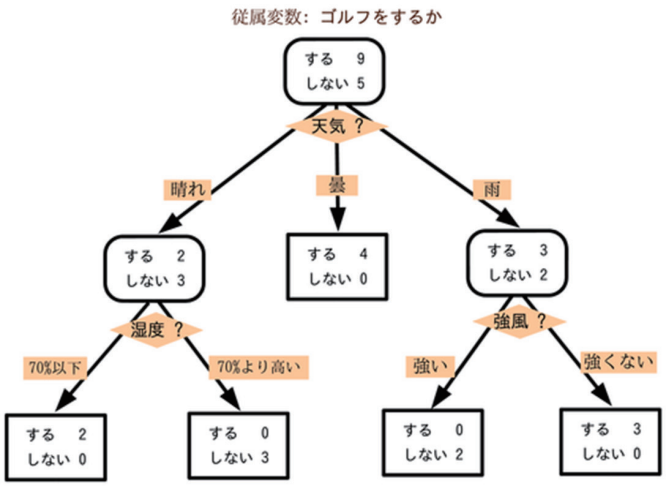
- バラツキが大きいほど大きな値を取る
- 情報エントロピーは以下の数式で表される

$$H(P) = -\sum_{A \in \Omega} P(A) \log P(A)$$

決定木による分類

ゴルフクラブ来場状況

独立変数				従属変数
天気	気温(℃)	湿度(%)	風が強いか	ゴルフをするか
晴れ	29	85	強くない	しない
晴れ	27	90	強い	しない
曇	28	78	強くない	する
雨	21	96	強くない	する
雨	20	80	強くない	する
雨	18	70	強い	しない
曇	18	65	強い	する
晴れ	22	95	強くない	しない
晴れ	21	70	強くない	する
雨	24	80	強くない	する
晴れ	24	70	強い	する
曇	22	90	強い	する
曇	27	75	強くない	する
雨	22	80	強い	しない



出典：https://ja.wikipedia.org/wiki/%E6%B1%BA%E5%AE%9A%E6%9C%A8

決定木による分類の演習

- IRISデータで分類を試してみる
- ジニ係数を用いた分類を試してみる
- エントロピーを用いた分類を試してみる
- その他のパラメータを変更して分類を試してみる

決定木による回帰

分類とほぼ同じアプローチで、扱うデータが回帰になっただけのもの

- つまり、AかBか、ではなく、連続値（1.0とか1.3）などを対象とすればそれは回帰木となる。

bostonデータセットを用いて回帰を試してみる

まとめ

- 決定木の基本を学んだ
- 決定木は判断過程を明示できるのが大きなメリットの一つ

第7回：アンサンブル学習による問題解決

アンサンブル学習を理解する

アジェンダ

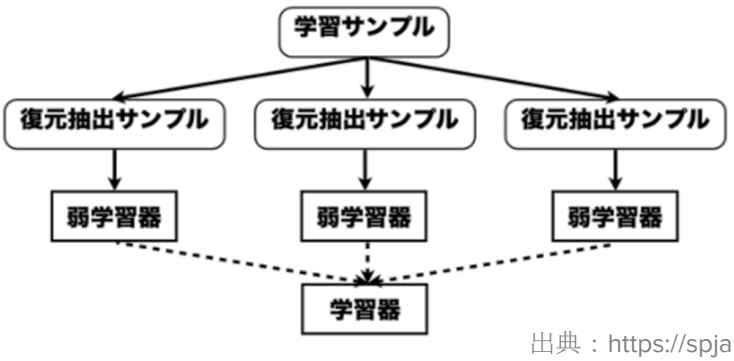
- アンサンブル学習とは？
- ランダムフォレスト
- アダブースト
- まとめ

アンサンブル学習の概要

- 一つの高度な学習器をつくるのではなく、弱学習器と呼ばれるシンプルな学習器を複数作り、多数決方式で処理を行う方法の総称
- 一般的に、並列処理が可能であり、良い学習結果が得られることが多く人気のある手法
- 代表的な手法
  - RandomForest
  - AdaBoost
  - XGBoost

バギング

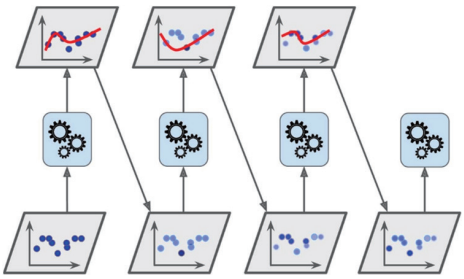
- 学習データの一部を使って学習する方法
  - 復元抽出サンプルといい、毎回異なるデータを利用する
- 並列に学習が可能になる
- 代表的な手法：ランダムフォレスト



出典：https://spjai.com/ensemble-learning/

ブースティング

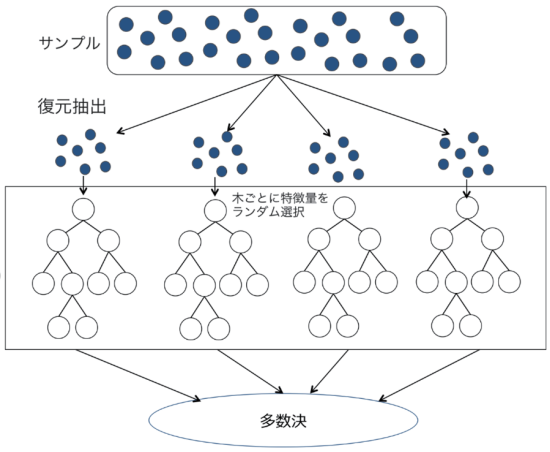
- 基本的にバギングと同じ枠組み
- 前に使ったデータを再度利用して「ブースト」していく
- そのため、バギングのような並列化はできない
- 代表的手法：AdaBoostg



参考：http://st-hakky.hatenablog.com/entry/2017/08/07/163216

ランダムフォレストの概要

- ランダムフォレストは弱学習木として決定木を使う手法
- 木を複数作る
- サンプルデータ全体からデータのサブセットを取得
  - サブセットは木の数だけ取得
- それぞれの木に適用して結果を集めて多数決



出典：https://alphaimpact.jp/2017/03/30/decision-tree/

ランダムフォレストの演習

breast cancerデータセットを使った演習

- ランダムフォレストで実際にデータセットを分類する
- グリッドサーチでパラメータの調整を行う

アダブーストの概要

アルゴリズムイメージ

- 弱学習器を組み合わせてデータを分類する
- 誤分類したデータに重みをつけて分類結果が改善するようにする
- 以後繰り返していく

アダブーストの演習

breast cancerデータセットを使った演習

- アダブーストで実際にデータセットを分類する
- グリッドサーチでパラメータの調整を行う

まとめ

- アンサンブル学習の基本を学んだ
- アンサンブル学習の代表的な手法として２つの手法を紹介した
  - ランダムフォレスト
  - アダブースト

第8回：次元削減による問題解決

次元削減を理解する

次元削減の概要

- データの特徴量数のことを次元とも呼ぶ
- 人間は3次元以上のデータを直感的に理解するのは困難
  - 場合によっては3次元でも困難
  - 2次元が最も理解しやすい
- データの見える化（可視化）も基本的には3次元以下に次元を落とす必要がある
- なるべくもとの特徴量と同等の情報量を保持したまま次元を削減する方法を次元削減と呼ぶ

次元、特徴量

ID	Column1	Column2	Column3
A	1	aaa	True
B	2	bbb	False

表形式のデータでは列方向が特徴量、次元となる
上記の例では、3次元のデータが2点あることになる。

次元の呪い

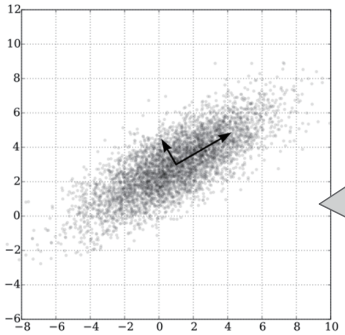
- 一般的に　データ数 >> 特徴量の次元数　となっていないと良い学習が行えない
  - 探索空間（次元数）の増加に応じてよりたくさんのデータが必要になるため
- つまり、データとしては、行方向 >> 列方向 となっている必要がある
- これは直感的にも正しい
  - 行方向 << 列方向 なデータの場合、1レコードあたりの情報が多い割にデータ点としては少ない。したがって、集まったデータ点だけでは、データ全体の特徴をうまく捉えられていない可能性が高い

次元削減

- 特徴量が多くある（高次元）データを、全体の情報を損なわずに低次元のデータに変換すること
  - 1000ある特徴量を3つの特徴量に絞り込む
    - 単純に1000の中から3つを選ぶ場合、特徴選択(Feature Selection)と呼ぶ
    - 1000の特徴量を合成し、3つの特徴量を新たに作る場合、次元削減(Dimensionaly Reduction)と呼ぶ
- メリット
  - 不要な情報を減らすことができるため、本質的な情報のみが残される
  - 低次元になることにより、可視化を行うことができる
- デメリット
  - 次元を落としすぎると必要な情報まで削られてしまい、全体の情報量が減ってしまう
  - 適切な次元の落とし方は簡単に求まらない

主成分分析

- 次元削減の手法で最も古典的であり有名な手法
- 特徴量の中で、相関の少ないものを選び出し、新たな特徴量を作ることにより、少ない特徴量でデータ全体の特徴を表現させる方法



【直感的説明】
・散らばってるデータから見て最も説明力の高い軸を新たに探す
・この例の場合、楕円状に散らばっているデータの真ん中を通る2軸
・数学的には共分散行列の固有値と固有ベクトルを計算することで求めることができる

参考：https://commons.wikimedia.org/wiki/File:GaussianScatterPCA.svg#/media/File:GaussianScatterPCA.svg

PCAの演習

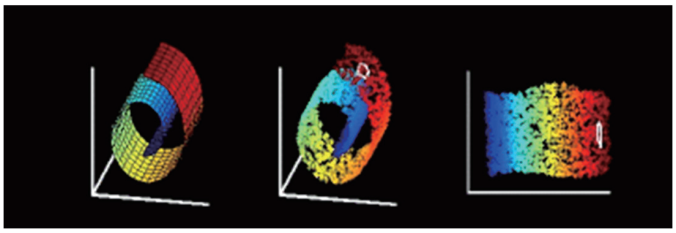
IRISデータを利用した演習

- IRISデータは4次元データである、これを2次元データに次元削減せよ

LLEの概要

どんなに複雑な形状でも、局所的に見るとユークリッド（線形の関係）で表せることを前提とした手法

- 巻物状にデータが分布している
- データの構造を保持したまま3次元データを2次元化
  - 巻物を開くようにデータを展開



出典：https://cs.nyu.edu/~roweis/lle/swissroll.html

LLEの演習

- 巻物上のデータ（Swiss roll）を使う
- Swis rollにLLEを適用して低次元化を図る
- また、irisデータにもLLEを適用してみる

総合演習

- IRISデータに対してPCAを用いて4次元データを3次元へと次元削減し、結果を可視化せよ
- 次元削減を具体的にどのような問題に適用すればよいか考えてみよ

まとめ

- 次元削減の必要性を述べた
- 次元の呪いについて説明した
- 次元削減の代表的手法を2つ紹介した
  - PCA
  - LLE

Apendix

- 主成分分析の数学的な理解には固有値・固有ベクトルなどの線形代数の知識が必要になる
  - 必要に応じて線形代数を学習するとより理解が深まる
- LLEの数学的な理解には多様体学習と呼ばれる分野の知識が必要となる
  - LLE自体はユークリッド距離をベースとしたアルゴリズムのため、そこまで高度な知識自体は不要

第9回：レコメンデーションの体系的理解

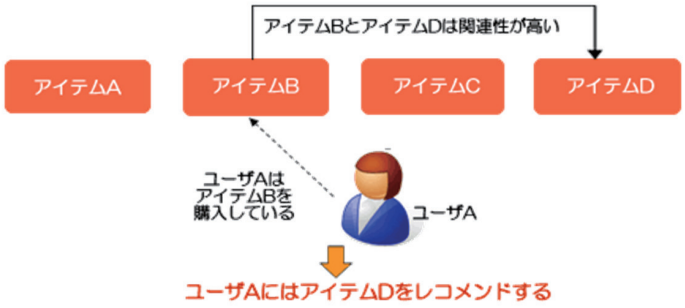
レコメンデーションを理解する

レコメンデーションとは？

- ECサイトなどで見かける「この商品を買った人はこんな商品も買っています」の裏側で動く推薦ロジック

アイテムベースレコメンド

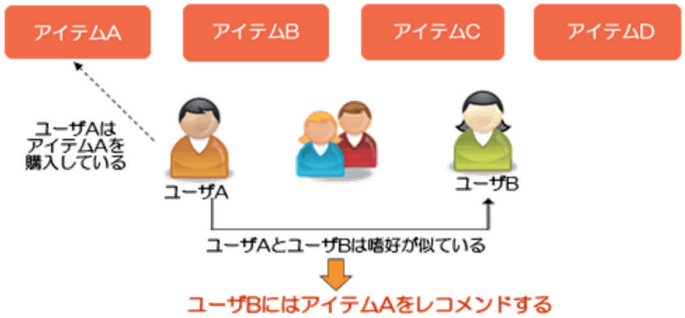
- 傾向の似ている商品をレコメンドする手法
  - アイテムにつけられるレビューの評価の傾向が似ている商品をおすすめする
  - アイテムの特徴が似ているアイテムをおすすめする
  - etc...



出典：http://www.activecore.jp/column/rwh_2/

ユーザーベースレコメンド

- 嗜好が似ているユーザーの情報をベースにレコメンドする
  - AさんとBさんは似たような好み
  - Aさんが商品Xを買った
  - Bさんに商品Xをおすすめする



出典：http://www.activecore.jp/column/rwh_2/

「似ている」の尺度

尺度化できるものならなんでも良い

- ユークリッド距離（の逆数）
- 加重平均
- ピアソンの相関
- etc...

データも同様にスコアがつけられるものならなんでも良い

- 行動ログ
- レビュー結果
- etc...

ユークリッド距離

- ユークリッド距離はいわゆる直感的な距離と同義
- 点pとqの距離は以下で表される

$$d(\mathbf{p},\mathbf{q}) = d(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

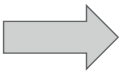
出典：
https://ja.wikipedia.org/wiki/%E3%83%A6%E3%83%BC%E3%82%AF%E3%83%AA%E3%83%83%E3%83%89%E8%B7%9D%E9%9B%A2

協調フィルタリングの概要

協調フィルタリングはユーザー x アイテムに関するデータをもとにユーザー間の類似度を求めて推薦商品を決めるユーザーベースレコメンドの手法

ユーザ	作品1	作品2	作品3	作品4	ダークナイト
鈴木さん	5	3	4	2	?
ユーザ1	3	1	2	3	3
ユーザ2	4	3	4	2	5
ユーザ3	3	3	1	5	4
ユーザ4	1	5	5	2	1

データ例



-	鈴木さん
鈴木さん	0.0000000
ユーザ1	0.2773501
ユーザ2	1.0000000
ユーザ3	0.2132007
ユーザ4	0.2182179

ユークリッド距離の逆数

出典：http://blog.brainpad.co.jp/entry/2017/02/03/153000

協調フィルタリングの演習

- scikit-learnには実装されていない
- step-by-stepで進めていく
  - データセットの準備
  - ユーザー同士の距離の定義
    - ユークリッド距離の逆数
    - スピアマンの相関係数
  - ユーザーの加重平均からの類似度算出
    - 類似度からのオススメ選出

総合演習

- 身近なテーマをもとにレコメンデーションを行ってみよ
- まずはじめにデータを ユーザー x アイテムの形式に変更する
- その後、類似度を決めて協調フィルタリングのロジックを構築する

まとめ

- レコメンデーションの概要を学んだ
- レコメンデーションの基本として以下2つがあることを学んだ
  - アイテムベースレコメンド
  - ユーザーベースレコメンド
- レコメンドの代表的手法である協調フィルタリングを学んだ

第10回：クラスタリングによる教師なし学習

クラスタリングを理解する

教師なし学習とは？

- 教師なし学習はラベルを持たないデータに対する機械学習の手法を指す
- 明確に正解と言われるものがないため、結果に対して何らかの指標を元に評価することはできない
- データの背後にある構造を探し、それを元にデータをまとめたり（クラスタリング）、次元を減らす（次元削減）を行うような手法群

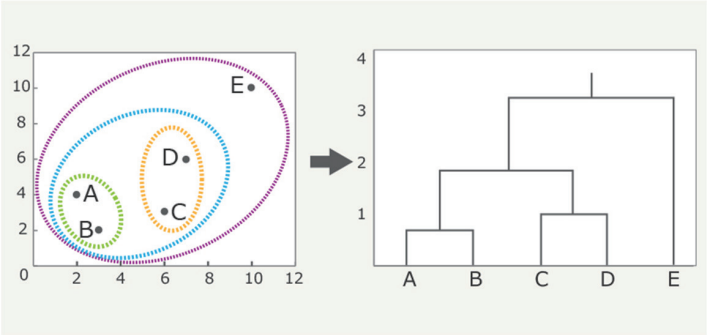
クラスタリング

- 与えられたデータを何らかのルールに従って分類する手法
  - ラベルなどの外部情報なく、データそのものから特徴を見つける方法

- 大きく分類すると階層的クラスタリングと非階層的クラスタリングがある
  - 階層的クラスタリングの代表的手法はウォード法
  - 非階層的クラスタリングの代表的手法はk-means

階層型クラスタリングの概要

- 階層構造を作りながらクラスタをマージしていく
- どこまでマージしたものを一つのクラスタとみなすかはケースバイケース
- 代表的な手法としてはウォード法がある



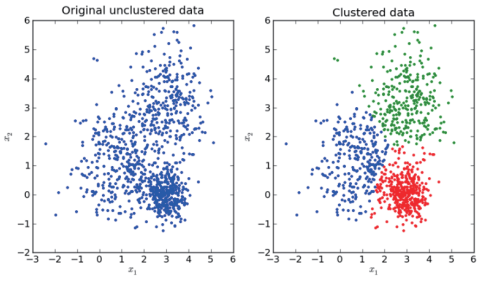
出典：http://business.nikkeibp.co.jp/atclbdt/15/258678/071500002/?ST=print

階層型クラスタリングを利用した演習

- 階層型クラスタリングには scipy.cluster.hieralchy を利用する
- 階層型クラスタリングをirisデータに適用してみる

非階層型クラスタリングの概要

- いわゆるクラスタリングと聞いて思い浮かべるような処理
- データを適当な方法でグルーピングする
- 代表的な手法はK-means法



出典：https://stackoverflow.com/questions/24645068/k-means-clustering-major-understanding-issue

k-means

一般的に下記のようなアルゴリズムである

1. 各データをランダムにクラスタ（クラス）に割り振る
2. 割り振ったデータからクラスタの中心を求める
3. 中心と各データ点の距離を求め、最も近いクラスタの中心に割当て直す
4. 2,3を繰り返し、一定の変化量以下となったら処理を打ち切る

非常にシンプルなアルゴリズムだが、同じクラスたが近い距離にあるような場合には非常にうまく働く

k-means法を利用した演習

- シミュレーションにより演習に使うデータを生成する
- そのデータに対してK-meansを適用してみる

総合演習

- irisデータに対してK-meansを適用してみよ
- irisデータにはあらかじめラベルが付けられている。クラスタリング結果を確認し、どれくらい正しくクラスタリングによってクラス分類が行われているか確認せよ

まとめ

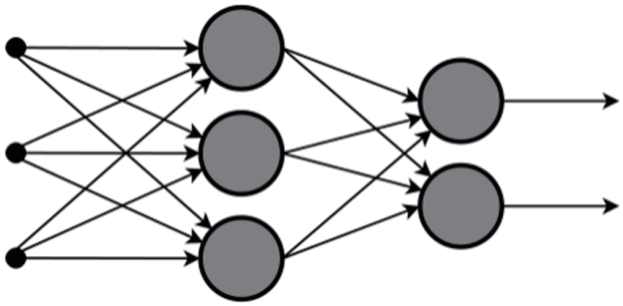
- 教師なし学習としてクラスタリングを学んだ
- クラスタリングの手法として階層型クラスタリングと非階層型クラスタリングがあることを学んだ
- 階層型クラスタリングとしてウォード法を使った手法を学んだ
- 非階層型クラスタリングとしてk-meansを使った手法を学んだ

第11回：ニューラルネットワークと深層学習（基礎）

ニューラルネットワークの基本を理解する

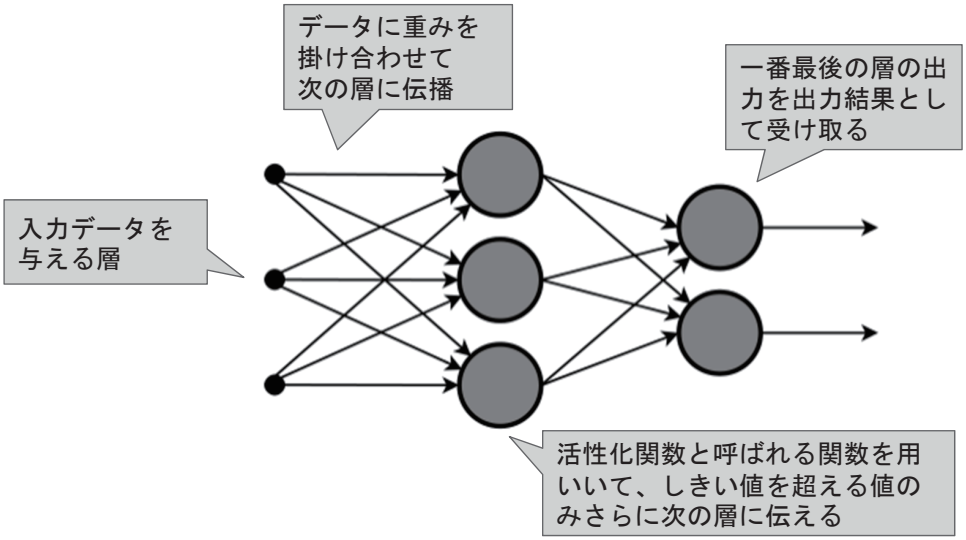
ニューラルネットワークとは？

- 機械学習の一手法
- 人間の脳の活動を模したアルゴリズム



出典：
https://ja.wikipedia.org/wiki/%E3%83%8B%E3%83%A5%E3%83%BC%E3%83%A9%E3%83%AB%E3%83%8D%E3%83%83%E3%83%88%E3%83%AF%E3%83%BC%E3%82%AF

ニューラルネットワークの基本



TensorFlow / Kerasとは？

- TensorFlowとは、Googleが2015年11月にオープンソース化した機械学習のライブラリ
  - 主にディープラーニングの実装が簡単になる
- こちらの資料により詳しい解説がある
  - https://speakerdeck.com/rindai87/talk-about-ml-and-dl-for-happy-engineers-life

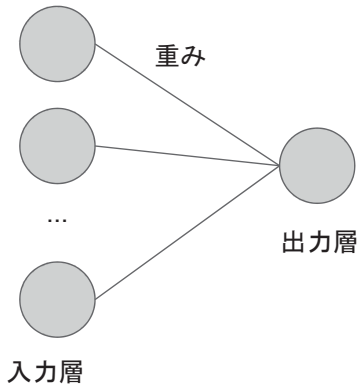
TensorFlow / Kerasのインストール

- TensorFlow, Kerasともインストール自体は簡単
- ついでに13回以降に利用する画像処理ライブラリのpillowもインストール
- モジュールを有効化するためにjupyterの再起動が必要

```
$ pip(conda) install tensorflow keras pillow pydot h5py
```

単純パーセプトロンの概要

- 入力層と出力層からなるニューラルネットワークの最もシンプルな形式
- 活性化関数としてはステップ関数が使われる
  - ステップ関数とは入力が0以下の場合は0、0より大きい場合は1を出力する関数

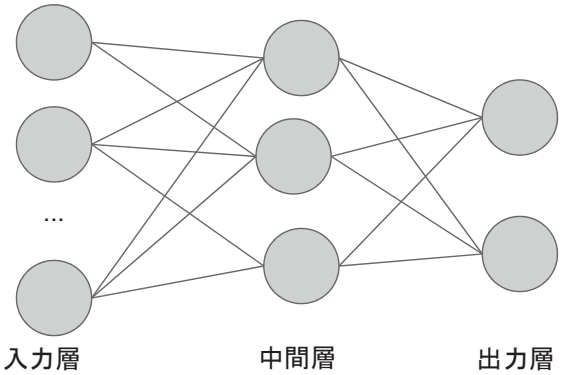


単純パーセプトロンを用いた演習

- irisデータにkerasを使って単純パーセプトロンを適用する
- irisデータは3クラスのデータ・セットであるが、単純パーセプトロンは2クラスしか扱えない
  - データの前処理で2クラス分類を行う必要がある
  - Kerasにはstep関数は提供されていないので、代わりにsigmoid関数で代用する

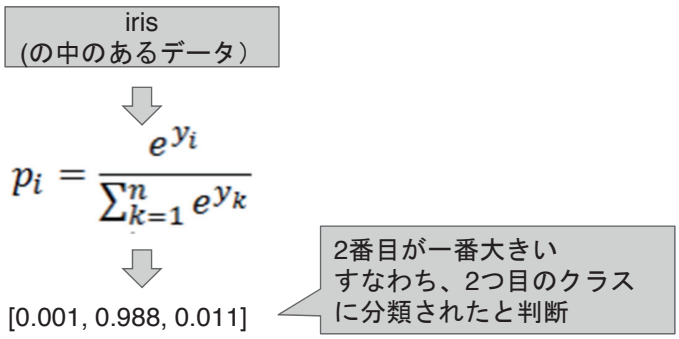
多層パーセプトロンの概要

- 多層パーセプトロンは単純パーセプトロンに少なくとも１つの中間層が追加されたもの
- 出力も少し工夫があり、softmax関数というものを利用する
  - softmax関数を使うと、多クラス分類問題を取り扱えるようになる



softmax関数

- softmax関数は以下の式で表される
  - nはクラス数を表す
- softmaxは「そのクラスに属している確率」を表す
  - softmax関数の結果が最も値が大きいクラスに分類された、と考える



多層パーセプトロンを用いた演習

- irisデータにkerasを使って多層パーセプトロンを適用する
- 単純パーセプトロンを拡張し、多層パーセプトロン化せよ
  - 変更点は、ネットワーク、活性化関数、ロス値、etc...

まとめ

- ニューラルネットワークの基本的な形としてパーセプトロンを紹介した
- TensorFlow / Kerasの基本について紹介した
- 単純パーセプトロン / 多層パーセプトロンをデータセットに適用した

第12回：ニューラルネットワークと深層学習

深層学習について理解する

前コマの振り返り

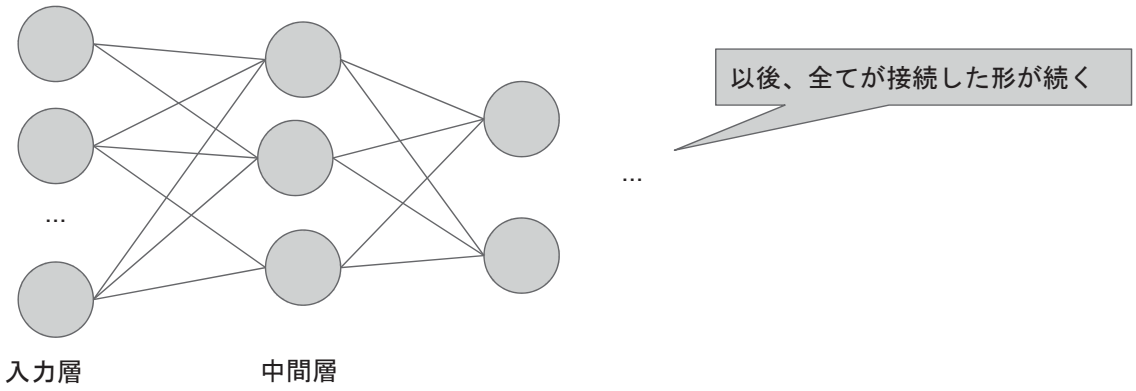
- ニューラルネットワークの基本とパーセプトロンについて学んだ
- TensorFlowの基本について学んだ

深層学習とは？

- ディープラーニングのこと
- 広義の定義としては多層パーセプトロンの中間層を非常に多くしたもの（数十層以上）のもの
  - これをDeep Neural Network(DNN)という
- そこから派生してネットワークの作り方など様々なものが登場している
- 14回では画像処理に強みのあるCNNを紹介する

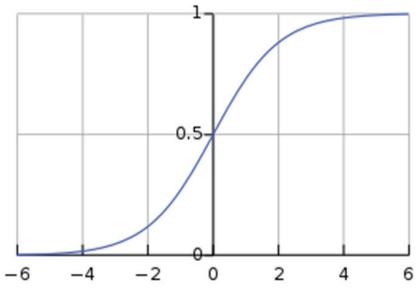
全結合ネットワーク

- 事実上パーセプトロンと同じもの
- ある層が次の層の全てのユニットと結合しているもの
- 一般的にDNN（Deep Neural Network）と呼ぶ

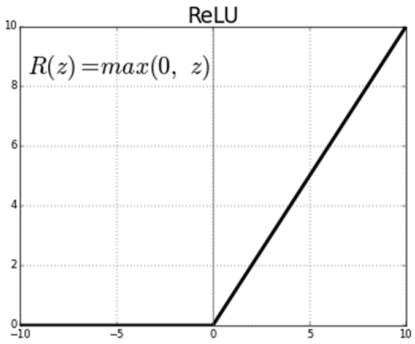


シグモイド関数 / ReLU関数

いずれもステップ関数よりも効率的に学習が進むことが知られている



sigmoid関数

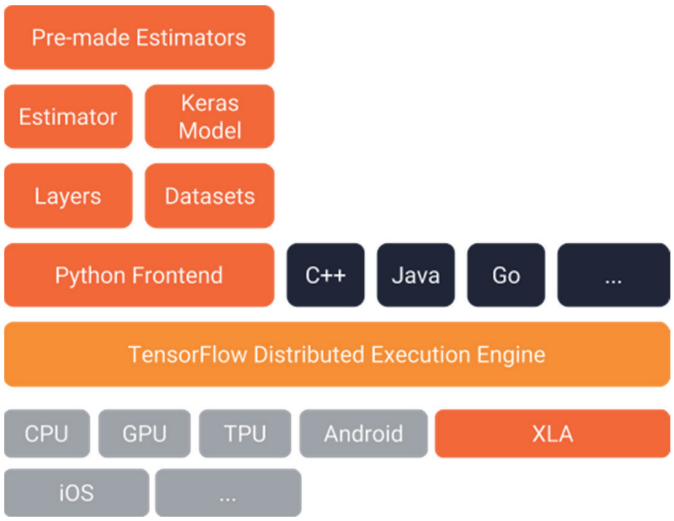


LeRU関数

TensorFlowの高レベルAPI

- TensorFlowにはより簡単にTensorFlowを使えるようにするための高レベルAPIが提供されている
- 高レベルAPIは現在充実の一途を辿っている過程である
  - tf.contrib.learnにScikit-learnライクなAPIがまとまっている
  - その他高レベルAPIは様々な流儀がある

参考：TensorFlowのテクノロジースタック



出典：https://developers.googleblog.com/2017/09/introducing-tensorflow-datasets.html

TensorFlowの高レベルAPIを用いたDNN

- DNN等のネットワークは高レベルAPIで提供されている
- そのため、API呼び出しだけで簡単にDNNを実行できる

```
tensorflow.contrib.learn.DNNClassifier()
```

演習問題

- IRISデータにTensorFlowの高レベルAPIを適用する

KerasによるDNN

- KerasによるDNNは実はすでに実装済み
- 多層パーセプトロンの層を深くするだけで事実上DNNとなる
- またパーセプトロンで利用したステップ関数は使われない
  - 代わりにシグモイド関数やReLU関数が利用される

演習問題

- 先程の演習問題と同じネットワークを構築する
- IRISデータにKerasによるDNNを適用する

総合演習問題

- iris以外のデータを対象にDNNを構築する
- まずはirisに適用したものと同じネットワークを適用し、結果を確認せよ
- より深いネットワークを持ったDNNを構築せよ

まとめ

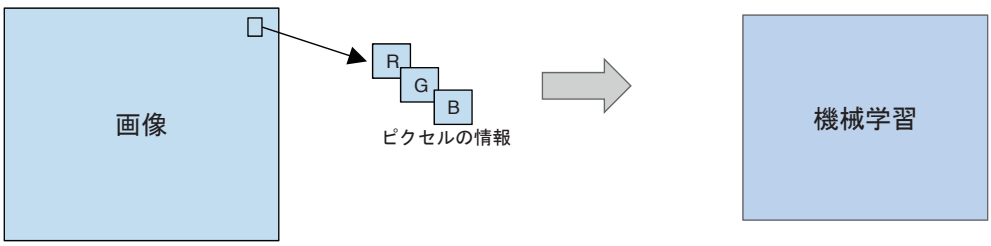
- 深層学習の全結合ネットワークであるDNNを学んだ
- TensorFlowの高レベルAPIを学んだ
- TensorFlowの高レベルAPI及び、KerasによるDNNを適用した

第13回：CNNによる画像処理（基礎）

CNNを理解する

画像処理の基本

- 画像データ（カラー）は厳密には各ピクセル毎にさらに、RGB三種類の情報を持っている。この数値情報を用いて画像処理は行われる
- 100 x 100ピクセルのカラー画像は、100 x 100 x 3（RGB）分のデータとなる。つまり画像データは特徴量として非常に大きくなる



Pythonによる画像処理

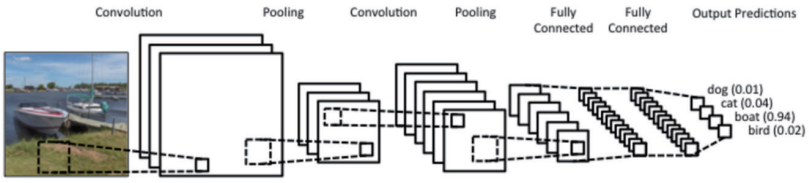
- Pythonによる画像処理ライブラリとしてはPILが有名
- PILの後継として、Pillowがあり、本講義ではこちらを利用する
    - 画像の回転、リスケール、フィルター、etc...と画像処理に必要な基本的な処理は全てPillowで実行可能
- TensorFlow / Keras上での画像処理に関する一通りのことはできるが、こちらも押さえておくと良い

Pillowを用いた画像処理の演習

- Pillowを用いた画像処理を実際に行ってみる
- 画像は自身で所有しているもの、あるいはライセンス的に問題のない画像を個人利用として使うとよい
    - たとえばこちらのサイトなどから画像は探しても良い

CNNによる画像処理

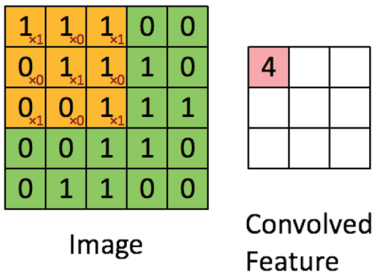
- CNNとはConvolutional Neural Netowrk（畳み込みニューラルネットワーク）の略で、ディープラーニングの一手法
- 「畳み込み」とはその名の通り、あるピクセルの周辺のピクセル情報を考慮して計算する（畳み込む）ことが根幹にあることから名付けられた方法
  - 画像処理では一般的な概念



出典：http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

畳み込みのイメージ

1	0	1
0	1	0
1	0	1



例えば、以上のようなフィルタを用意する
画像にこのフィルタを重ね合わせて、
フィルタと画像のピクセル同士をかけ合わせて総和を取り、
結果をフィルタの中央に一致するピクセルの出力とする

出典：http://postd.cc/image-scaling-using-deep-convolutional-neural-networks-part1/

CNNを構成する畳み込み

- CNNは様々な層がある
- Convolution層（畳み込み層）
  - 前項で説明した
  - 畳み込むことで、複数ピクセルの情報を集約することに相当する
- Pooling層
  - フィルタをかけ合わせて最大（最小）のピクセル情報のみ残す層
  - 画像の縮小に相当する
- Fully Connected層
  - 全結合層。DNNで紹介したものと同じ

KerasによるCNN

- Kerasを使うと比較的簡単にCNNを構築できる
- Kerasに提供されている各種ヘルパー関数を使うと、層の定義が非常に簡単にできる
- その他以下を行うとCNNが実装できる
  - 入力画像を用意すること
  - 層を適切に選択し、ネットワークを構築すること
  - 実際に学習を行うこと

CNNを利用した演習問題

- CIFAR10というデータセットを使う
  - Kerasで簡単に利用できるようになる
- CNNを構築し、分類を行う

まとめ

- 画像処理の基本について説明した
- Pythonを使った画像処理の基本を説明した
- CNNの基本について説明した
- Kerasを用いたCNNの演習を行った

第14回：CNNによる画像処理（応用）

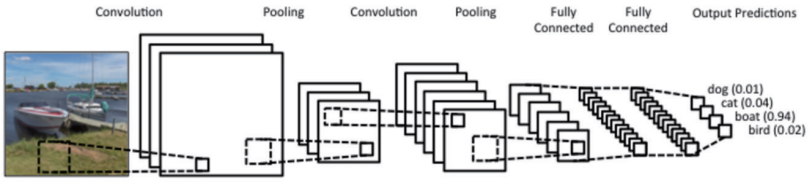
転移学習を理解する

前コマの振り返り

- 深層学習の一手法としてCNNを学んだ
- CNNは畳み込みニューラルネットワークの略語で、主に画像処理に力を発揮する手法

CNNの改良方法

- CNNを1から改良するのは非常に大変
  - どのようなネットワークを作るのか？
  - 学習データとしてどのようなものを使うのか？
  - 学習をどのくらい実行するのか？



学習済みネットワークの活用

- 学習済みネットワークとは？
  - すでに学習を終えた状態で公開されているネットワーク
  - 論文等で高い性能を達成した有名なネットワークは学習済みモデルとして公開されている

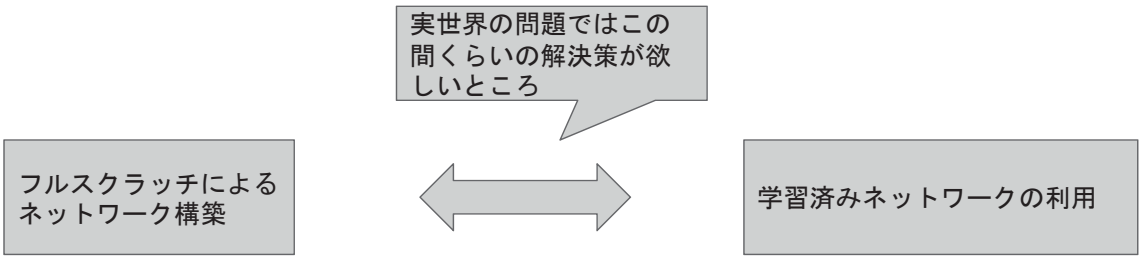
- 学習済みネットワークのメリット
  - 学習を行わずにすぐに推論を行える
    - 画像分類のタスクはすぐに実行可能
- 学習済みネットワークのデメリット
  - もし推論で性能が出なかった時に対処のしようがない
    - すでに学習済みのため

Kerasによる学習済みネットワークの利用

- Kerasを使うと非常に簡単に学習済みネットワークを利用することができる

- 演習
  - 学習済みネットワークを利用して画像の分類を行う

ここまでの知識の整理



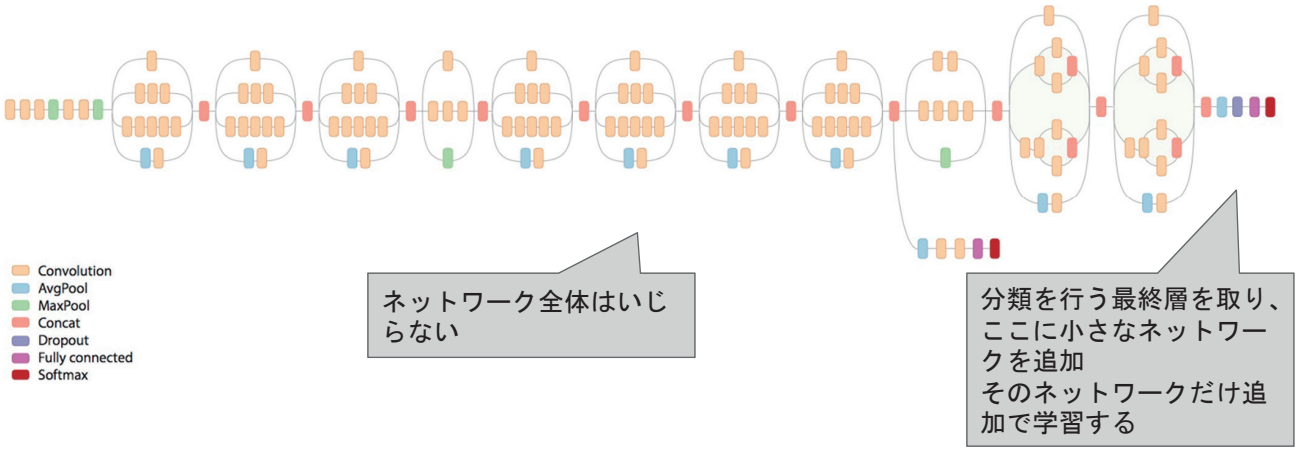
・ネットワークの検討が必要
・大量の学習データを使った学習が必要
・学習に耐えうる実行環境が必要

・ネットワークは固定
・学習は不要
・推論だけならそこまで
ハイスペックな環境は不要

転移学習 / ファインチューニング

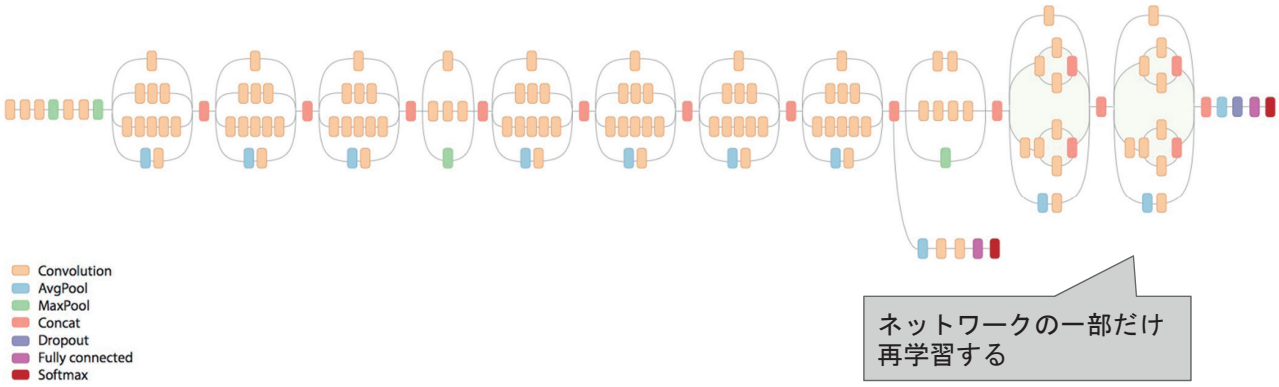
- 学習済みネットワークを活かす方法
- 大きく分けると転移学習とファインチューニングの２つがある
- 転移学習
  - 学習済みネットワークの一部を改造する
  - 最終の分類層などを取り去って浅いネットワークを追加
  - そこだけ学習する
- ファインチューニング
  - 一部の層だけ再度学習させる

転移学習



出典：https://research.googleblog.com/2016/03/train-your-own-image-classifier-with.html

ファインチューニング



出典：https://research.googleblog.com/2016/03/train-your-own-image-classifier-with.html

Kerasによる転移学習

- Kerasを使うと比較的簡単に転移学習とファインチューニングが実現可能
- Kerasを利用して転移学習を体験してみる

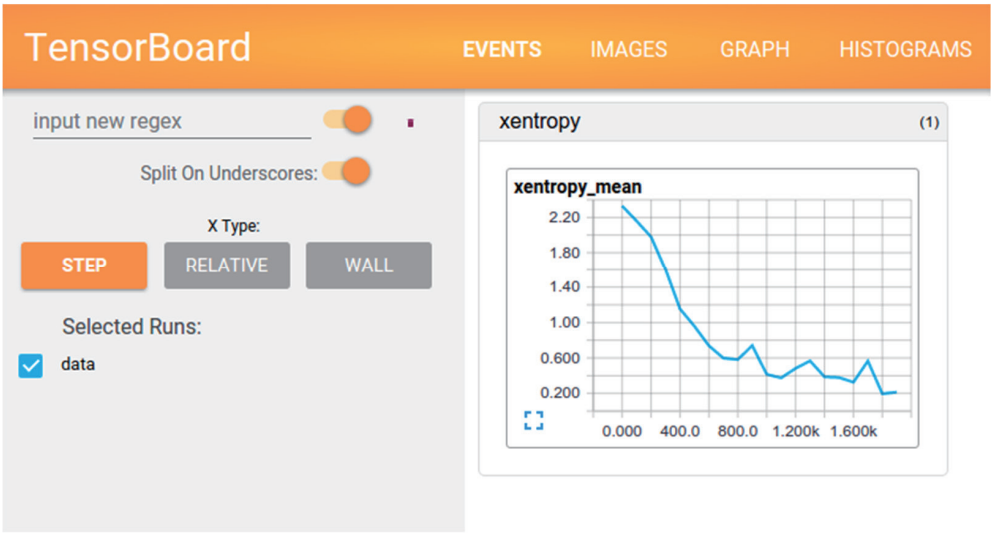
- 体験したらわかるが基本的にGPUがない状態では、ある程度以上の規模のネットワークは学習させるのは辛い
  - 演習問題として体験する

TensorBoard

- TensorFlowに付与しているツール
- TensorFlowに関する様々な情報を可視化してくれる
- 以下のコマンドで起動する

```
$ tensorboard --log=./transfer_model
```

TensorBoard



出典：https://www.tensorflow.org/get_started/summaries_and_tensorboard

まとめ

- 学習済みネットワークについて学んだ
- Kerasによる学習済みネットワークの利用方法を学んだ
- 転移学習とファインチューニングについて学んだ
- Kerasによる学習済みネットワークの利用方法を学んだ

第15回：機械学習プロジェクトのチェック項目

機械学習プロジェクトについて学ぶ

機械学習プロジェクトとは？

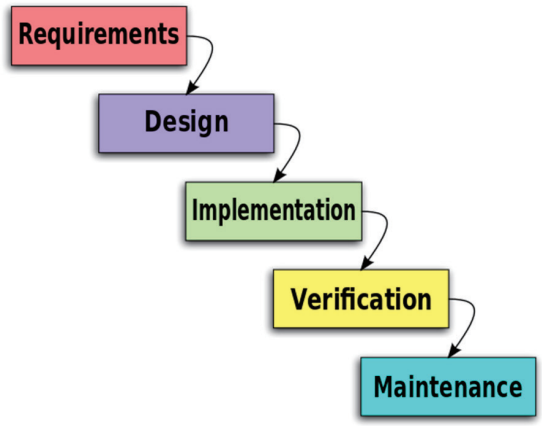
- ビジネスやサービスで機械学習を活用しようとした時、機械学習のアルゴリズム以外にも気を使わなければいけないことが数多くある
- 実際に機械学習を活用するためには、プロジェクトという形で進めていく

機械学習の特性

プロジェクトとして考えた場合、機械学習の「試してみるまでどのような結果が得られるかは分からない」というのは大きな足かせとなる

いわゆるウォーターフォール型の一直線に計画から実装まで進むタイプのプロジェクトマネジメント法とは相性が悪い

ウォーターフォール型



一つずつプロセスが進んでいく
基本的にプロセスを後戻りすることはない

出典：
https://ja.wikipedia.org/wiki/%E3%82%A6%E3%82%A9%E3%83%BC%E3%82%BF%E3%83%BC%E3%83%95%E3%82%A9%E3%83%BC%E3%83%AB%E3%83%BB%E3%83%A2%E3%83%87%E3%83%AB

PoC：Proof of Concept

Proof of Concept（概念実証）の略

ビジネス上の課題の解決に機械学習が役立つのかを見極める小さな実験のこと

想定される課題に対して、サンプルデータで機械学習アルゴリズムを適用してみる

機械学習プロジェクトの進行

- 課題のヒアリング
- 課題を解決するための機械学習アルゴリズムの検討
- PoCの実行
- PoCの結果を受けた要件定義
- 設計
- 実装
- テスト

実際はこのように一直線に進むわけではない

特にPoCの結果を受けて再度アルゴリズム検討に戻るケースもある

チェックすべき項目①：データ

データの量と質

- そもそも一定以上のデータ量は確保できるのか
  - 機械学習には一定以上のデータが必要
  - 一定以上に明確な定義はない
- データの質も重要
  - 学習を行うために重要な要素が含まれていないデータなら

チェックすべき項目②：プロジェクトメンバー

機械学習に長けたメンバーだけではプロジェクトはうまく回らない

以下のような3つの職種が三位一体に協力し合うことが必要

- プロジェクトマネージャー
  - 機械学習にもある程度精通しているのが望ましい
- エンジニア
  - システムの実装部分の責務を負う
- データサイエンティスト / 機械学習エンジニア
  - 機械学習部分担当

チェックすべき項目③：アルゴリズム

アルゴリズムは高度であれば良いというわけではない

- 可能な限りシンプルなアルゴリズムの検討
- ルールベースとの組み合わせの検討

ということが実運用においては重要となる

演習

- 実世界の問題を機械学習プロジェクトとして解決することを検討せよ
- どのようなデータを使うのか
- どのような計画を立てるのか
- 機械学習としてはどのようなアルゴリズムを使うのか

まとめ

- 機械学習プロジェクトについて説明した
  - 実世界に機械学習を適用するには重要なプロセス
- 単なるアルゴリズムだけなく、検討すべきことが数多くある
  - 逆に言うとそのような領域に興味を持つことが、機械学習の領域で活躍するためには重要

平成 30 年度「専修学校による地域産業中核的人材養成事業」
Society5.0 実現のための IT 技術者養成モデルカリキュラム開発と実証事業

平成 30 年度「専修学校による地域産業中核的人材養成事業」
Society5.0 実現のための IT 技術者養成モデルカリキュラム開発と実証事業

機械学習Ⅲ

平成 31 年 3 月

学校法人電子学園（日本電子専門学校）
〒169-8522　東京都新宿区百人町 1-25-4
TEL 03-3369-9333　FAX 03-3363-7685