ニコニコAIスクール 第2回
機械学習入門

18/01/13

講師：八木 拓真

本日の目標

- 機械学習の目的及び種類、評価方法を理解する
- 第1回に引き続き、numpyで頻出する関数の使い方を実践的に理解する
- k-NN法を理解し、numpyを用いて実装できる
- ファイル入出力の基本手順を学ぶ

機械学習アルゴリズムの種類

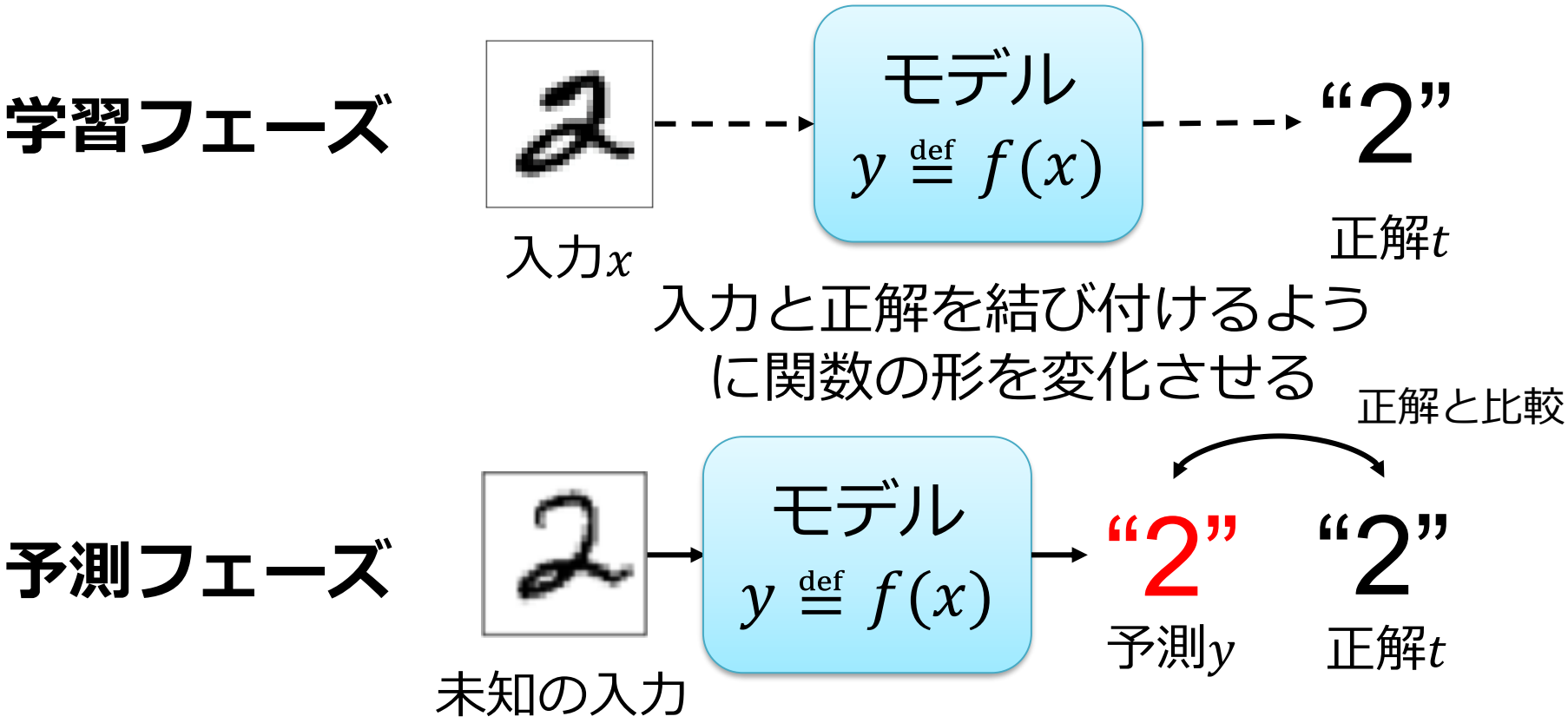
機械学習アルゴリズムの分類

- ざっくり3つ※に分けることができる：
  - 教師あり学習 (supervised learning)
  - 教師なし学習 (unsupervised learning)
  - 強化学習 (reinforcement learning)

※ とはいえ、3つに分ける必然性は薄く、実情はもう少し複雑である (semi-supervised learning, inverse reinforcement learning, positive-unlabeled learning, etc.、いろいろある)

教師あり学習

- 入力xに対する正解tを用意し、$y \stackrel{\text{def}}{=} f(x) = t$ なる対応関係を学習する
  - 集合 $\left(x^{(i)}, t^{(i)}\right)(i = 1 \ldots N)$ を教師データと呼ぶ



教師あり学習の具体例

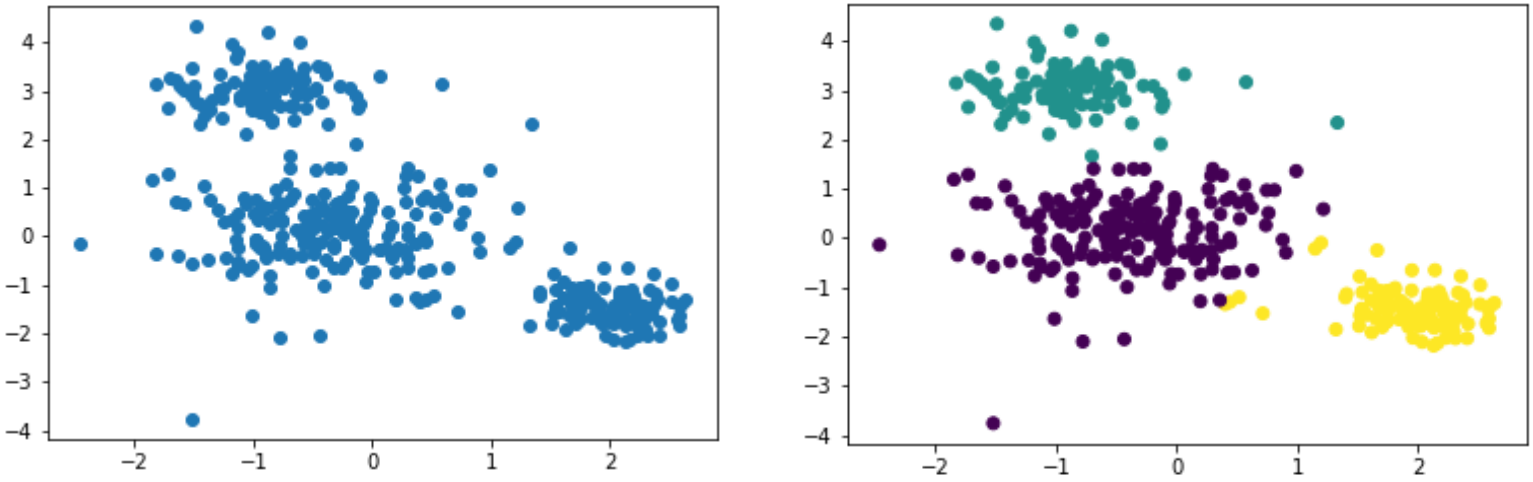
- 出力データの種類により2種類に分類できる：
  - 回帰：連続量を予測する
    - 販売量予測 (何個売れるか)
    - 株価予測 (将来の株価がいくらになるか)
    - 位置予測 (将来どの座標に移動するか)
  - 分類：(大小のない) 離散値を予測する
    - スパムメール分類 (スパムであるかないか)
    - 文字認識 (0~9のうちどれか)
    - 画像分類 (cカテゴリのうちどれか)

*予測と推論の違い

- ちょっと気になったので調べてみました：
  - 予測：あるタスクが与えられた時に、モデルを使ってタスクに回答すること
  - 推論：答えるだけでなく、起こりうる事象の確信度 (e.g. 確率、分散) を計算すること、(少数の) 予測因子の結びつき (e.g. 2群に差が認められるか否か、相関があるかないか) について何らかの結論を示すこと

教師なし学習

- データ$X = (x_1, \ldots, x_N)$が与えられた時、データを適当な規準に従って回帰・分類する
  - 明示的な正解は与えられない※



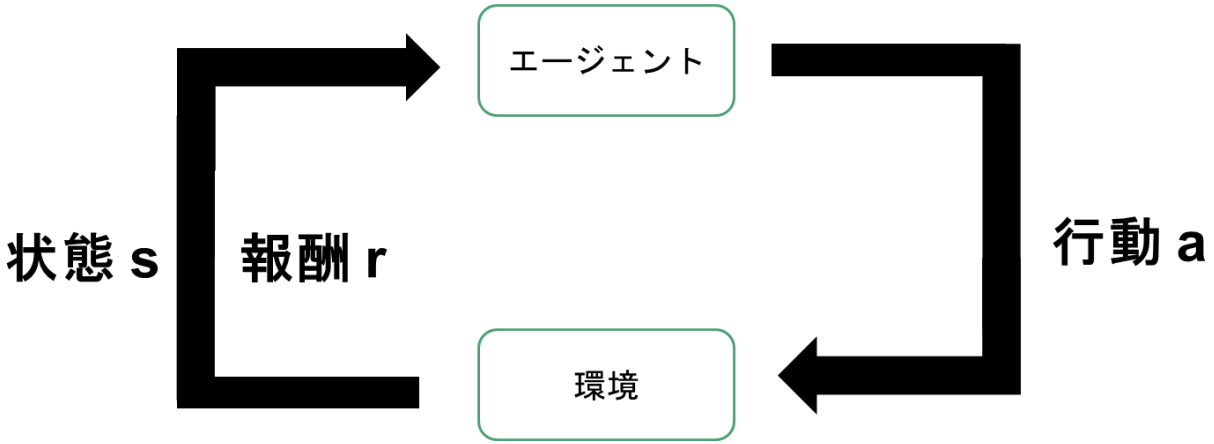
※ 教師なし学習と教師あり学習の境目はややあいまいであるが、欲しい出力を明示的に指定する場合は教師あり学習で、そうでない場合を教師なし学習と呼ぶのが一般的である。例えば、Autoencoderと時系列予測は共に自分自身を目的関数として使っている点で同じくくりになるが (self-supervised learningとも呼ばれる)、前者は中間層に現れる非明示的な特徴を得るのが目的なので教師なし、後者は目的関数に一致する将来の値自身を得ることが目的なので教師ありと判断される。

教師なし学習の具体例

- クラスタリング
  - ユーザが購入した商品一覧から 未知のユーザーセグメントをあぶりだす
- 異常検知
  - センサの値が正常の範囲を逸脱した場合に通知
- 次元圧縮
  - 高次元 (e.g. 100~10000次元) のデータをそのデータの特性を維持しつつ低次元に圧縮する
  - 例：PCAによる可視化 (平面にプロットできるようデータを2次元に圧縮)

強化学習

- 明示的な教師ペア (x, t) は与えられず、エージェントが行動した結果与えられる報酬から学習する
  - エージェントが試行錯誤しながら報酬が多くなるように学習し、所望のタスクを達成するような行動を獲得する (詳しくは第11回で)



※ 強化学習と教師あり学習もまた同様に境界があいまいである。実際、任意の教師あり学習問題は強化学習問題として定式化が可能であるし、一部の強化学習問題は教師あり学習として解釈することができる。ただ、強化学習問題は関数近似に留まらず、絶えず (自身の行動の影響も受けながら) 変化する環境中における振る舞いやインタラクションをも学習するという点で教師あり学習より汎用的な設定である。また、時間方向のある意思決定問題 (sequential decision problem) において人間が様々な可能性を網羅した教師データを提供することは多くの場合難しいため、その場合モデル自身が様々な状態を探索する方が未知の状態に対してよく働くことが期待できる。

強化学習の具体例

- ロボットの運動制御
  - 不整地での歩行に対して明示的なアルゴリズムを与えるのは不可能に近いが、ロボットが試行錯誤して歩いた距離を報酬として、人間では考えられないような行動を獲得する
- ゲームプレイ
  - Deep Q-Learning (DQN): 2D/3Dゲームをスコアが大きくなるように試行錯誤しながら学習、明示的に見本を見せずとも様々なゲームがプレイできるようになる

本講義で扱う手法たち

- 教師あり学習
  - パーセプトロン
  - k-近傍法 (k-Nearest Neighbor)
  - 線形回帰
  - サポートベクターマシン (SVM)
  - ニューラルネットワーク (NN) ※NN全てが教師ありというわけではない
- 教師なし学習
  - k-means
  - 主成分分析 (PCA)
  - (混合ガウスモデル・Autoencoder・RBM)
- 強化学習
  - Q学習 (Q-Learning)
  - Actor-Critic法
  - Deep Q-Network (DQN)

機械学習アルゴリズムの評価

機械学習アルゴリズムの評価

- アルゴリズムの性能を公平に評価するためには細心の注意が要ります
- 本節では次の事項について説明します
  - どのデータを使って評価するか
    - モデル選択
    - Hold-out validation
    - 交差検証 (K-fold Cross Validation)
  - どの指標を使って評価するか
    - 適合率 (Precision)
    - 再現率 (Recall)
    - F値 (F-value)
    - 正解率 (Accuracy)

機械学習を使う場面

そもそも、どのような場面で機械学習は有効だろうか？

→良質のデータが十分に存在する

→一定の法則性がある作業に恒常的に人手が割かれている (そしてそれが負担になっている)

→複雑な厳密計算を効率よく近似したい (最近増加)

→例外が多く、制御ルールを手動で列挙しきれない

何ができれば、タスクとして成功なのだろうか？

→今まで人手で行ってきた作業の一部or全部を代替する

→新しく入手したデータに対しても性能を発揮する

例：スパムフィルタ

- タスク：メールの文章からスパムかスパムでないかを判定したい (2値分類問題)
  - 入力x：本文中に出現する単語の集合
  - 出力t：スパムである ($t = 1$) かない ($t = 0$) か

訓練データ (学習に使うデータ)

文章 (出現単語)	正解
当社の新しい商品を買いませんか？今なら25%OFF！ (当社 新しい 商品 買う 今 25% OFF)	1 (スパム)
山田です。明日のミーティングは12時からにしましょう。 (山田 明日 ミーティング 12時)	0 (正常)
あなたのPCは危険です！添付のマニュアルに従ってください！ (あなた PC 危険 添付 マニュアル 従う)	1 (スパム)

例：スパムフィルタ

文章 (出現単語)	正解
当社の新しい商品を買いませんか？今なら25%OFF！ (当社 新しい 商品 買う 今 25% OFF)	1 (スパム)
山田です。明日の会議は12時からにしましょう。 (山田 明日 会議 12時)	0 (正常)
あなたのPCは危険です！添付のマニュアルに従ってください！ (あなた PC 危険 添付 マニュアル 従う)	1 (スパム)

そこで、論理式で識別器を作成しました：

IF (NOT 会議 OR 商品 OR 添付) THEN 1 ELSE 0

上のモデルに従えば、正解率 (accuracy) は3/3=100%だ！

→このモデルは実際に使えるだろうか？

そんなことはない

新しくメールが届きました：

文章 (出現単語)	予測
山田です。議事録を添付しましたのでご活用ください。 (山田 議事録 添付 する 活用)	1 (スパム)
無駄な会議に消耗していませんか？セミナーを受講して業務改善しましょう！ (無駄 会議 消耗 する セミナー 受講 業務改善)	0 (正常)

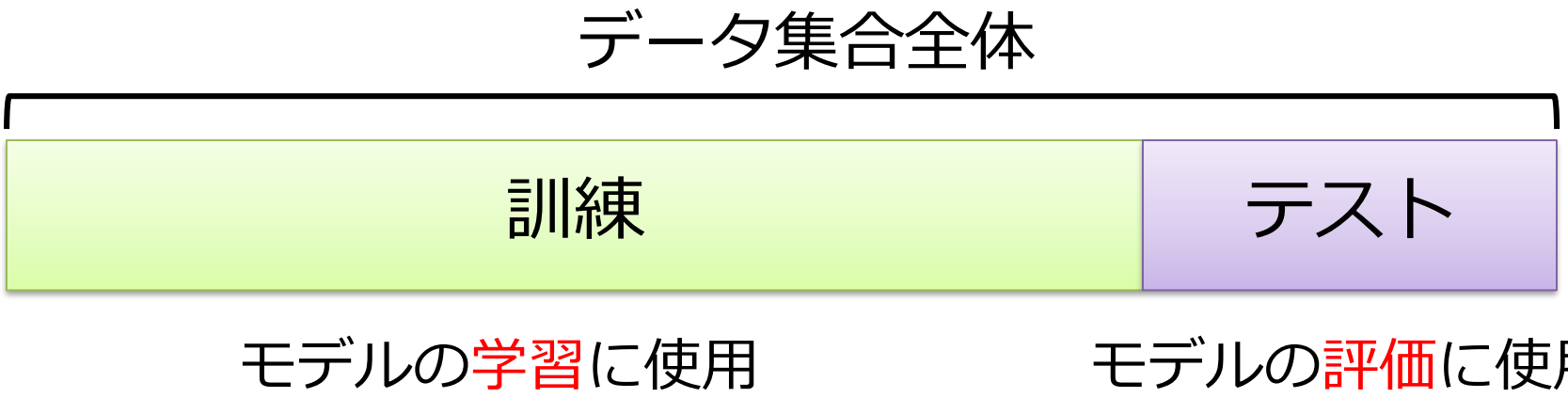
正解率100%の識別器は全然使えなかった・・・

→なぜだろう？

作成したモデルは訓練データによく適合したが、新しいデータ (真のデータ分布) に対しては全く適合しなかった

→訓練データに対する正解率は使えることを意味しない

テストデータへの分割



- 訓練データの誤差 (予測と正解の乖離度) を訓練誤差 (training error)、テストデータの誤差をテスト誤差 (test error) と呼ぶ
- 機械学習においては、(無限量の)未知のデータに対する誤差＝汎化誤差 (generalization error)※ を小さくすることが最終的な目標となり、テスト誤差を汎化誤差の近似とみなして評価を行う

※汎化誤差の定義は省略 (統計的学習の理論としては渡辺澄夫『ベイズ統計の理論と方法』が読み物として面白いですが、講師も数理的な議論は追えません)、その他おすすめ http://ibisml.org/archive/ibis2012/ibis2012-suzuki.pdf

誤差関数の例

- 平均二乗誤差 (回帰問題)

$\|\boldsymbol{x}\|$はL2ノルム $\sqrt{|x_1|^2 + \cdots + |x_D|^2}$

$$\frac{1}{2N}\sum_{i=1}^{N}\|\boldsymbol{y}_i - \boldsymbol{t}_i\|^2$$

- クロスエントロピー誤差 (分類問題)

$$\frac{1}{2}\sum_{i=1}^{N}\sum_{c=1}^{C} t_{ic}\log y_{ic}$$

- KL誤差、Kullback-Leibler divergence

$$KL(q(x)|p(x)) = \int q(x)\log\frac{q(x)}{p(x)}dx$$

$\boldsymbol{y}_i, \boldsymbol{t}_i$は予測・正解ベクトル、$N, C$はサンプル数、クラス数、$p(x), q(x)$は確率分布

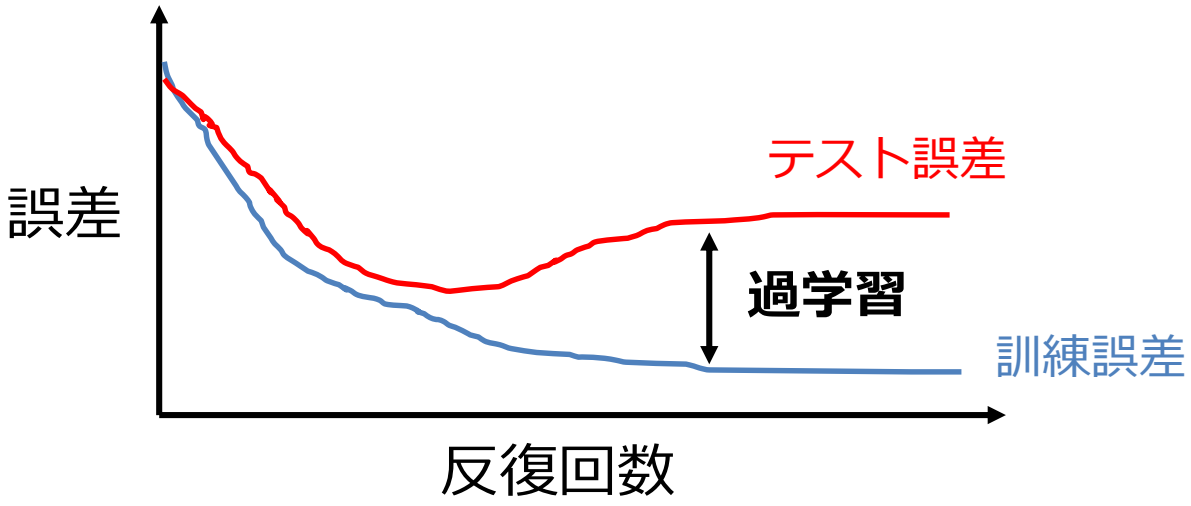
過学習 (overfitting)

あるモデルについて、

訓練誤差 << テスト誤差

であるとき、そのモデルは過学習しているという

例：ニューラルネットの学習

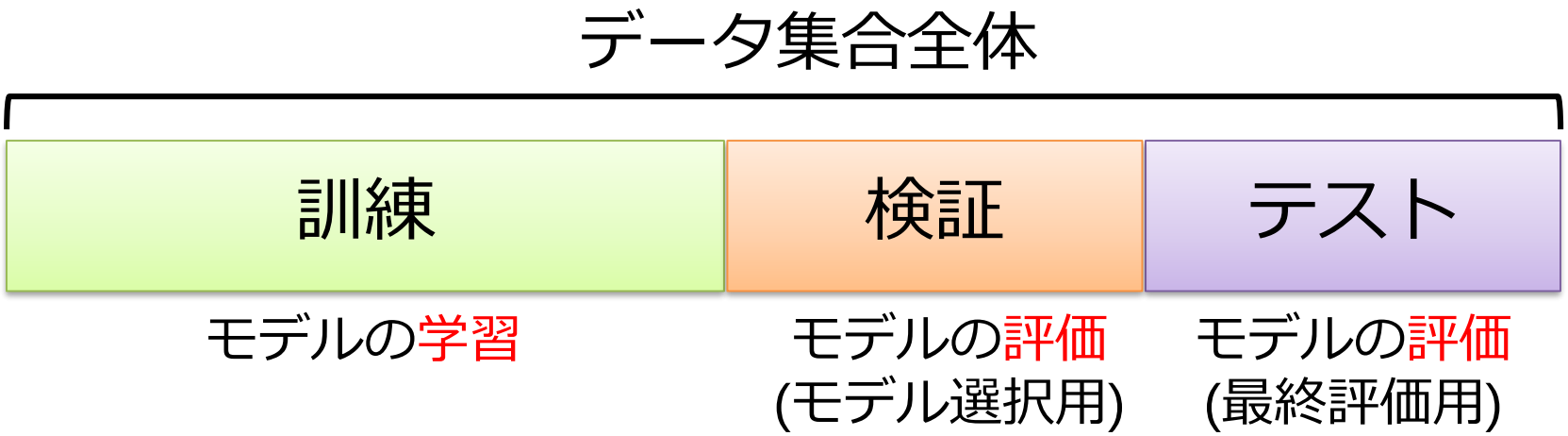


ケーススタディ

- ありがちな例
  - MNISTデータセット (手書き文字分類)
    - 訓練データ60000枚、テストデータ10000枚
  - 3層100ユニットのニューラルネットで学習
    - 訓練データに対する正解率：95%
    - テストデータに対する正解率：93%
  - もっと改善できそうだ。ユニット数を増やしてみよう
    - 訓練データに対する正解率：97%
    - テストデータに対する正解率：96%
  - やった！3%改善したぞ！

これはルール違反。何が問題だろうか？

Hold-out validation



- テストデータはモデル選択に使ってはいけない！
  - モデル選択：学習器の選択、パラメータの調整など
  - 評価した時点で未知であるべきテストデータを間接的に使っている->テストデータに「適合」している
- そこで、別に検証 (バリデーション) データを用意し、訓練誤差と検証誤差とを比較してモデル選択を行う
- 訓練:検証:テストの割合は60:20:20程度が一般的
  - データの総量が少ない場合、後述の交差検証を使用

テストデータの取り扱い

- テストデータは未知のデータとみなすべきであるため、理想的には
  - データを収集した時点でテストデータを分割して、その中身も見ない
  - テストデータを使って評価するのは、モデル選択がすべて終了してからの1回のみ
- 自分のデータセットを用意する場合でも、自身のモデル選択 (特徴量の選択・学習係数・イテレーション数、etc.) にテストデータの影響が一切入らないようにする必要がある
  - さもなければ、テスト誤差≒汎化誤差の仮定が崩れてしまう

テストデータの取り扱い

kaggle

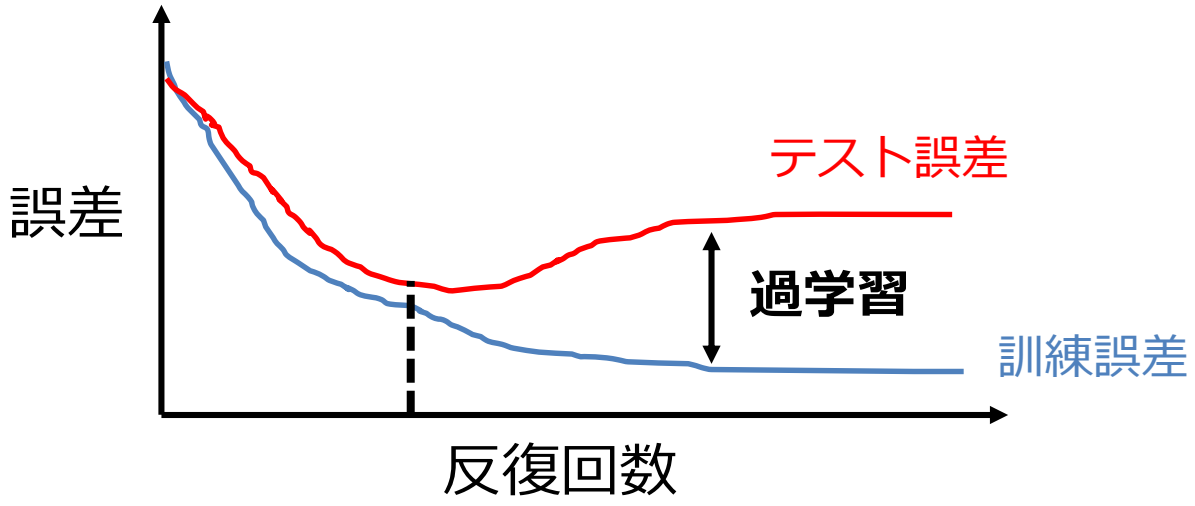
- データ分析コンペティションのKaggleでは、
  - 複数回評価可能なPublic leaderboard
    - 検証誤差に対応
  - 最終成績を決めるPrivate leaderboard
    - テスト誤差に対応

の2つの順位表があり、テストデータは未公開

- Public leaderboardで高順位でも、最終順位が大幅に落ちることは少なくない (検証誤差に過適合した場合)

参考：The Dangers of Overfitting or How to Drop 50 spots in 1 minute

考えてみよう



Q: あるニューラルネットを学習させ、訓練誤差とテスト誤差をモニタリングしたところ、上記のような結果が得られたので、点線に示す地点のモデルを最終モデルに選択した。これは誤った方法であるが、なぜ誤っているのか、どう改善すればよいかを答えよ。

Smerity
@Smerity

フォローする

Researchers: at no time should you ever produce or should I ever see a graph with an axis label of "Test X". You're either cheating during your research or allowing others to cheat after.

英語から翻訳

8:46 - 2017年12月3日 場所: San Francisco, CA

30件のリツイート **108**件のいいね

交差検証 (Cross Validation, CV)

- テストデータの量が十分であるならば、Hold-out validationでのテスト誤差は汎化誤差を十分に近似している (はず)
- しかし、データ総量が少ない場合、訓練データの割合をなるべく多くして、テストデータの割合を小さくしたい。しかし…
  - 問題：検証誤差が汎化誤差の正確な見積もりでなくなる
- そこで、データをランダムなk個のfoldに分割し、残りのk-1個のfoldを訓練データとしたモデルで検証誤差を計算
- k < N個に分割する場合k-fold CVと呼び、N個に分割する (検証データは各1個) 場合Leave-one-out CV (LOOCV)と呼ぶ

交差検証 (Cross Validation, CV)

- (別々に学習した) テスト誤差の平均をとることで、データ集合全体に対するテスト誤差を産出
- **メリット**
  - 汎化誤差の見積もりがより正確になる
  - テスト誤差の評価で訓練データを多く使用できる
- **デメリット**
  - 計算量が増大する (特に深層学習では深刻)
- 一般的に、kは5~10程度で良い

パラメータ選択のための交差検証

- 実は、hold-out validationの”validation”とcross validationの”validation”では意味が異なる
  - 前者はモデル選択のための検証
  - 後者は汎化誤差見積もりのための検証
- 多くの教科書や解説でも明確に扱われないため、他の教科書を見る際は注意が必要
- 一方、モデル選択の場面においても交差検証を使うことができる (inner cross validation)
  - ただし、呼称は必ずしも普及しておらず、2つの交差検証が混ざって使われていることに注意

Inner Cross Validation

(もちろん、本来の意味での交差検証を併用しても良い)

- 各foldの検証誤差の平均が最も小さいモデルを選択
- (テストデータを除く) 全データを学習に使える
- 最終評価にも交差検証を使ってよい
  - Nested (double) cross validationと呼ばれる
  - ただし、Fold数の2乗のオーダーで計算量が増大するので、通常は用いられない

FAQ

- (本来の意味での) 交差検証では見るべきでないテストデータを訓練データとして使うことを許容しているが、それは良くないのではないか？
  - はい。その通りです。
- 一般的に交差検証は汎化誤差の見積もりとモデル選択のどちらで使われているのか？
  - 基本的には前者 (汎化誤差の正確な見積もり) に使われている (一方、モデル選択については曖昧な論文が多い)

誤った交差検証はしばしば誤った結論を導くため、細心の注意を払いましょう

ケーススタディ：『統計的学習の基礎』7.10.2 (動画 https://www.youtube.com/watch?v=S06JpVoNaA0)
, http://www.uvm.edu/~rsingle/stat295/F17/extra/Cross-Validation_%20The%20Right%20and%20Wrong%20Way.html,
http://www.alfredo.motta.name/cross-validation-done-wrong/,

機械学習アルゴリズムの評価指標

機械学習アルゴリズムの評価指標

- 評価指標にはタスクに応じて様々なものがありますが、2値分類問題に的を絞って代表的な指標を紹介します※
  - 適合率 (Precision)
  - 再現率 (Recall)
  - F値 (F-measure)
  - 正解率 (Accuracy)

※適合率・再現率・F値は本来は情報検索分野の指標だが、現在はより一般的に使われている

2値分類問題

- 2値分類問題 (binary classification problem)
  - 正 (positive) か負 (negative) かを予測
- 例題：犬画像検索システム
  - 犬が含まれる画像に対して正、それ以外に負を返す

- システムA:
  - 50枚ヒットした。全て犬画像で誤りは1つもなかったが、取りこぼした犬の画像が70枚あった。
- システムB:
  - 200枚ヒットした。データ中の全ての犬画像を含んでいたが、80件誤って犬が含まれていると判定した画像があった。

どちらが良いシステムだろうか？

2値分類問題における評価指標

	正解が正	正解が負
予測が正	TP: True Positive “正と予測して正解だった”	FP: False Positive “正と予測したが間違いだった” (実際は負)
予測が負	FN: False Negative “負と予測したが間違いだった” (実際は正)	TN: True Negative “負と予測して正解だった”

適合率 (precision): $\frac{TP}{TP+FP}$ (予測がどれだけ正しいか)

再現率 (recall): $\frac{TP}{TP+FN}$ (正解がどれだけ含まれるか)

F値: $\frac{2 \cdot precision \cdot recall}{precision+recall}$ (適合率と再現率の調和平均)

正解率 (accuracy): $\frac{TP+TN}{TP+FP+TN+FN}$ or $\frac{\text{正解数}}{\text{サンプル数}}$

考えてみよう

- システムA:
  - 50枚ヒットした。全て犬画像で誤りは1つもなかったが、取りこぼした犬の画像が70枚あった。
- システムB:
  - 200枚ヒットした。データ中の全ての犬画像を含んでいたが、80件誤って犬が含まれていると判定した画像があった。

Q: システムA及びBの適合率、再現率、F値を求めよ

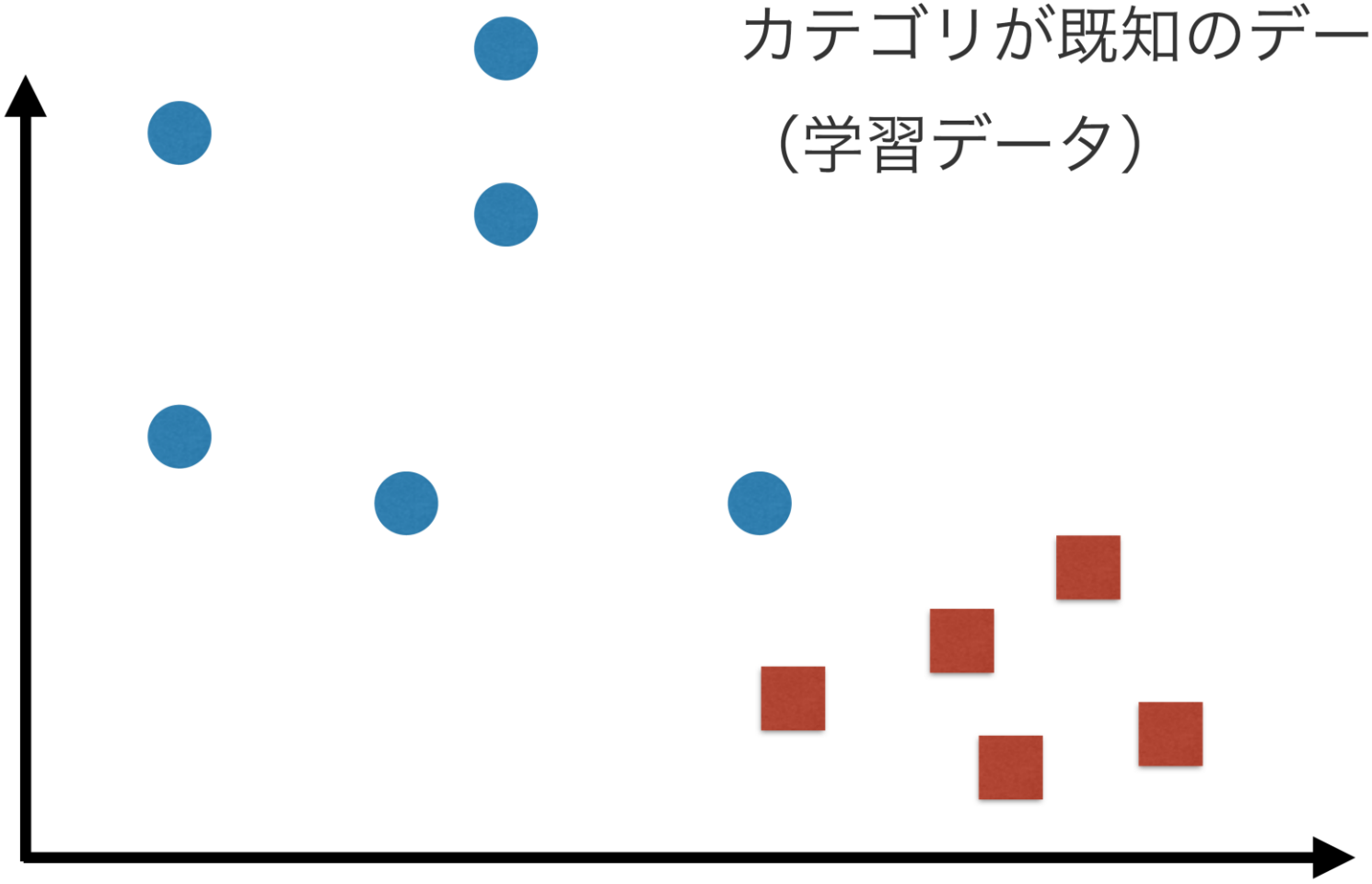
k-近傍法 (k-NN法)

k-近傍法 (k-NN)

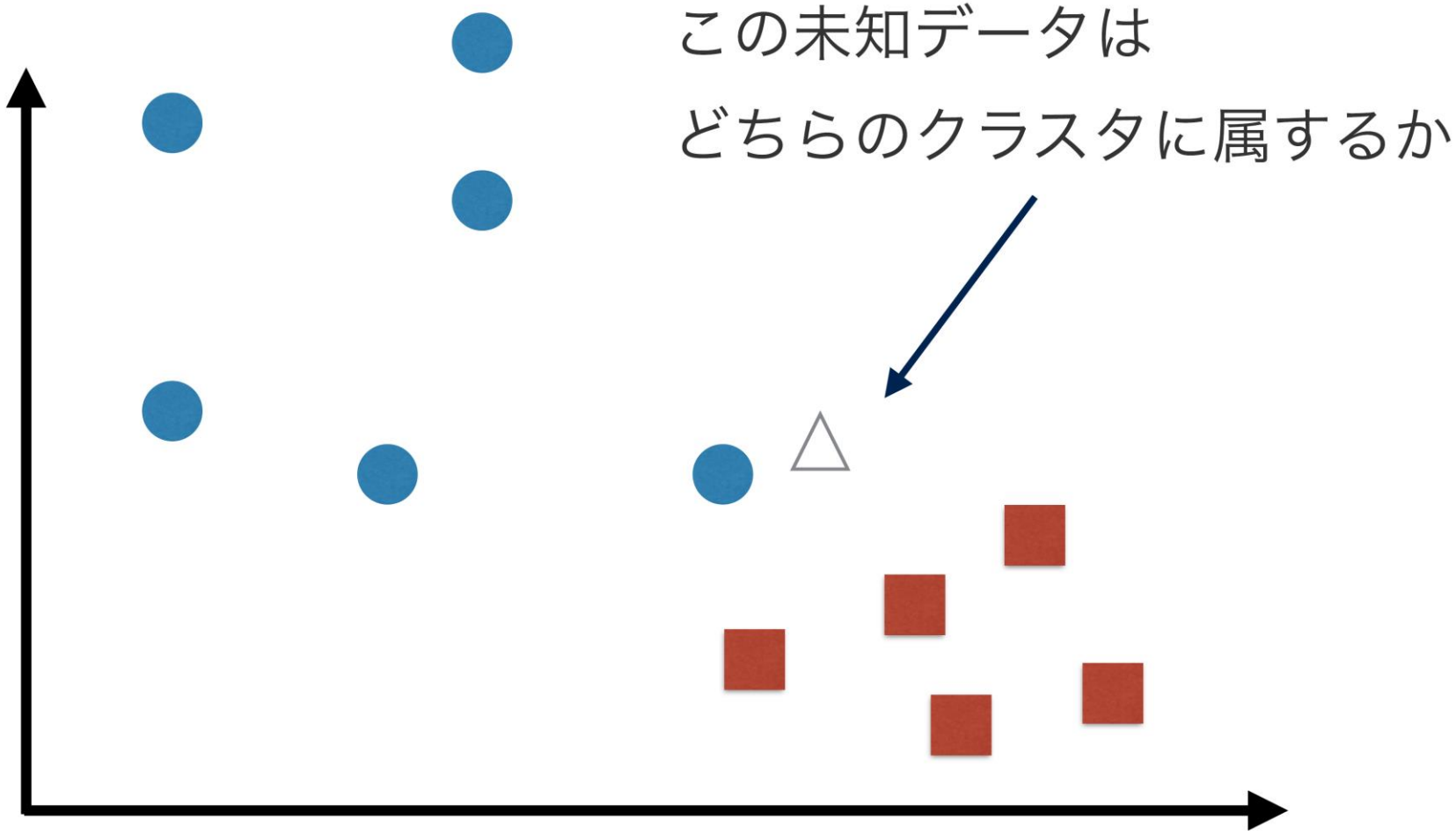
- k-Nearest Neighborsの略
- シンプルで強力な多クラス分類手法
- 新規のデータ点 (ベクトル)と訓練データとの距離を比較して、最も近いk点の所属するクラスの多数決によって分類する

- **利点：**シンプル、十分なデータの下で非線形分類が可能、学習が不要
- **欠点：**距離尺度 (スケールなど) の影響を強く受ける、予測時の計算量が多い、高次元データに弱い (次元の呪い)
- **ハイパーパラメータ：**近傍数k
  - Hold-out validationや (inner) CVで決定

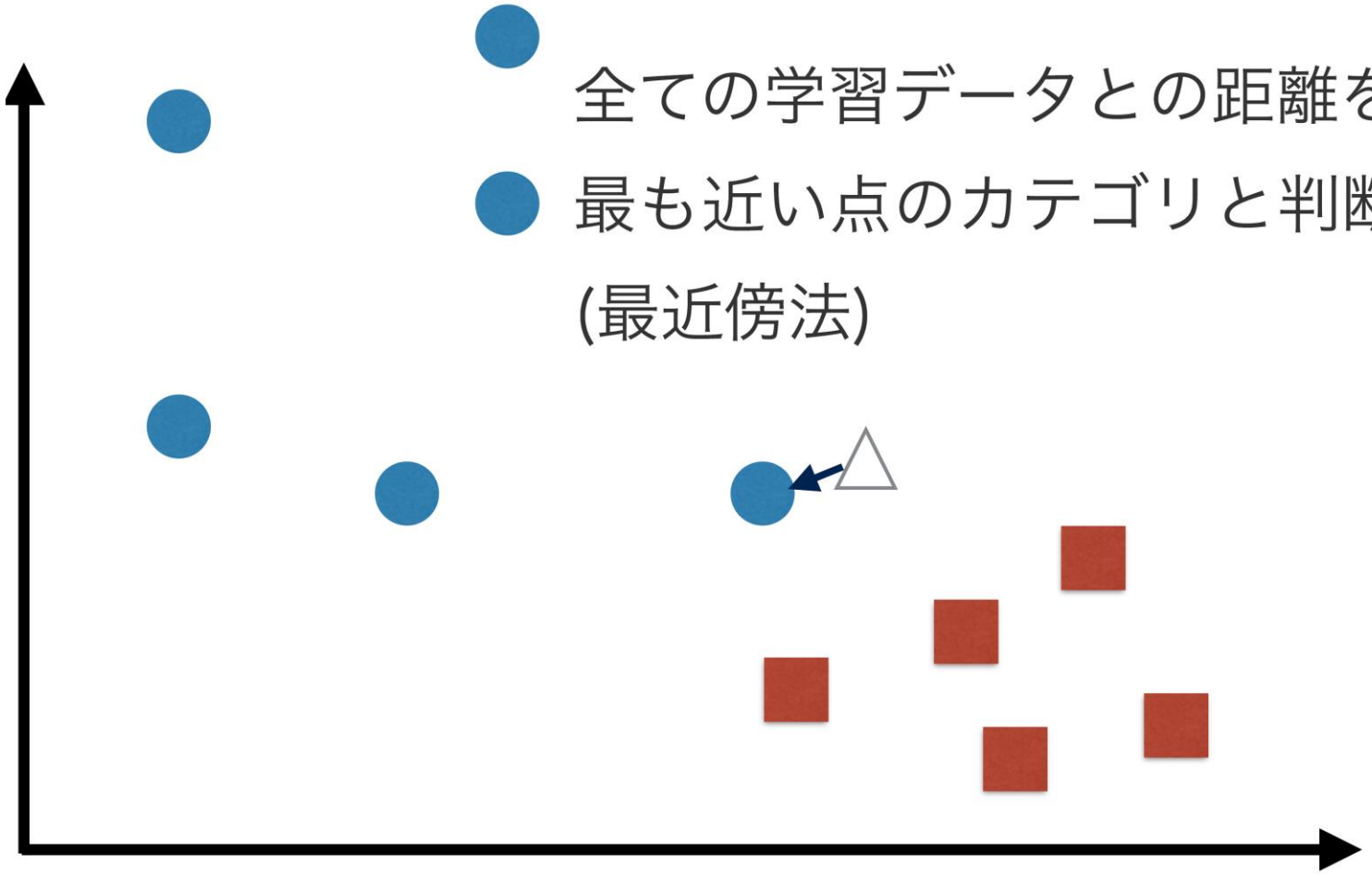
問題設定



問題設定

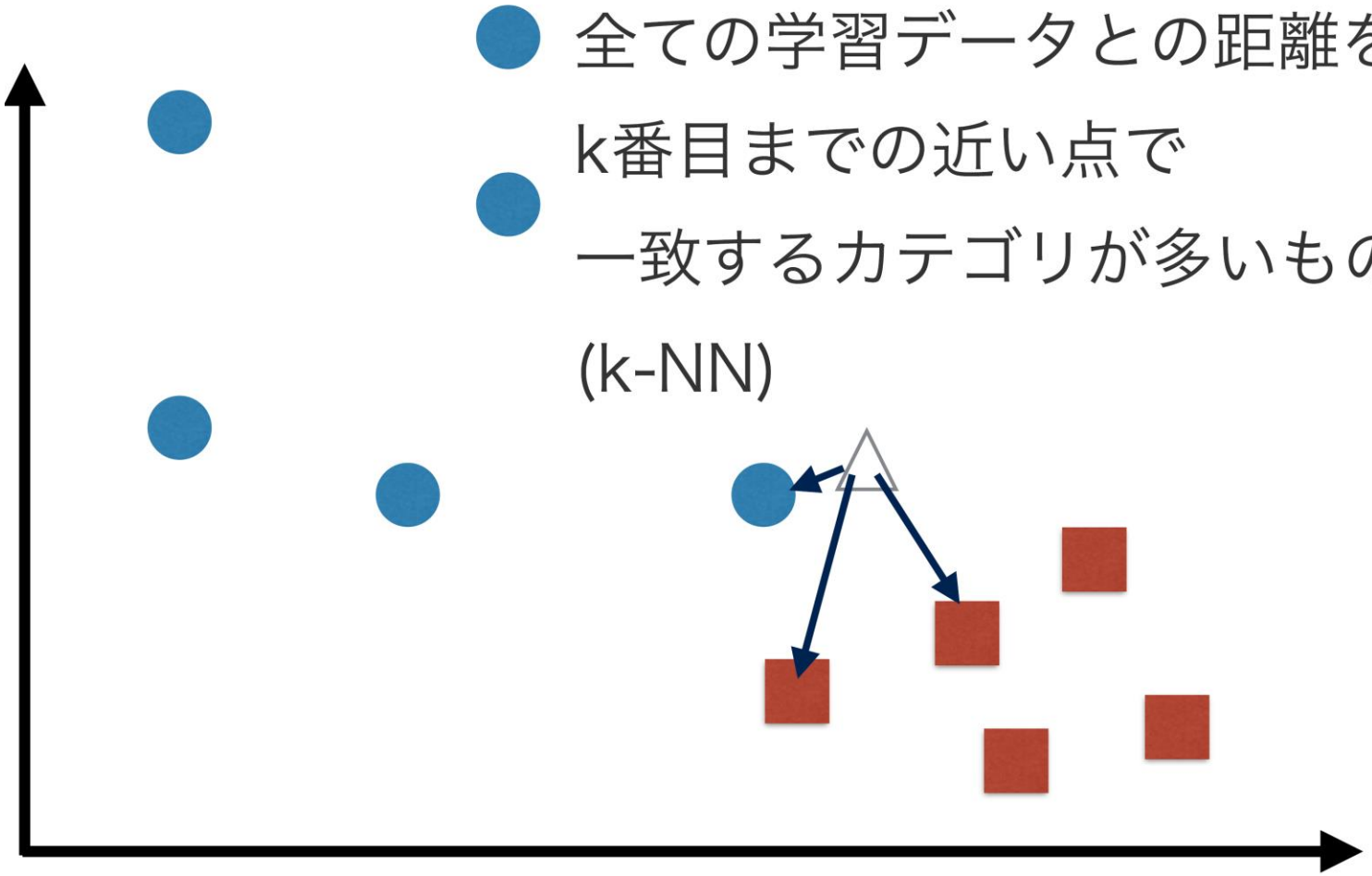


最近傍法 (1-NN)



△は● カテゴリと判定

k-NN法



全ての学習データとの距離を計算し

k番目までの近い点で

一致するカテゴリが多いものと判断

(k-NN)

△は■カテゴリと判定

k-NN法の距離尺度

- 通常はユークリッド距離を用いる

$$d(p,q) = \sqrt{\sum_{i=1}^{D} (q_i - p_i)^2}$$

- K-NN法の性能は距離尺度の設定に強く依存する

Cesarの広告

画素の明るさで比較すると、
左記の人と犬は
距離が近い＝「似ている」
と判定されることになる！
→物体認識の難しさ (第8回)

*k-NN法の計算量

- データ数をN、次元数をD、近傍数をKとする
- 新しいデータ点を分類したい
- 素朴な方法
  1. ある1点 (D次元)との距離計算に$O(D)$
  2. 全データ点 (N)との距離計算に$O(ND)$
  3. 最も距離が短いK個を選び出す: 線形探索で$O(NK)$
  4. 合計計算量は$O(ND + NK)$

*パーセプトロン

(単層) パーセプトロン

- 単一のニューロンを模したモデル (Rosenblatt, 1958)
- 線形分離可能な分類問題に有効
- **モデル**

$$y = \mathrm{sign}\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} + b\right)$$
$$= \mathrm{sign}(w_1 x_1 + \cdots + w_D x_D + b)$$

ただし、$\mathrm{sign}(x) = 1 \text{ if } x > 0 \text{ else } -1$

- **学習 (更新則)**
  - 間違った ($y_n \neq t_n$) データ点に対してのみ

$$\boldsymbol{w}^{\tau+1} = \boldsymbol{w}^{\tau} + \eta \boldsymbol{x}_n t_n$$
$$b^{\tau+1} = b^{\tau} + \eta t_n$$

ただし、τは反復回数、ηは学習率、t_nは正解クラス (1 or -1)

Further reading

- 統計的推論と汎化誤差
  - 渡辺澄夫『ベイズ統計の理論と方法』
  - 渡邊澄夫『ベイズ推論』 (講演資料)
  - Hastie, Tibshirani, Friredman『統計的学習の基礎』(英語版PDF)
- 機械学習アルゴリズムの評価
  - Evaluating Machine Learning Methods
- 今回扱えなかった「次元の呪い」に関するカジュアルな解説
  - 次元の呪い、あるいは「サクサクメロンパン問題」
  - コサイン類似度が高いベクトルはどれくらい似ているか

次回 (線形回帰)
不安な方は
線形代数の復習をよろしく
お願いします

Appendix
数式 v.s. 実装

数式の表記

- 筆者が知りうる限り以下のような慣習が一般的です：
  - スカラ (単一の値): x (小文字通常)
  - ベクトル: $\boldsymbol{x} = (x_1, \ldots, x_D)^T$ (小文字太字、縦ベクトル)
  - 行列：$X = \begin{pmatrix} \boldsymbol{x}_1^T \\ \vdots \\ \boldsymbol{x}_N^T \end{pmatrix} = \begin{pmatrix} x_{11} & \cdots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{ND} \end{pmatrix}$ (大文字通常)
- Numpyに関しては、行ベクトルと列ベクトルの区別を通常しないため (1次元配列で充分)、数式との整合性が必ずしも取れない場合があります
  - 常にどの形状の配列を扱っているのかを意識しましょう

2値分類問題

- 2値分類問題：入力xが0か1かを予測する
- 通常は、予測が正である確率 $[0, 1]$ を出力する
- 入力が1サンプル (x: D次元ベクトル) の場合

スカラ　$(D,)$　$(D,)$

$$y = \boldsymbol{x}^T \boldsymbol{w} = (x_1 \quad \cdots \quad x_D) \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_D \end{pmatrix}$$

y = np.dot(x, w)

- 入力がNサンプル (x: (N, D)の行列) の場合

$(N,)$　(N, D)　$(D,)$

$$\boldsymbol{y} = X\boldsymbol{w} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix} = \begin{pmatrix} x_{11} & \cdots & x_{1D} \\ x_{21} & \ddots & x_{2D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{ND} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_D \end{pmatrix}$$

y = np.dot(X, w)

多値分類問題

- 多値分類問題：入力xがクラスc $(\mathrm{c} = 1, \ldots, C)$のどれに所属するかを予測する
- 確率予測の場合、各クラスに属する確率を出力する
- 1サンプルの場合　y = np.dot(x, W)

$$\underset{(C,)}{\boldsymbol{y}^T} = \underset{(D,)}{\boldsymbol{x}^T} \underset{(D,C)}{W} = \begin{pmatrix} y_1 & \cdots & y_C \end{pmatrix} = \begin{pmatrix} x_1 & \cdots & x_D \end{pmatrix} \begin{pmatrix} w_{11} & \cdots & w_{DC} \\ \vdots & \ddots & \vdots \\ w_{D1} & \cdots & w_{DC} \end{pmatrix}$$

- Nサンプルの場合　Y = np.dot(X, W)

$$\underset{(N,C)}{Y} = \underset{(N,D)}{X} \underset{(D,C)}{W} = \begin{pmatrix} y_{11} & \cdots & y_{1C} \\ \vdots & \ddots & \vdots \\ y_{N1} & \cdots & y_{NC} \end{pmatrix} = \begin{pmatrix} x_{11} & \cdots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{ND} \end{pmatrix} \begin{pmatrix} w_{11} & \cdots & w_{DC} \\ \vdots & \ddots & \vdots \\ w_{D1} & \cdots & w_{DC} \end{pmatrix}$$

N行C列の行列
y_{ij}はiサンプル目がクラスjに所属する確率を表す

予測クラス
各要素には$0, \ldots, c-1$のいずれかの値が入る