線形回帰モデル

教師あり学習であり、回帰問題を解くための機械学習の一つ。

回帰問題とは、ある入力から出力を予測する問題のこと。

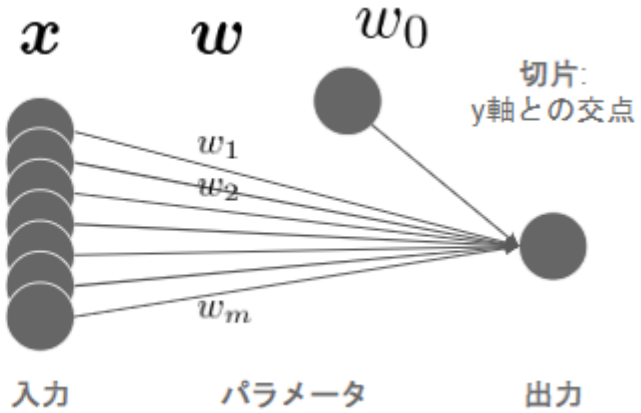
入力と m 次元パラメータの線形結合を出力する。

線形結合とは、入力とパラメータの内積のこと。

線形結合

$$\hat{y} = \boldsymbol{w}^T \boldsymbol{x} + w_0 = \sum_{j=1}^{m} w_j x_j + w_0$$

パラメータは未知



※予測値にはハット”^”を付けるのが慣例。

最小二乗法

線形回帰モデルのパラメータを推定するのに利用する学習データの平均二乗誤差を最小とするパラメータを探索する方法。

平均二乗誤差の最小とは、その勾配（=微分した値）が 0 になる点を求めればよい。

$MSE_{train} = \frac{1}{n_{train}} \sum_{i=1}^{n_{train}} (\hat{y}_i^{(train)} - y_i^{(train)})^2$ より、$\frac{\partial}{\partial w} MSE_{train} = 0$ を求めると以下となる。

回帰係数

$$\hat{\boldsymbol{w}} = (X^{(\mathrm{train})T} X^{(\mathrm{train})})^{-1} X^{(\mathrm{train})T} \boldsymbol{y}^{(\mathrm{train})}$$

予測値

$$\underset{n_{new} \times m+1}{\hat{\boldsymbol{y}} = X} (X^{(train)T} X^{(train)})^{-1} X^{(train)T} \boldsymbol{y}^{(train)}$$

<実装演習>

skl_regression.ipynb

np_regression.ipynb

非線形回帰モデル

複雑な非線形構造を内在する現象に対して、非線形回帰モデリングを実施する手法。

データの構造を線形で捉えることができるケースは限られており、非線形な構造を捉えられる仕組みが必要なため利用される。

基底展開法

回帰関数として、基底関数と呼ばれる既知の非線形関数とパラメータベクトルの線形結合を使用する手法。

未知パラメータは線形回帰モデルと同様に最小二乗法や最尤法により推定する。

基底関数には多項式関数やガウス型基底関数、スプライン関数/B スプライン関数などが利用される。

$$y_i = f(\boldsymbol{x}_i) + \varepsilon_i \qquad y_i = w_0 + \sum_{j=1}^{m} w_j \phi_j(\boldsymbol{x}_i) + \varepsilon_i$$

多項式の場合

$$\phi_j = x^j$$

ガウス型基底関数の場合

$$\phi_j(x) = \exp\left\{\frac{(x-\mu_j)^2}{2h_j}\right\}$$

未学習の対策

学習データに対して、十分に小さな誤差が得られていない場合は未学習であるため、以下の対策を行う。

・モデルの表現力が低いため、表現力の高いモデルを利用する。

過学習の対策

小さな誤差は得られたが、テスト集合誤差との差が大きい場合は過学習が発生しているため、以下のような対策を行う。

1. 学習データの数を増やす
2. 不要な基底関数(変数)を削除して表現力を抑止
   基底関数の数、位置やバンド幅によりモデルの複雑さは変化するため、多くの基底関数を用意してしまうと過学習が起こるため、適切な基底関数を用意する。
3. 正則化法を利用して表現力を抑制
   正則化法(罰則化法)とは、モデルの複雑さに伴って、その値が大きくなる正則化項(罰則項)を課した関数を最小化する手法。
   正則化項(罰則項)は、形状によっていくつもの種類があり、それぞれ推定量の性質が異なる。

$$S_\gamma = (\boldsymbol{y} - \overset{n \times \kappa}{\Phi}\boldsymbol{w})^T(\boldsymbol{y} - \Phi\boldsymbol{w}) + \gamma R(\boldsymbol{w}) \quad \gamma(>0)$$

基底関数の数(k)が増加するとパラメータが増加し、
残差は減少 (モデルが複雑化)

モデルの複雑さに伴う罰則

汎化性能

学習に使用した入力だけでなく、これまで見たことのない新たな入力に対する予測性能のこと。

汎化誤差(=テスト誤差)が小さいものほど性能が良いモデルと判断できる。

$$MSE_{test} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} (\hat{y}_i^{test} - y_i^{(test)})^2$$

モデルの性能評価に以下のような手法を用いる。

ホールドアウト法

有限のデータを学習用とテスト用の 2 つに分割し、予測精度や誤り率を推定するために使用する。

手元に大量にデータがある場合を除いて、良い性能評価を行うことができない。

クロスバリデーション(交差検証)

データを学習用と評価用に分割して使用する。

K-分割交差検証法というものがよく利用されており、これはデータを K 個に分割してそのうちの 1 つをテストデータ、残りを学習用データとして評価を行い、これを K 回行う(=分割した K 個の全てがテストデータとして利用されるまで)学習を行い、その平均をとる手法。

グリッドサーチ

全てのチューニングパラメータの組み合わせで評価値を算出し、最も良い評価値を持つチューニングパラメータを持つ組み合わせを採用する。

<実装演習>

skl_nonlinear regression.ipynb

ロジスティック回帰モデル

分類問題を解くための教師あり機械学習モデル(教師データから学習)。

入力と m 次元パラメータの線形結合をシグモイド関数により入力し、出力は y=1 になる確率の値となる。

シグモイド関数

入力は実数で出力は必ず 0〜1 の値。

クラス 1 に分類される確率を表現している単調増加関数。

$$\sigma(x) = \frac{1}{1 + \exp(-ax)}$$

a を増加させると、x=0 付近での曲線の勾配が増加する。

a を極めて大きくすると、単位ステップ関数に近づく。

微分はシグモイド関数自身で表現することができる。

$$\frac{\partial \sigma(x)}{\partial x} = a\sigma(x)(1 - \sigma(x))$$

尤度関数

データは固定し、パラメータを変化させる。

尤度関数を最大化するようなパラメータを選ぶ推定方法を最尤推定という。

$$P(y_1, y_2, \cdots, y_n | w_0, w_1, \cdots, w_m) = \prod_{i=1}^{n} p_i^{y_i} (1 - p_i)^{1-y_i}$$

$$= \prod_{i=1}^{n} \sigma(\boldsymbol{w}^T \boldsymbol{x}_i)^{y_i} (1 - \sigma(\boldsymbol{w}^T \boldsymbol{x}_i))^{1-y_i}$$

$$= L(\boldsymbol{w})$$

尤度関数は
パラメータのみに依存する関数

尤度関数を最大化するパラメータについては以下により探すことができる。

対数をとることで微分の計算を簡単にする。

対数尤度関数が最大になる点と尤度関数が最大になる点は同じである。

尤度関数にマイナスをかけたものを最小化し、最小 2 乗法の最小と合わせる。

$$E(w_0, w_1, \cdots, w_m) = -\log L(w_0, w_1, \cdots, w_m)$$

$$= -\sum_{i=1}^{n} \{y_i \log p_i + (1 - y_i) \log(1 - p_i)\}$$

勾配降下法

対数尤度関数を微分して 0 になる値を求める必要があるが、解析的にこの値を求めることが困難なため利用する手法。

$$\boldsymbol{w}(k+1) = \boldsymbol{w}^k - \eta \frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}}$$

対数尤度関数を係数とバイアスに関して微分する。

$$
\begin{aligned}
\frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}} &= \sum_{i=1}^{n} \frac{\partial E_i(\boldsymbol{w})}{\partial p_i} \frac{\partial p_i}{\partial \boldsymbol{w}} \\
&= \sum_{i=1}^{n} \left(\frac{y_i}{p_i} - \frac{1-y_i}{1-p_i} \right) \frac{\partial p_i}{\partial \boldsymbol{w}} \\
&= \sum_{i=1}^{n} \left(\frac{y_i}{p_i} - \frac{1-y_i}{1-p_i} \right) p_i(1-p_i)\boldsymbol{x}_i \\
&= -\sum_{i=1}^{n} (y_i(1-p_i) - p_i(1-y_i))\boldsymbol{x}_i \\
&= -\sum_{i=1}^{n} (y_i - p_i)\boldsymbol{x}_i
\end{aligned}
$$

$$
\boldsymbol{w}^{(k+1)} = \boldsymbol{w}^{(k)} + \eta \sum_{i=1}^{n} (y_i - p_i)\boldsymbol{x}_i
$$

確率的勾配降下法

勾配降下法では全てのデータに対する和を求める必要があるため、データをランダムに選んでパラメータを更新する手法。

勾配降下法でパラメータを 1 回更新するのと同じ計算量でパラメータを n 回更新できる。

$$
\boldsymbol{w}(k+1) = \boldsymbol{w}^{k} + \eta(y_i - p_i)\boldsymbol{x}_i
$$

再現率(Recall)

本当に Positive なものの中から Positive と予測できる割合。

$$
\frac{TP}{TP + FN}
$$

TP：本当に Positive であり、Positive と予測された数。

FN：本当は Positive だが、Negative と予測された数。

適合率(Precision)

Positive と予測したものから本当に Positive である割合。

$$\frac{TP}{TP+FP}$$

TP：本当に Positive であり、Positive と予測された数。

FP：本当は Negative だが、Positive と予測された数。

F 値

再現率と適合率の調和平均のこと。

再現性と適合率は両方とも高いモデルが理想であるが、両者はトレードオフの関係にあるため、どちらかを小さくするともう片方が大きくなるため、F 値を利用してモデルの最適化を判断する。

<実装演習>

skl_logistic_regression.ipynb

np_logistic_regression.ipynb

主成分分析

主成分分析

多変量データの持つ構造をより少数個の指標に圧縮(次元圧縮)する手法。

・変量の個数を減らすことに伴う、情報の損失はなるべく小さくしたい。

・少数変数を利用した分析や可視化(2 次元・3 次元の場合)が実現可能。

学習データ	$x_i = (x_{i1}, x_{i2}, \cdots, x_{im}) \in \mathbb{R}^m$
平均(ベクトル)	$\bar{x} = \frac{1}{n}\sum_{i-1}^{n} x_i$
データ行列	$\bar{X} = (x_1 - \bar{x}, \cdots, x_{in} - \bar{x})^T \in \mathbb{R}^{m \times n}$
分散共分散行列	$\Sigma = V_{ar}(\bar{X}) = \frac{1}{n}\bar{X}^T\bar{X}$
線形変換後のベクトル	$s_j = (s_{1j}, \cdots, s_{nj})^T = \bar{X}a_j \quad a_j \in \mathbb{R}^m$

ラグランジュ関数を微分することで最適解を求める。

$E(a_j) = a_j^T Var(\bar{X}) a_j - \lambda(a_j^T a_j - 1)$

$$\frac{\partial E(a_j)}{\partial a_j} = 2Var(\bar{X})a_j - 2\lambda a_j = 0 \text{ より} \quad Var(\bar{X})a_j = \lambda a_j$$

寄与率

第 k 主成分の分散の全分散に対する割合(第 k 主成分が持つ情報量の割合)

2 次元のデータを 2 次元の主成分で表示した時、固有値の和と元データの分散が一致する。

第 k 主成分の分散は主成分に対応する固有値。

$$c_k = \frac{\lambda_k}{\sum_{i=1}^{m} \lambda_i}$$

$\sum_{i=1}^{m} \lambda_i$：主成分の総分散

λ_k：第 k 主成分の分散

累積寄与率

第 1-k 主成分まで圧縮した際の情報損失量の割合。

$$r_k = \frac{\sum_{j=1}^{k} \lambda_j}{\sum_{i=1}^{m} \lambda_i}$$

$\sum_{j=1}^{k} \lambda_j$：第 1～k 主成分の分散

<実装演習>

skl_pca.ipynb

np_pca.ipynb

アルゴリズム

k 近傍法(kNN)

教師あり学習の分類問題のための機械学習手法。

近くにあるデータ k 個を選択し、k 個のデータが最も多くのデータが所属しているクラスに該当データは所属するという手法。

パラメータ:k の値により、結果は異なる。

また、k の値が大きいと決定境界は滑らかになる。

k-means 法(k-平均法)

教師なし学習のクラスタリング手法。

与えられたデータを以下のアルゴリズムにより k 個のクラスタに分類する。

<アルゴリズム>

1. 各クラスタ中心の初期値を設定する。
2. 各データ点に対して、各クラスタ中心との距離を計算し、最も距離が近いクラスタを割り当てる。
3. 各クラスタの平均ベクトル(中心)を計算する。
4. 収束するまで 2〜3 の処理を繰り返す。

クラスタ中心の初期値をどこに設定するかにより結果が異なる。

初期値同士が近いとうまくクラスタリングされず、離れているとうまくクラスタリングできる。

<実装演習>

np_knn.ipynb

skl_kmeans.ipynb

np_kmeans.ipynb

サポートベクターマシン(SVM)

教師あり学習のパターン認識モデルの一つ。

2 クラス分類問題の代表的な手法の一つであり、未知データに対して高い予測精度を持つ関数が構築できる。

分類境界線を挟んで 2 つのクラスがどのくらい離れているかをマージンという。

マージンが大きいほど良い分類境界線となるため、マージンを最大化するような分類境界線を探す。

訓練データを完璧に分類できると仮定することをハードマージン、分離できないデータの存在を許容するものをソフトマージンと呼ぶ。

ソフトマージンの場合、マージン内に入るデータや誤分類されたデータに対する誤差を表すスラック変数が存在する。

<実装演習>

np_svm.ipynb