

Workout Report

最終報告会

津野 龍

発表の流れ

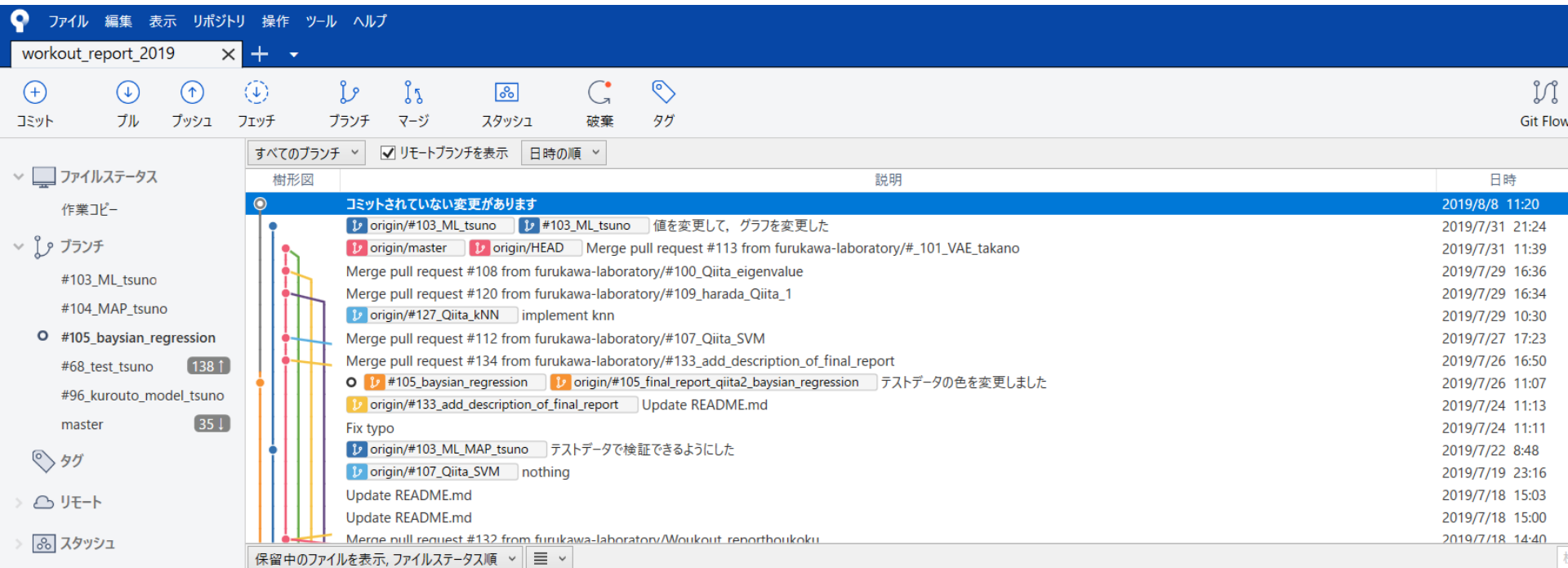
- 約3か月間でやってきたことをダイジェスト的に発表する
- 頑張ったところを詳しく発表する

三ヶ月で獲得したスキルセット

	ビギナー	常人	玄人	達人	神
準備	Python, Git, Github ✓	/	/	/	/
線形代数	ベクトル空間 ✓	線形変換 ✓	特異値分解と主成分分析	ガウス過程	情報幾何
ベイズ (確率)	ベイズルール ✓	点推定 ✓	分布推定 ✓	ガウス過程	モデル選択
前処理	正規化 ✓	データセット ✓	生データ	特徴量エンジニアリング	表現学習
モデル	実装済みモデルの利用 ✓	kerasによる実装 ✓	スクラッチ実装 ✓	問題設定 (教師あり)	問題設定 (教師なし)
検証	描画 ✓	評価指標 (教師あり) ✓	評価指標 (教師なし)	実験デザイン (教師あり)	実験デザイン (教師なし)

- 線形代数 → 常人まで
- ベイズ → 玄人まで
- 前処理 → 常人まで
- モデル → 玄人 (未完成) まで
- 検証 → 常人まで

Git & Github
















The screenshot shows the Git GUI interface. The top bar includes a search field with 'workout_report_2019' and a menu with options like 'ファイル', '編集', '表示', 'リポジトリ', '操作', 'ツール', and 'ヘルプ'. Below the bar is a toolbar with icons for 'コミット', 'プル', 'プッシュ', 'フェッチ', 'ブランチ', 'マージ', 'スタッシュ', '破棄', and 'タグ'. The main area displays a commit history table with columns for '説明' (Description) and '日時' (Date/Time). The left sidebar shows a list of branches, including '#103_ML_tsuno', '#104_MAP_tsuno', '#105_baysian_regression', '#68_test_tsuno', '#96_kurouto_model_tsuno', and 'master'. The commit history table lists various commits, including merges and updates to README.md.

説明	日時
コミットされていない変更があります	2019/8/8 11:20
origin/#103_ML_tsuno #103_ML_tsuno 値を変更して、グラフを変更した	2019/7/31 21:24
origin/master origin/HEAD Merge pull request #113 from furukawa-laboratory/#_101_VAE_takano	2019/7/31 11:39
Merge pull request #108 from furukawa-laboratory/#100_Qiita_eigenvalue	2019/7/29 16:36
Merge pull request #120 from furukawa-laboratory/#109_harada_Qiita_1	2019/7/29 16:34
origin/#127_Qiita_kNN implement knn	2019/7/29 10:30
Merge pull request #112 from furukawa-laboratory/#107_Qiita_SVM	2019/7/27 17:23
Merge pull request #134 from furukawa-laboratory/#133_add_description_of_final_report	2019/7/26 16:50
#105_baysian_regression origin/#105_final_report_qiita2_baysian_regression テストデータの色を変更しました	2019/7/26 11:07
origin/#133_add_description_of_final_report Update README.md	2019/7/24 11:13
Fix typo	2019/7/24 11:11
origin/#103_ML_MAP_tsuno テストデータで検証できるようにした	2019/7/22 8:48
origin/#107_Qiita_SVM nothing	2019/7/19 23:16
Update README.md	2019/7/18 15:03
Update README.md	2019/7/18 15:00
Merge pull request #132 from furukawa-laboratory/Woukout_reportthoukoku	2019/7/18 14:40



- 最低限だけでも、意識せずに使えるようになった。
- 失敗してよい環境で、練習できてよかった。

Git & Github

<input type="checkbox"/>	 11 Open	<input checked="" type="checkbox"/> 53 Closed	Author ▾	Labels ▾	Projects ▾	Milestones ▾	Reviews ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	 Qiita2本目	MAP推定を用いた回帰	final report					×	10
	#128 opened 23 days ago by tsuno0829 • Approved								
<input type="checkbox"/>	 [Qiita記事3本目]	確率的主成分分析	final report	linear algebra	mini report	training			9
	#125 opened 23 days ago by takanoshuhei • Approved								
<input type="checkbox"/>	 [Qiita3本目]	3種類の基底関数によるベイズ線形回帰の実装	final report	training					7
	#124 opened 23 days ago by noguchikazuki • Approved								
<input type="checkbox"/>	 [Qiita3本目]	機械学習に挑むときの心構え(仮) → データの前処理についての記事に	final report	training					24
	#122 opened 23 days ago by forusufia • Changes requested 2 of 2								
<input type="checkbox"/>	 Qiita2本目	リッジ回帰の実装	final report	training					5
	#118 opened 24 days ago by noguchikazuki • Changes requested								
<input type="checkbox"/>	 Qiita1本目	特異値分解について	final report	training					7
	#117 opened 24 days ago by noguchikazuki • Approved								
<input type="checkbox"/>	 Qiita3本目	ベイズ推定を用いた回帰						×	13
	#116 opened 26 days ago by tsuno0829 • Approved								

- 作業に透明性ができた
- 周りの進捗も意識するようになった

前処理（ビギナー & 常人）

- 獲得スキルセット

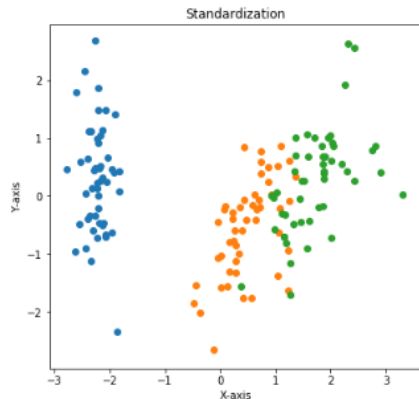
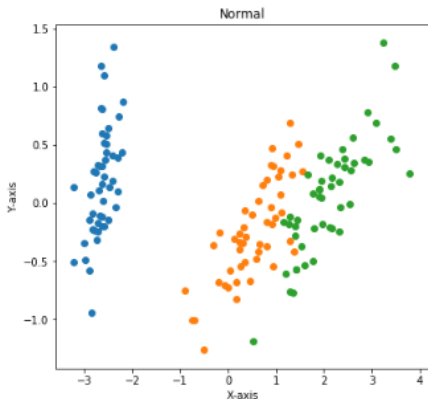
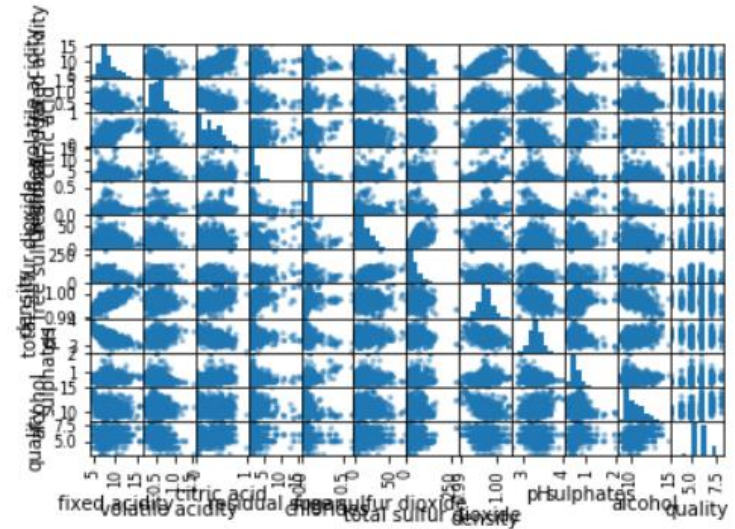
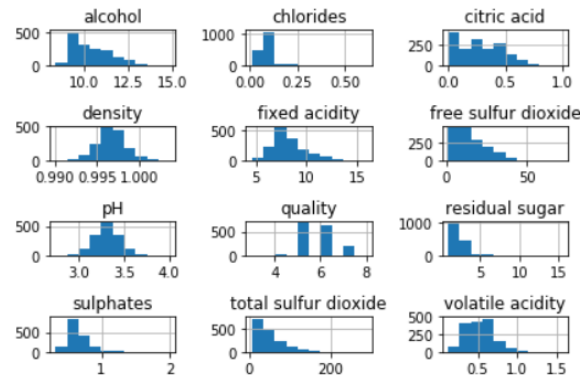
- 平均 0, 分散 1 に標準化できる ✓
- 任意のデータセットデータを使える ✓
- タスクに応じて適切なデータセットを調べられる ✓
- 標準的な形式のファイルを読み込める ✓

- データセット

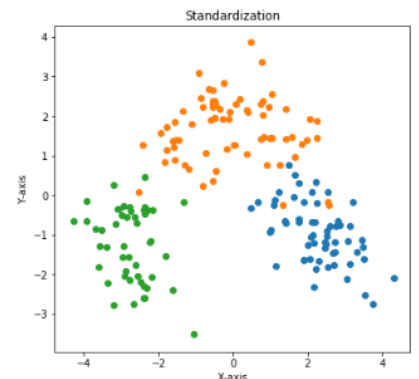
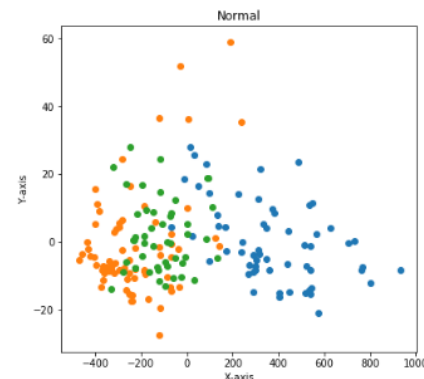
- sklearnのデータセット
 - Iris
 - Wine
 - Boston
- UCIレポジトリ
 - Adult
 - Wine

前処理（ビギナー＆常人）

- 正規化
 - 標準化
 - バイナリ化
 - One Hot Encoding
 - Label Encoding
- など



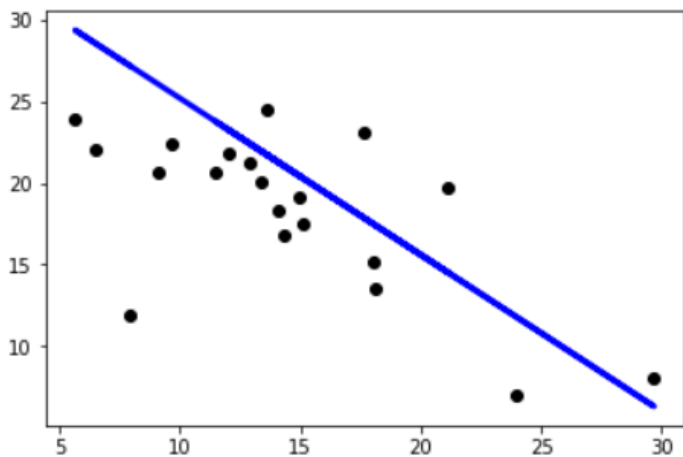
IrisデータをPCAかけたもの（左）普通（右）標準化



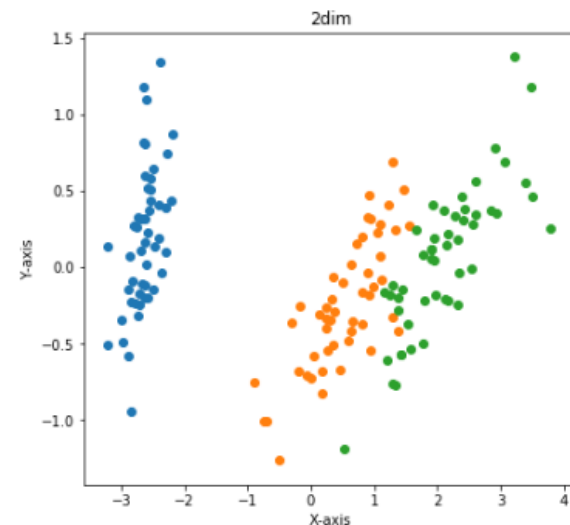
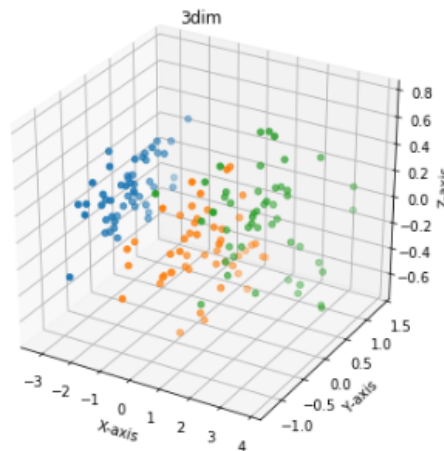
UCIのwineデータにPCAをかけたもの（左）普通（右）標準化

モデル（ビギナー）

- 獲得スキルセット
 - irisデータセットに対してPCAをかける ✓
 - Boston house-pricesに対して線形回帰を行い予測できる ✓
- sklearnを用いた実装済みモデルの利用
 - 単回帰
 - PCA



単回帰（Boston house-prices）



PCA（iris）

モデル（常人）

- 獲得スキルセット

- 任意のニューラルネットワークを構築できる✓

- Kerasでニューラルネットワーク（NN）の実装



- MLPでmnist分類
- CNN（畳み込みニューラルネットワーク）でmnist分類
- AutoEncoder（エンコーダ, デコーダ別々に作るのが難しかった）
 - スパース制約
 - Deep AutoEncoder
 - Convolutional AutoEncoder
 - ノイズ除去
 - Variational AutoEncoder

モデル（玄人）

獲得スキルセット

- 任意の数式をnumpy, tensorflowなどを用いて実装できる✓

内容

- Pythonのnumpyライブラリの特訓

感想

- 100 numpy exercises + a 英語で問題文の意味がよく分からない, 80番目以降は良く分からないのが多かった
- ブロードキャストのイメージやaxis方向の設定などが分かるようになった
- スクラッチ実装
 - 単回帰（簡単）
 - PCA（まだ終わってないです）

検証（ビギナー）

獲得スキルセット

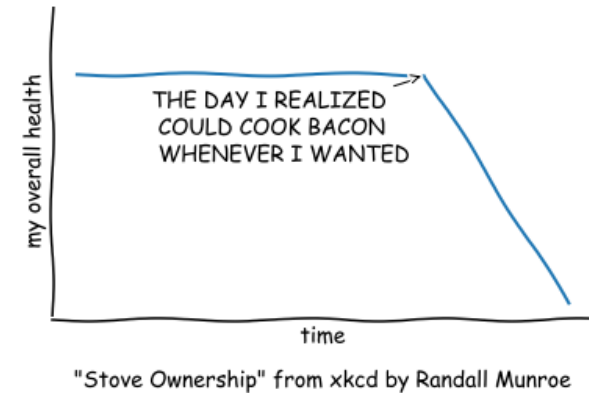
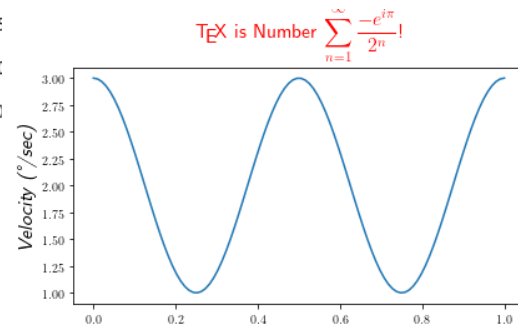
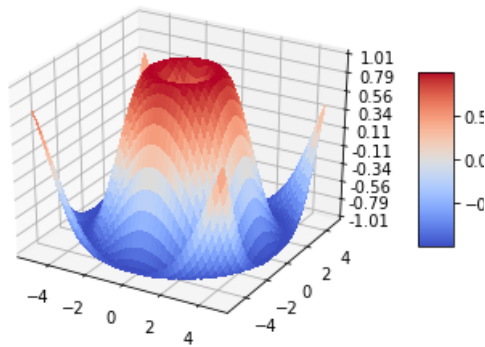
- 結果を出力できる ✓

matplotlibチュートリアルの色々な描画をひたすら写経した

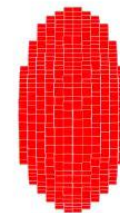
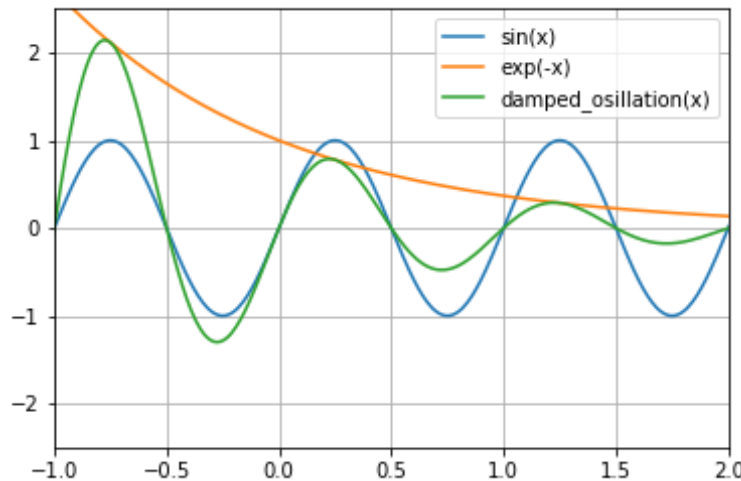
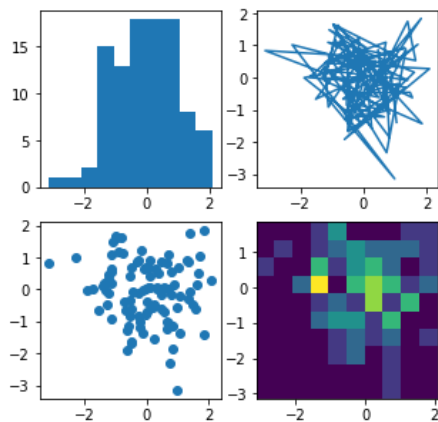
- 結構しんどかった...
 - Matplotlib公式のサンプルコードが様々な編集者によって投稿されてあるものだったので、書き方が毎度違っていた
 - Cbook（matplotlibがあらかじめ用意してあるサンプル画像）ライブラリのエラー
 - 今見返すと、過去自分が試行錯誤したときのコメントなどが残っててやって良かった
- matplotlibのアニメーションが動かない問題(.mp4)
 - Githubが静的なものにしか対応していなかったため
→ Gifなら大丈夫

検証（ビギナー）

matplotlibだけでも色々できることが分かった



好きな関数を3つ同じfigureに描画する課題 好きな関数をアニメーションで動かす



検証（常人）

- 獲得スキルセット

- 教師あり学習モデルの評価・選択が出来る ✓
- Cross Validation: CVができる ✓
- 結果の平均分散を計算できる（バイアスバリエンス）✓

1. バイアスとバリエンスについて

MSE(Mean Square Error) の式から Variance と Bias の式を導出する

母集団の真値: t , モデルの予測値: $y(x)$, 観測値（標本, 真値+ノイズ）: $h(x)$

平均損失または期待損失は

$$E[L] = \int \int L(t, y(x)) p(x, t) dx dt \quad (1)$$

で与えられる。回帰問題に良く使われる損失関数は二乗誤差 $L(t, y(x)) = \{y(x) - t\}^2$ である。この場合、期待損失は

$$E[L] = \int \int y(x) - t^2 p(x, t) dx dt \quad (2)$$

と書ける。目標は $E[L]$ を最小にする $y(x)$ を選ぶことなので、この最適解を変分法を使って求めると

$$\frac{\delta E[L]}{\delta y(x)} = 2 \int \{y(x) - t\} p(x, t) dt = 0 \quad (3)$$

また、確率の加法・乗法定理を使って式変形を行う・確率の加法定理 $P(X) = \sum_Y P(X, Y)$ ・確率の乗法定理 $P(X, Y) = P(Y|X)P(X)$ (3) 式より,

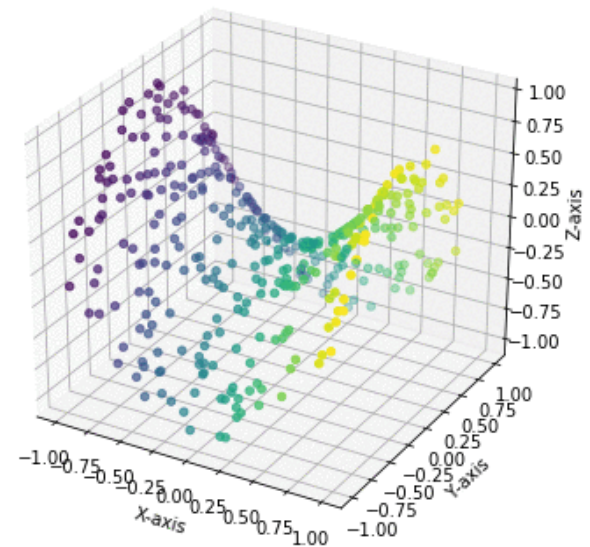
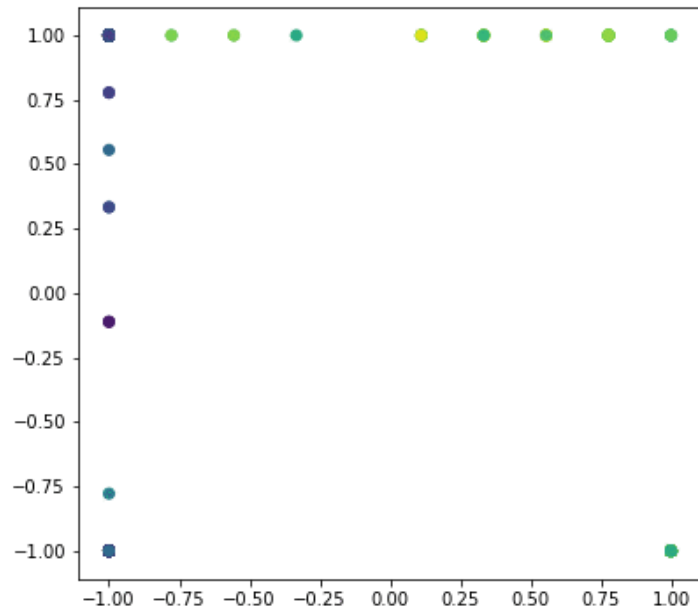
2. Cross Validation(交差検証)について

[交差検証のWikipedia](#)を参考

- 交差検証とは、統計学において標本データを分割し、その一部をまず解析して、残る部分でその解析のテストを行い、解析自身の妥当性の検証・確認に充てる手法を指す。データの解析がどれだけ本当に母集団に対処できるかを良い近似で検証・確認するための手法である。つまり、交差検証によってデータの分割によらない頑強な汎化精度の評価が可能になる。また、データセットを学習データとテストデータに分けた場合、テストデータを使わずに汎化性能が高いかどうか調べることができるのも特徴である。
- 交差検証の主な種類
 - ・ ホールドアウト検証・・・初期標本群から事例を無作為に選択しテスト事例を形成し、残る事例を訓練事例とする。ただし、データを交差させないため、交差検証には分類されない

SOM講習

1epoch



SOMのプログラム

- 学んだことが結構使えてる？

```
# 近傍半径を単調現象させるスケジュールの作成
sigma_t = max(sigma_max - (sigma_max - sigma_min)*(t/tau), sigma_min)
print('sigma_t:{}'.format(sigma_t))

if t == 0:
    # 潜在変数Zの初期化
    k = np.random.randint(K*K, size=(N*N))

# 参照ベクトルYの推定
# h : K×N
h = np.exp( -1/(2*(sigma_t**2)) * np.sum((zeta[k][None,:,:] - zeta[:,None,:])**2, axis=2) )

# g : K×1
g = np.sum(h, axis=1)

# y : K×D
y = (1/g[:,None]) * h @ X

# 潜在変数Zの推定
k = np.argmin( np.sum((X[:,None,:] - y[None,:,:])**2, axis=2), axis=1 )
|
# 描画
ax1.scatter(zeta[k][:,0], zeta[k][:,1], c=X[:,0])
ax2.scatter3D(X[:,0], X[:,1], X[:,2], c=X[:,0])
Y = np.reshape(y, (K,K,3))
ax2.plot_wireframe(Y[:, :, 0], Y[:, :, 1], Y[:, :, 2])
ax2.set_xlabel("X-axis")
ax2.set_ylabel("Y-axis")
ax2.set_zlabel("Z-axis")
fig.suptitle('{}epoch'.format(t+1), fontsize=16)
```

Qiita記事（final report）



全部ベイズ（確率）関係の記事

最近の記事 4

いいねした記事

限定共有記事

コメント



tsnry7913 が2019/08/06に投稿
ベイズ推定を用いた回帰

👍 4



Python

機械学習

ベイズ推定

回帰



tsnry7913 が2019/08/01に投稿
MAP推定を用いた回帰

👍 9



Python

機械学習

回帰

MAP推定



tsnry7913 が2019/07/27に投稿
最尤推定を用いた回帰

👍 10



Python

機械学習

最小二乗法

最尤推定

回帰

「取り組んだ理由」

ワークアウトで自分が行ってきた内容を振り返ると応用的なものばかりだったので、もっと理論的な知識を身に着けたいと思ったため

ベイズの獲得スキルセット

- ビギナー：ベイズルール
 - 条件付き確率，周辺化，同時確率，独立の定義が書ける ✓
 - 条件付き確率の定義からベイズの定理を導出できる ✓
 - ベイズの定理の式の各部分の名前が言える ✓
- 常人：点推定
 - 問題設定を説明できる ✓
 - 最尤推定でデータからモデルのパラメータを推定できる ✓
 - MAP推定で（以下同 ✓
- 玄人：分布推定
 - 点推定との違いを説明できる ✓
 - ベイズ推定でデータからモデルのパラメータの確率分布を推定できる ✓

ベイズの定理

事象Aが起きる確率を $p(A)$ ，事象Bが起きる確率を $p(B)$ ，
事象Bが起こった後，事象Aが起こる確率を $p(A|B)$ は

$$p(A|B) = \frac{p(A, B)}{p(B)}$$

と定義されます．事象AとBが同時に起きる確率は $p(A, B)$
で表されます．また，上式を式変形すると，

$$p(A, B) = p(A|B)p(B)$$

$$p(B, A) = p(B|A)p(A)$$

$p(A, B)$ と $p(B, A)$ は同じものなので，次の式が得られます．

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

この式のことを，ベイズの定理と呼びます．

ベイズの定理

ここで, 例えば $B=X$ (観測データ, 既知), $A=C_i$ (クラス, 未知) とすると,

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

ベイズの定理の各項は以下のように呼ばれます

$p(C_i|X)$: 事後分布

$P(X|C_i)$: 尤度, 尤度関数

$P(C_i)$: 事前分布

$P(X)$: エビデンス, 周辺尤度

また, 周辺尤度は

$$P(X) = \sum_{i=1} P(X, C_i) = \sum_{i=1} P(X|C_i)P(C_i)$$

より,

$$P(C_i|X) = \frac{P(X, C_i)}{\sum_{i=1} P(X, C_i)} = \frac{P(X|C_i)P(C_i)}{\sum_{i=1} P(X|C_i)P(C_i)}$$

最尤推定を用いた回帰

与えられているもの

入力 $\mathbf{x}=(x_1, \dots, x_N)$, $x \in \mathbb{R}$ と出力 $\mathbf{y}=(y_1, \dots, y_N)$, $y \in \mathbb{R}$ のデータセット

仮定

- ・ データセットは, 独立同分布 (i.i.d.) に従うとします.
- ・ y_i, f, ϵ を次のように仮定します.

$$y_i = f(x_i; \mathbf{w}) + \epsilon_i$$

$$f(x_i; \mathbf{w}) = \mathbf{w}^T \boldsymbol{\phi}(x_i)$$

$$\epsilon_i \sim N(0, \beta^{-1})$$

- ・ 基底関数 $\boldsymbol{\phi}(x)$ とパラメータ \mathbf{w} はD個あるとします.

$$\boldsymbol{\phi}(x_i) = (\phi_1(x_i), \dots, \phi_D(x_i))$$

$$\mathbf{w} = (w_1, \dots, w_D)$$

最尤推定を用いた回帰

ベイズの定理

$$p(\mathbf{w}|\mathbf{x}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{x})}$$

最尤推定のタスク

$$\mathbf{w}_{ML} = \operatorname{argmax}_{\mathbf{w}} p(\mathbf{y}|\mathbf{x}, \mathbf{w})$$

推定した \mathbf{w}_{ML} から, 新規入力 x^* の出力 $y^* = \mathbf{w}_{ML}^T \boldsymbol{\phi}(x^*)$ を求めることができます.

最尤推定を用いた回帰

観測データが (x_i, y_i) のとき, $y_i = f(x_i; \mathbf{w}) + \epsilon_i$
と仮定すると尤度は $p(y_i|x_i, \mathbf{w})$ を次のようになります

$$p(y_i|x_i, \mathbf{w}) = N(y_i|f(x_i; \mathbf{w})) = \sqrt{\frac{\beta}{2\pi}} \exp(-\frac{\beta}{2} (y_i - \mathbf{w}^T \boldsymbol{\phi}(x_i))^2)$$

データセット全体の尤度は, 独立同分布(i.i.d.)に従うとしているので,

$$\prod_{i=1}^N p(y_i|x_i, \mathbf{w}) = p(y_1|x_1, \mathbf{w}) \times \cdots \times p(y_N|x_N, \mathbf{w})$$

となります. 更に計算を簡単にするために対数をとると,

$$\begin{aligned} \ln \left(\prod_{i=1}^N p(y_i|x_i, \mathbf{w}) \right) &= \sum_{i=1}^N \ln p(y_i|x_i, \mathbf{w}) \\ &= -\frac{\beta}{2} \|\mathbf{y} - \Phi \mathbf{w}\|^2 + \text{const} \end{aligned}$$

最尤推定を用いた回帰

ここで、 Φ は次のようになります。

$$\Phi = \begin{pmatrix} \phi_1(x_1) & \dots & \phi_D(x_1) \\ \vdots & \ddots & \vdots \\ \phi_1(x_N) & \dots & \phi_D(x_N) \end{pmatrix}$$

求めた対数尤度はパラメータ w の関数とみなせるので、対数尤度関数 $L(w)$ とおくと次のようになります。

$$L(w) = -\frac{\beta}{2} \|\mathbf{y} - \Phi \mathbf{w}\|^2 + \text{const}$$

ここで、回帰における目的関数（最小二乗法）は、

$$E(w) = \frac{\beta}{2} \|\mathbf{y} - \Phi \mathbf{w}\|^2$$

です。つまり、尤度関数を最大化することと二乗誤差を最小化することは等価であることが分かります。また、最尤推定が回帰における最小二乗誤差を統計的に妥当性を裏付けていることが分かります。ただし、ガウスノイズを仮定したときのみ成り立ちます。

最尤推定を用いた回帰

対数尤度関数 $L(\mathbf{w})$ を \mathbf{w} で微分して0となる極値を求めると

$$\begin{aligned}\frac{\partial L(\mathbf{w})}{\partial \mathbf{w}} &= \frac{\partial}{\partial \mathbf{w}} \left(-\frac{\beta}{2} \|\mathbf{y} - \Phi \mathbf{w}\|^2 + \text{const} \right) \\ &= -\frac{\beta}{2} (2\Phi^T \mathbf{y} + 2\Phi^T \Phi \mathbf{w}) = 0\end{aligned}$$

より, 式変形すると

$$\Phi^T \Phi \mathbf{w} = \Phi^T \mathbf{y}$$

となつて, 最尤推定で得られた \mathbf{w}_{ML} は,

$$\begin{aligned}\mathbf{w}_{ML} &= (\Phi^T \Phi)^{-1} \Phi^T \mathbf{y} \\ &= \Phi^\dagger \mathbf{y}\end{aligned}$$

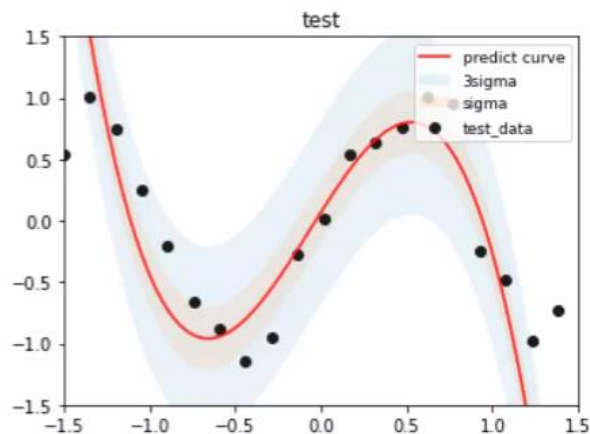
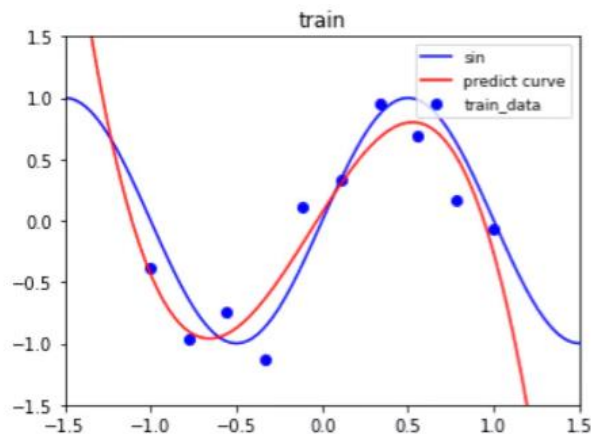
となります. この式は, 最小二乗法の正規方程式として知られています. また, $N \times D$ 行列 Φ^\dagger はムーア=ペンローズの一般化逆行列もしくは疑似逆行列と呼ばれ, 長方形行列に対しての逆行列みたいなものとみなすことができます.

最尤推定を用いた回帰

推定した \mathbf{w}_{ML} から新規入力 x^* の出力 y^* は次のようになります.

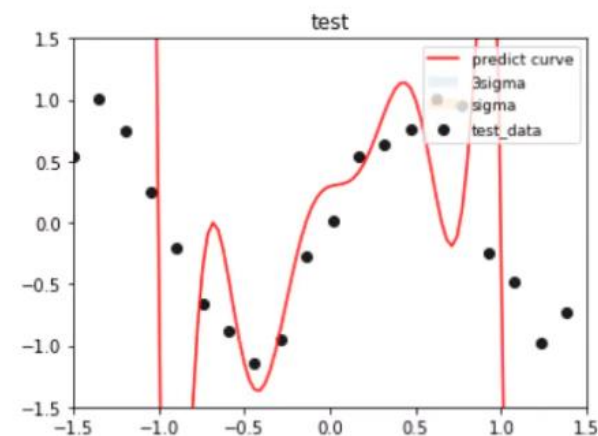
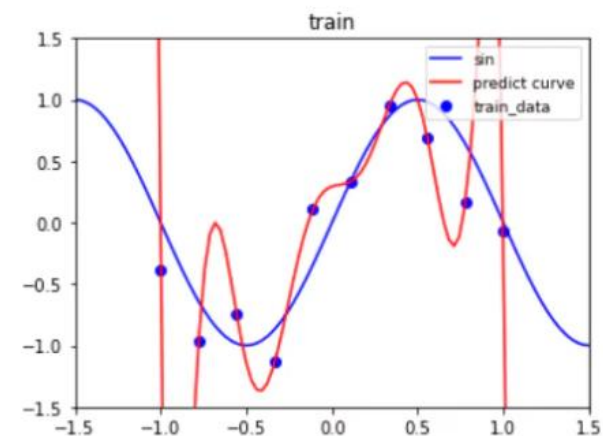
$$y^* = \mathbf{w}_{ML}^T \boldsymbol{\phi}(x^*)$$

実験結果



基底関数が3次多項式の時

w0: 0.07 w1: 2.20 w2: -0.42 w3: -2.12



基底関数が9次多項式の時

w0: 0.30 w1: 0.48 w2: -6.38 w3: 47.05 w4: 32.27 w5: -253.78 w6: -58.38 w7: 423.28 w8: 31.97 w9: -216.86

MAP推定を用いた回帰

与えられているもの

入力 $\mathbf{x}=(x_1, \dots, x_N)$, $x \in \mathbb{R}$ と出力 $\mathbf{y}=(y_1, \dots, y_N)$, $y \in \mathbb{R}$ のデータセット

仮定

- ・ データセットは, 独立同分布 (i.i.d.) に従うとします.
- ・ y_i, f, ε_i を次のように仮定します.

$$\begin{aligned}y_i &= f(x_i; \mathbf{w}) + \varepsilon_i \\f(x_i; \mathbf{w}) &= \mathbf{w}^T \boldsymbol{\phi}(x_i) \\ \varepsilon_i &\sim N(0, \beta^{-1})\end{aligned}$$

- ・ 基底関数 $\boldsymbol{\phi}(x)$ とパラメータ \mathbf{w} はD個あるとします.

$$\begin{aligned}\boldsymbol{\phi}(x_i) &= (\phi_1(x_i), \dots, \phi_D(x_i)) \\ \mathbf{w} &= (w_1, \dots, w_D)\end{aligned}$$

- ・ 事前分布はあまり大きな \mathbf{w} を取らないと仮定します (α はハイパーパラメータ)

$$p(\mathbf{w}) = N(\mathbf{w} | \mathbf{0}, \alpha^{-1} \mathbf{I})$$

MAP推定を用いた回帰

ベイズの定理

$$p(\mathbf{w}|\mathbf{x}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{x})}$$

最尤推定のタスク

$$\mathbf{w}_{MAP} = \operatorname{argmax}_{\mathbf{w}} p(\mathbf{y}|\mathbf{x}, \mathbf{w})p(\mathbf{w})$$

推定した \mathbf{w}_{MAP} から, 新規入力 x^* の出力 $y^* = \mathbf{w}_{MAP}^T \boldsymbol{\phi}(x^*)$ を求めることができます.

MAP推定を用いた回帰

仮定より事前分布は

$$p(\mathbf{w}) = N(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$$

より, 対数事前分布は

$$\ln p(\mathbf{w}) = -\frac{\alpha}{2} \|\mathbf{w}\|^2 + \text{const}$$

となる. よって, ベイズの定理から事後分布に対数をとると

$$\begin{aligned} \ln p(\mathbf{w}|\mathbf{x}, \mathbf{y}) &= \ln p(\mathbf{y}|\mathbf{x}, \mathbf{w}) + \ln p(\mathbf{w}) + \text{const} \\ &= -\frac{\beta}{2} \|\mathbf{y} - \Phi\mathbf{w}\|^2 - \frac{\alpha}{2} \|\mathbf{w}\|^2 + \text{const} \end{aligned}$$

この式を目的関数 $M(\mathbf{w})$ とみなすと次のようになります.

$$M(\mathbf{w}) = -\frac{\beta}{2} \|\mathbf{y} - \Phi\mathbf{w}\|^2 - \frac{\alpha}{2} \|\mathbf{w}\|^2 + \text{const}$$

ところで, Ridge回帰の目的関数 $E(\mathbf{w})$ は

$$E(\mathbf{w}) = \frac{1}{2} \|\mathbf{y} - \Phi\mathbf{w}\|^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

となっており, $M(\mathbf{w})$ を β で割って $\lambda = \frac{\alpha}{\beta}$ とすると, $M(\mathbf{w})$ の最大化と $E(\mathbf{w})$ の最小化は同じです. よって, MAP推定とRidge回帰の結果は等価になります. ただし, 事前分布がガウス分布でノイズもガウスのとき.

MAP推定を用いた回帰

事後分布が最大となるように目的関数 $M(\mathbf{w})$ を偏微分して0となる極値を求めると

$$\begin{aligned}\frac{\partial M(\mathbf{w})}{\partial \mathbf{w}} &= \frac{\partial}{\partial \mathbf{w}} \left(-\frac{\beta}{2} \|\mathbf{y} - \Phi \mathbf{w}\|^2 - \frac{\alpha}{2} \|\mathbf{w}\|^2 + \text{const} \right) \\ &= \beta \Phi^T \mathbf{y} - \beta \Phi^T \Phi \mathbf{w} - \alpha \mathbf{w} = 0\end{aligned}$$

より, 式変形すると

$$(\Phi^T \Phi + \lambda \mathbf{I}) \mathbf{w} = \Phi^T \mathbf{y}$$

となつて, MAP推定で得られた \mathbf{w}_{MAP} は,

$$\mathbf{w}_{MAP} = (\Phi^T \Phi + \lambda \mathbf{I})^{-1} \Phi^T \mathbf{y}$$

推定した \mathbf{w}_{MAP} から新規入力 x^* の出力 y^* は次のようになります.

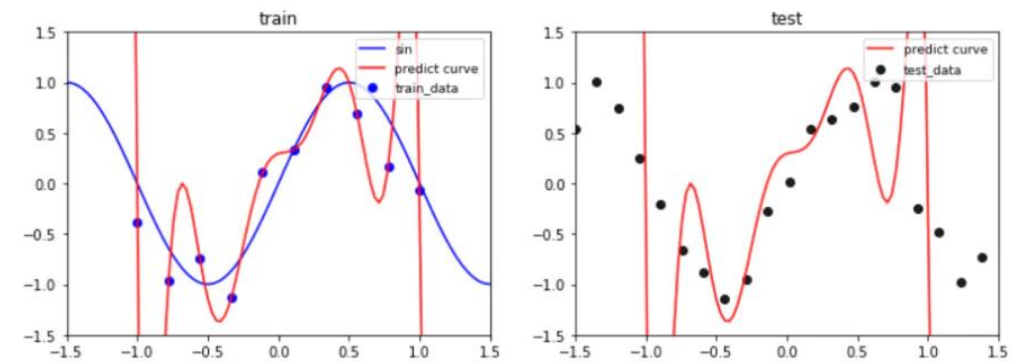
$$y^* = \mathbf{w}_{MAP}^T \boldsymbol{\phi}(x^*)$$

実験結果

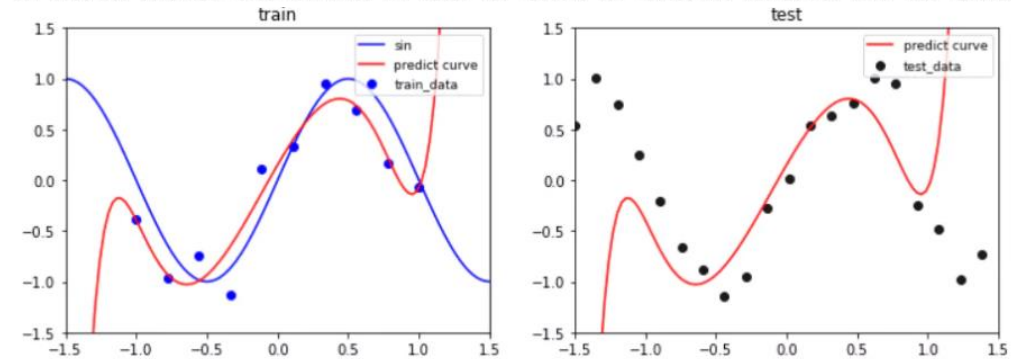
基底関数が9次多項式で $\alpha=0$ のとき

基底関数が9次多項式で $\alpha=0.03$ のとき

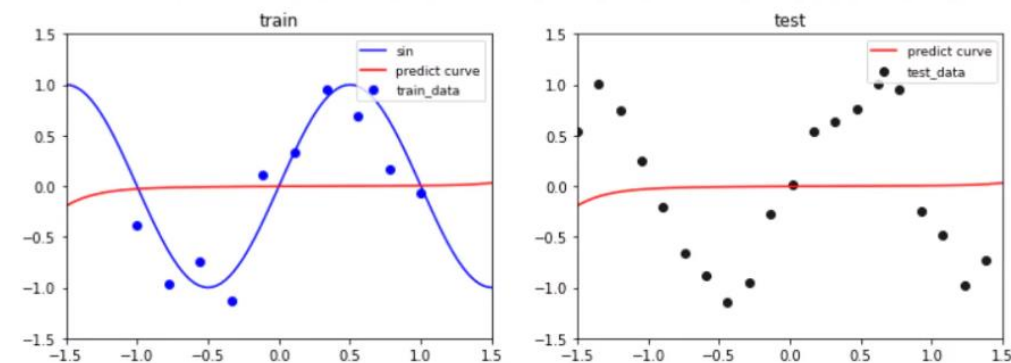
基底関数が9次多項式で $\alpha=100$ のとき



w0: 0.30 w1: 0.48 w2: -6.38 w3: 47.05 w4: 32.27 w5: -253.78 w6: -58.38 w7: 423.28 w8: 31.97 w9: -216.86



w0: 0.150 w1: 2.394 w2: -0.853 w3: -2.522 w4: -0.247 w5: -0.806 w6: 0.125 w7: 0.304 w8: 0.592 w9: 0.788



w0: -0.003 w1: 0.008 w2: -0.003 w3: 0.003 w4: -0.002 w5: 0.002 w6: -0.002 w7: 0.002 w8: -0.002 w9: 0.001

点推定と分布推定の違い

最尤推定とMAP推定がモデルのパラメータ w に関して、最も「良い」答えをただ一つ求める点推定（それ以外の他の可能性は全て切り捨てる）なのに対して、ベイズ推定はパラメータ w に関して「良い」だけでなく、「まあまあ良い」答えなどあらゆる答え（可能性）を含んだ分布を推定できることが大きな違いです。

ベイズ推定を用いた回帰

与えられているもの

入力 $\mathbf{x}=(x_1, \dots, x_N)$, $x \in \mathbb{R}$ と出力 $\mathbf{y}=(y_1, \dots, y_N)$, $y \in \mathbb{R}$ のデータセット

仮定

- ・ データセットは, 独立同分布 (i.i.d.) に従うとします.
- ・ y_i, f, ε_i を次のように仮定します.

$$\begin{aligned}y_i &= f(x_i; \mathbf{w}) + \varepsilon_i \\f(x_i; \mathbf{w}) &= \mathbf{w}^T \boldsymbol{\phi}(x_i) \\ \varepsilon_i &\sim N(0, \beta^{-1})\end{aligned}$$

- ・ 基底関数 $\boldsymbol{\phi}(x)$ とパラメータ \mathbf{w} はD個あるとします.

$$\begin{aligned}\boldsymbol{\phi}(x_i) &= (\phi_1(x_i), \dots, \phi_D(x_i)) \\ \mathbf{w} &= (w_1, \dots, w_D)\end{aligned}$$

- ・ 事前分布はあまり大きな \mathbf{w} を取らないと仮定します (α はハイパーパラメータ)

$$p(\mathbf{w}) = N(\mathbf{w} | \mathbf{0}, \alpha^{-1} \mathbf{I})$$

ベイズ推定を用いた回帰

ベイズ推定のタスク

$$p(\mathbf{w}|\mathbf{x}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{x})}$$

推定した \mathbf{w} の事後分布から, 新規入力 x^* の出力 y^* の予測分布を求めることができます.

ベイズ推定を用いた回帰

尤度と事前分布はMAP推定と同じとすると, ベイズの定理から対数事後分布は

$$\begin{aligned}\ln p(\mathbf{w}|\mathbf{x}, \mathbf{y}) &= \ln p(\mathbf{y}|\mathbf{x}, \mathbf{w}) + \ln p(\mathbf{w}) + \text{const} \\ &= -\frac{\beta}{2} \|\mathbf{y} - \Phi \mathbf{w}\|^2 - \frac{\alpha}{2} \|\mathbf{w}\|^2 + \text{const} \\ &= -\frac{1}{2} \mathbf{w}^T (\beta \Phi^T \Phi + \alpha \mathbf{I}) \mathbf{w} + (\beta \mathbf{y}^T \Phi) \mathbf{w} + \text{const}\end{aligned}$$

となります. ここで, 尤度関数と事前分布は共役な関係にあるので事後分布は多変量ガウス分布になります. よって, 対数事後分布は

$$\begin{aligned}\ln p(\mathbf{w}|\mathbf{x}, \mathbf{y}) &= -\frac{1}{2} (\mathbf{w} - \mathbf{m})^T S^{-1} (\mathbf{w} - \mathbf{m}) + \text{const} \\ &= -\frac{1}{2} \mathbf{w}^T S^{-1} \mathbf{w} + \mathbf{m}^T S^{-1} \mathbf{w} + \text{const}\end{aligned}$$

\mathbf{w} の2次の項と1次の項は対応した関係になっているので式変形をすると次のようになります.

$$\begin{aligned}S &= (\beta \Phi^T \Phi + \alpha \mathbf{I})^{-1} \\ \mathbf{m} &= (\Phi^T \Phi + \lambda \mathbf{I})^{-1} \Phi^T \mathbf{y}\end{aligned}$$

ベイズ推定を用いた回帰

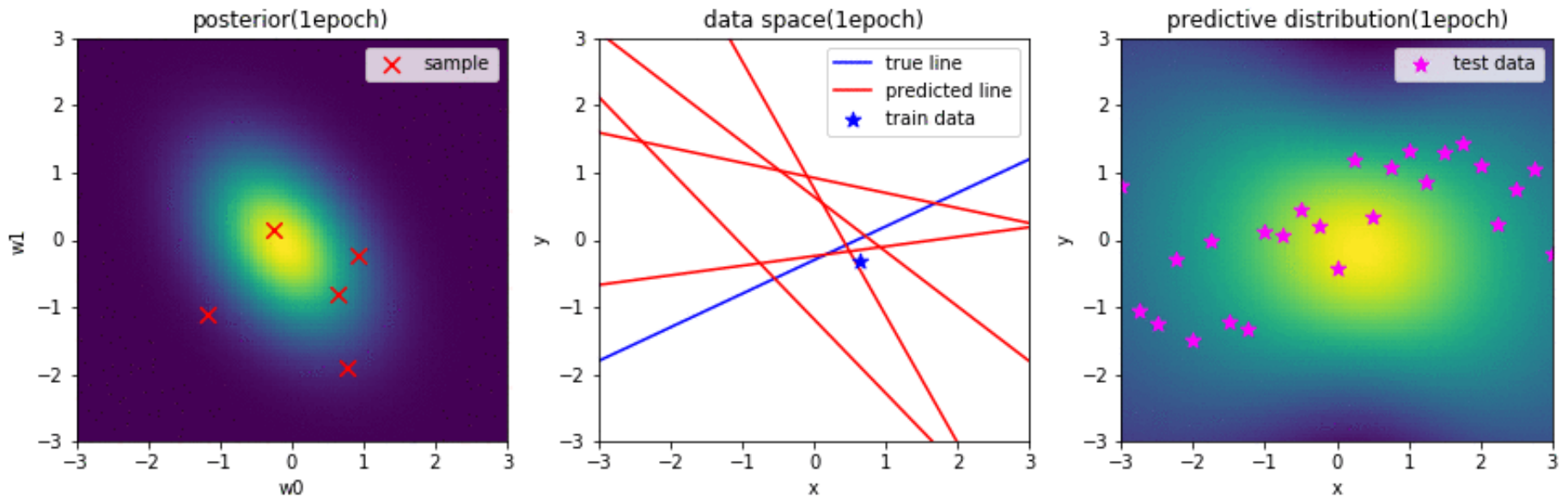
推定した \mathbf{w} の事後分布から新規入力 x^* の出力 y^* は次のようになります.

$$\begin{aligned} p(y^*|x^*, \mathbf{x}, \mathbf{y}) &= \int p(y^*|x^*, \mathbf{w})p(\mathbf{w}|\mathbf{x}, \mathbf{y})d\mathbf{w} \\ &= \int N(y^*|\mathbf{w}^T \boldsymbol{\phi}(x^*), \beta^{-1})N(\mathbf{w}|\mathbf{m}, \mathbf{S})d\mathbf{w} \end{aligned}$$

この予測分布の式は, ガウス分布の畳み込み積分になっており, この積分はPRMLの2-116の公式を使うことで以下の式になります.

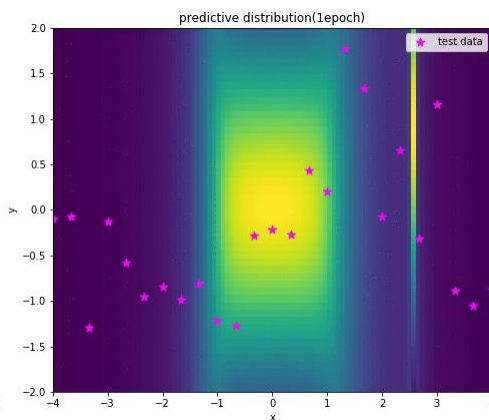
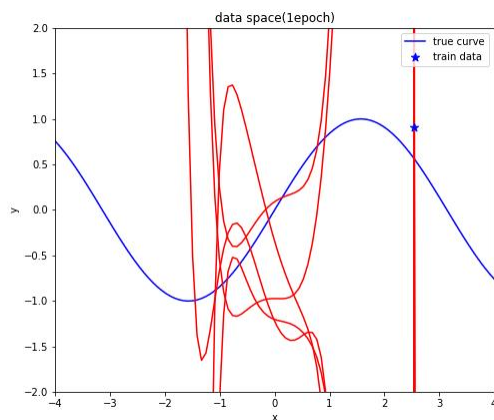
$$p(y^*|x^*, \mathbf{x}, \mathbf{y}) = N(y^*| \mathbf{m}^T \boldsymbol{\phi}(x^*), \beta^{-1} + \boldsymbol{\phi}(x^*)^T \mathbf{S} \boldsymbol{\phi}(x^*))$$

実験結果



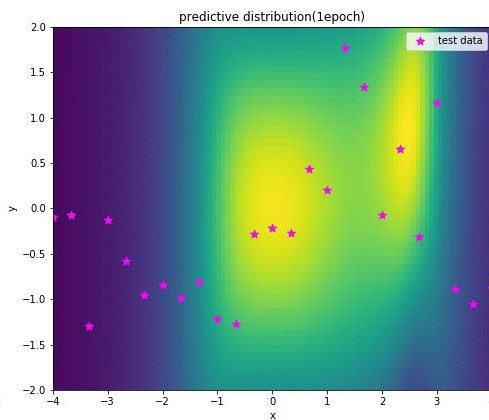
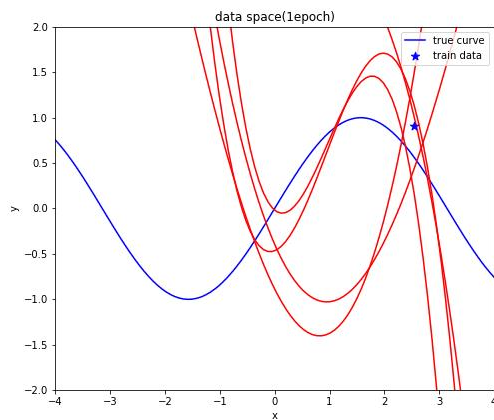
ベイズ推定を用いた線形回帰 ($w_0=0.3, w_1=0.5, \alpha=1, \beta=0.5$)

実験結果

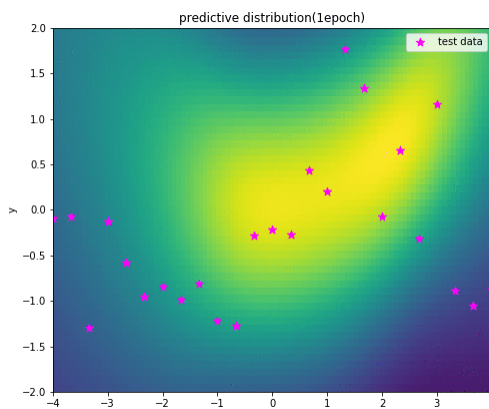
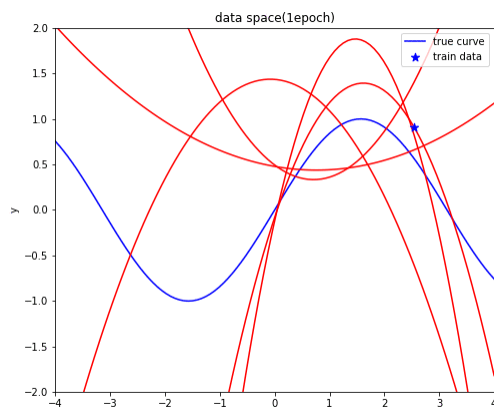


$$\alpha=1, \beta=0.5$$

9次多項式



3次多項式



2次多項式

まとめ

- 線形代数の復習
- スクラッチ実装したものが本当にあっているかをsklearn等を用いて検証する