

パターンテンプレート

1.2 LOGISTIC REGRESSION

デザインパターン

- パターン名: Logistic Regressionパターン
- 分類名: 「分類」(線形分離可能な教師あり学習)
- 目的: 既知の量的変数から未知の質的変数を予測する

■ロジスティック回帰が適している例

- 未知の質的変数(=目的変数)が、相反関係にあるもの(ON・OFF、あり・なし、良品・不良品、など)

■用語

量的変数: データが数値(身長/体重など)で示されるもの

質的変数: データがカテゴリ(性別など)で示されるもの

問題: Titanic



- 問題: Titanic: Machine Learning from Disaster
 - <https://www.kaggle.com/c/titanic>
 - 入門用としてほぼ必ず取り上げられる題材
- 内容:
 - 豪華客船「タイタニック号」の沈没事故の生存者を予測する
- 予測値の評価
 - 正解率: 全データのうち、正しく予測("1", "0" いずれも)できた割合
 - 精度: "1"と予測したデータのうち、実際に"1"だった割合
 - 再現率: 実際に"1"であるデータのうち、"1"と予測できた割合
 - F値: 精度と再現率の調和平均(精度と再現率はトレードオフの関係にあるため、バランスをとった指標として使われる)

適用条件

- 目的変数が相反関係にある二値（ON・OFF など）に分類される場合に、ロジスティック回帰が適用可能である。

適用手順

1. データを読み込む
2. データを観察する(図表を使う)
3. データを整形する(欠損値、カテゴリ属性)
4. ロジスティック回帰 分析器に訓練データを登録する
5. ロジスティック回帰 分析器ハイパラメータを決定する
6. 訓練データを用いてロジスティック回帰 分析器の学習モデルを作成する
7. 学習モデルのスコア値(正解率など)を算出する。
8. 学習後モデルに評価データを適用し目的変数の予測を行う

実装上の注意点 (scikit-learnを使う前提)

＜ロジスティック回帰に限らず、一般的な注意事項＞

■カテゴリ属性のOne Hot Encoding化の方法

- 属性のユニーク値が訓練/評価データで同じ場合 (★本サンプルではこちら)
 - 訓練/評価データに対して、Pandasの `get_dummies` を実行する。(追加されるダミー列が、訓練データと評価データで一致する為問題なし)
- 属性のユニーク値が訓練/評価データで同じでない場合
 - 訓練/評価データに追加されるダミー列が一致するように実装する。

■多重共線性

- 1つの属性が2列以上にダミー化された場合、任意の1列を消すことで、モデルの予測精度向上が期待できる。

サンプルコード





- Github
 - https://github.com/topse2018-kaggle/team/tree/master/oouchi/1.2_LogisticRegression




ファイル名	説明
kaggle-titanic-lg.ipynb	サンプルコード (IPython)
kaggle-titanic-lg.html	サンプルコード (結果付き) のHTML出力
train.csv	訓練データ
test.csv	評価データ
submission.csv	Kaggle提出用データ
1.2_RogisticRegression.pptx	本ファイル

適用結果 (2018/11/25時点)

■ チュートリアル (Titanic) コンペの結果

- Score: 0.76076
- 順位: 8230/10354位 (下位 約20%) → スキルアップが必要！

8227	new	Takuma Kawahara		0.76076
8228	new	Jihye Seung		0.76076
8229	new	skenmrc		0.76076
8230	▼ 494	Kazunori Oouchi		0.76076

#	△1w	Team Name	Kernel	Team Members	Score
1	—	ChristinaPR			1.00000
2	—	Heitor Jurkovich			1.00000
3	—	sorry			1.00000

ちなみに、1位(19名)のスコアは1.00です。

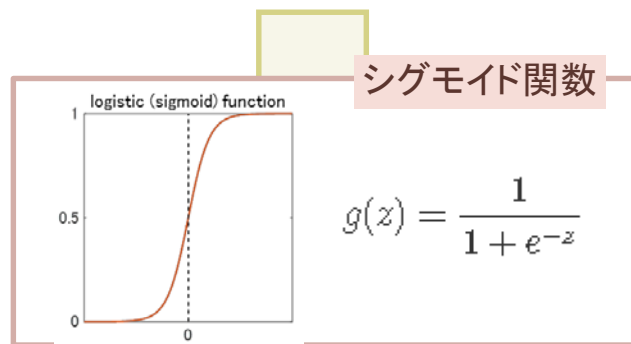
理論的背景(目次)

1. 考え方
2. 目的関数
3. 最急降下法
4. コスト関数

理論的背景(1. 考え方)

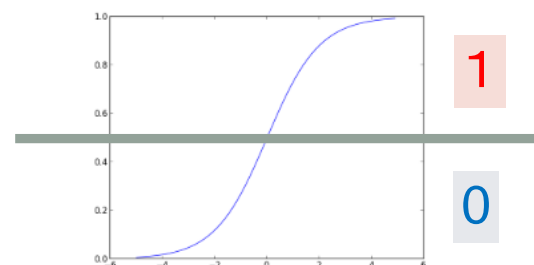
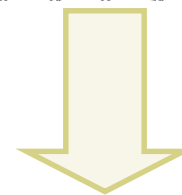
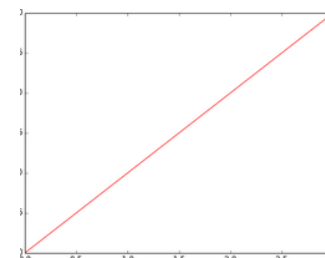
- 目的変数の予測値 Y を表す回帰式(仮定関数 $h_{\theta}(x)$)にシグモイド関数を通すことで、予測値を二値(1-0)として扱うことが可能になる。

$$h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1 + \dots + \theta_n x_n = \theta^T x$$



$$h_{\theta}(x) = \frac{1}{1 + e^{-\theta^T x}}$$

- $h_{\theta}(x) \geq 0.5$ なら $y = 1$
- $h_{\theta}(x) \leq 0.5$ なら $y = 0$



二値扱い
が可能

理論的背景(2. 目的関数)

- 目的関数 $J(\theta)$ とは、仮定関数 $h_{\theta}(x)$ と実際のデータ Y との差を関数で示したもの。
- 差を最小にする重みづけパラメータ θ を求めることがねらい。

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

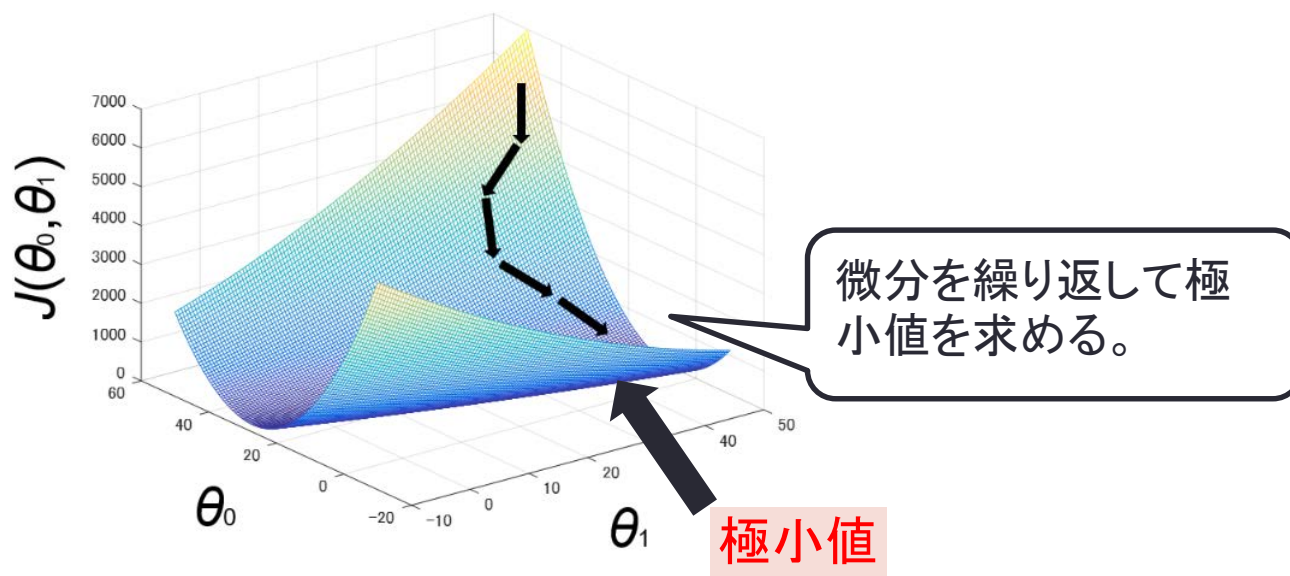
ロジスティック回帰の特徴を踏まえ、
最終的に以下のように変形

$$J(\theta) = -\frac{1}{m} \left[\sum_{i=1}^m y^{(i)} \log(h_{\theta}(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)})) \right]$$

m : データのサンプル数 (= 行数)

理論的背景(3. 最急降下法)

- 目的関数 $J(\theta)$ は下に凸な曲線であり、微分によって $J(\theta)$ の 極小値(最も凸な部分)を求める。
- 極小値を満たす $\theta(=\theta^0, \theta^1, \dots)$ が説明変数の係数値となる。



理論的背景(4. コスト関数)

本ページを削除し、別パターンとして作成する予定(詳細は11月29日に確認)

- 重みづけパラメータを増やしすぎることによる過学習を防ぐため、目的関数 $J(\theta)$ にコスト関数(点線で囲んだ部分)を追加する。

$$J(\theta) = \frac{1}{2m} \left[\sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^n \theta_j^2 \right]$$

コスト関数

- scikit-learnのLogisticRegressionのパラメータ C は「 $1/\lambda$ 」と同じ。
- 考え方はL2正則化と同じ(本ファイルでは詳細説明を省略する)

出典

- ロジスティック回帰分析
 - <https://qiita.com/fujin/items/7f0a7b6fc8fb662f510d>
- Coursera Machine Learning (3): ロジスティック回帰、正則化
 - <https://qiita.com/katsu1110/items/e4ef613559f02f183af5>

関連するパターン

- 線形分離可能な観点

- 線形回帰 LinearRegression
- Lasso回帰(L1正則化) Lasso
- Lasso回帰(L2正則化) Ridge

- 「分類」観点

- 線形サポートベクターマシン LinearSVC
- 決定木 DecisionTreeClassifier
- サポートベクターマシン SVC
- ニューラルネットワーク ※ パーセプトロン、CNN、RNNなど
- K近傍法 KNeighborsClassifier

※ TensorFlow/Keras のみ