課題 1
SVM の作成と評価

宮口 直樹
1029-29-1418

October 18, 2019

1　プログラム概要

課題 1 では，SVM を作成するということで以下の 2 つを目的として取り組んだ．また，自分でデータを作成して試してはいないが，n 次元の特徴に対応している．

1. SVM のプログラムを作成して識別器を出力する．
2. 1 で作成した SVM を交差検定によって評価する．

2　外部仕様

2.1　実行環境

- MacOS
- Python 3.6.9
- numpy 1.16.5
- matplotlib 3.1.1
- cvxopt 1.2.0

2.2　ファイルの説明

svm.py SVM を実行するためののアルゴリズムが記述されたファイル．

cross_validation.py 交差検定を行うためのファイル．

init.py データ読み込みやコマンドライン引数を定義する関数などを含むファイル．

kernel.py 3 種類のカーネルが定義されたファイル．

2.3　実行方法

コマンドライン引数を読み込むためには，argparse を使用した．SVM の識別器を出力するためには svm.py を実行する．この際に訓練データのファイルパスと使用するカーネル名の 2 つの引数を渡す必要がある．引数の指定の仕方は-h コマンドで確認することができるが，訓練データのファイルパスは-f コマンドの後に記述，カーネル名は-k コマンドの後に記述することで指定できる．またカーネル名に関しては {None, gaussian_kernel, polynomial_kernel, sigmoid_kernel} の中から指定する必要があり，指定がなければカーネルなし

でプログラムを実行する．交差検定を行うには，cross_validation.py を実行する．この際に上の 2 つの引数に加えて入力データの分割数も-n コマンドの後に指定する必要がある．コマンドの使用方法を表にまとめると下の Table 1 のようになる．

Table 1: コマンド使用方法

コマンド (略)	記述 1	備考
–filename(-f)	訓練データのパス	なし
–kernel_type(-k)	カーネルの種類	None, polynomial_kernel, gaussian_kernel, sigmoid_kernel から選択
–division(-n)	入力データ分割数	2 以上の整数を指定

2.4 実行例

2.4.1 svm.py

sample_circle.txt をガウスカーネルを用いて解きたいときの SVM 実装プログラムの実行コマンドと出力結果は下の通りである．実際にはグラフも出力されるのだが，後の評価結果の章で説明を行う．

実行コマンド

```
python3 svm.py -f 'sample_circle.txt' -k 'gaussian_kernel'
```

出力結果

```
----結果出力----
解：α=array([[5.32028316e-14],
                    :
             [9.50296185e-14]])
重み：w=array([-5.32028316e-14, -6.72074405e-14,
                                :
                2.34076683e+00, -9.50296185e-14])
閾値：θ=3.562198e+00
```

2.4.2 cross_validation.py

sample_circle.txt をガウスカーネルを用いて解きたいときに交差検定を実行した時の例である．ここで，データ分割数は 5 とする．

実行コマンド

```
python3 cross_validation.py -f ”sample_circle.txt” -k ’gaussian_kernel’ -n 5
```

出力結果

```
accuracy = 1.000000e+00
accuracy = 8.000000e-01
accuracy = 8.500000e-01
accuracy = 9.500000e-01
accuracy = 8.500000e-01
total accuracy = 8.900000e-01
```

2.5 エラー処理

エラー処理に関しては，python 備え付けの argparse を利用することで不正な入力が与えられた時に必要な処理を教えてくれる．

3 内部仕様

3.1 各関数の説明

関数の説明に関しては，ファイルごとに章を分けて説明を行う．各ファイルの説明については，2.2 章で既に述べているため省略する．

3.1.1 init.py

・set_parser()
コマンドラインの引数を argparse を用いて設定する関数．コマンド入力で得た最大 3 つの引数を返す．

・min_max(x, axis=None)
1 次元配列の入力データを受け取り 0 1 の間にスケーリングした配列を返す関数．今課題では使わなかったが，一応残している．

・load_data(filename)
コマンドライン引数から受け取った filename を引数として，テキストデータ

から特徴データと正解ラベルのデータに分けて配列として返す関数.

3.1.2 kernel.py

・polynomial_kernel(x, y)
多項式カーネルを実装するための関数.

・gaussian_kernel(x, y)
ガウスカーネルを実装するための関数.

・sigmoid_kernel(x, y)
シグモイドカーネルを実装するための関数.

3.1.3 svm.py

・fit(x, y, kernel)
読み込んだデータから 2 次計画問題を解き，解を出力する関数.

・func_no_kernel(x1, w, b)
カーネルなしの時に識別器を計算して求める関数.

・func_kernel(x, mesh_lst, alphas, y, b, kernel)
カーネルが指定されている時に識別器を計算して求める関数.

・draw_graph(x, y, f)
カーネルなしの時にグラフを描写するための関数.

・draw_graph_kernel(x, y, mesh_x, mesh_y, f)
カーネルありの時にグラフを描写するための関数.

・main()
データ読み込みから結果出力まで SVM 全体を動かすための関数.

3.1.4 cross_validation.py

・split_data(x, y, n, i)
入力データをコマンドライン引数から得られた分割数を元にして，訓練データと教師データに分ける関数.

・cross_validate_no_kernel(test_x, test_y, w, b, total_accuracy)
カーネルなしの時に交差検定を実装して結果を出力する関数.

・**cross_validate_kernel(train_x, test_x, train_y, test_y, alphas, b, kernel, total_accuracy)**
カーネルありの時に交差検定を実装して結果を出力する関数.

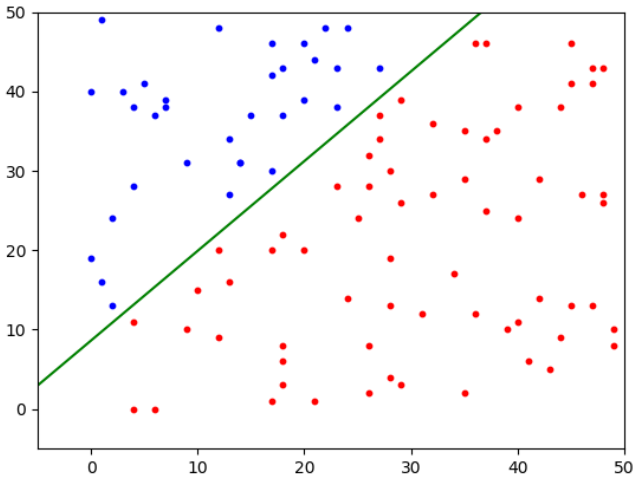
・**main()**
データ読み込みと分割，結果出力までの動作を全て実行する関数.

4 評価結果

4.1 SVM 実装結果

この章では評価結果を出力の図と共に示す．なおグラフの出力には matplotlib の pyplot を使用した．まず初めに sample_linear.txt に対して，カーネルトリックなしで SVM を実装した結果の図が Figure 1 である．この章の図では青い点が-1，赤い点が+1 の正解ラベルを持つとする．識別器は緑の直線で表されており，緑の直線を境界として正しく分類ができていることが分かる．

Figure 1: カーネルトリックなし



次にカーネルトリックを用いて，sample_circle.txt に SVM を実装した．ガウスカーネルを用いた結果については Figure 2，多項式カーネルを用いた結果については Figure 3 である．ここでガウスカーネルでは $\sigma = 10$ とした．先ほどと色が変わってしまったが，黒の曲線が識別器の境界を示しており，正しく分類できていることが図から読み取れる．

Figure 2: ガウスカーネル

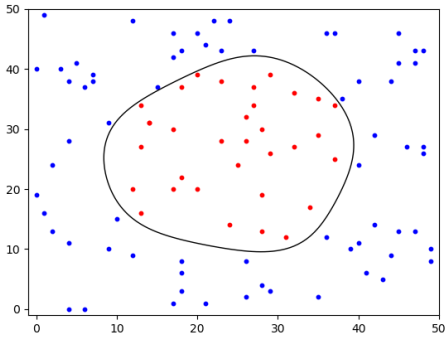
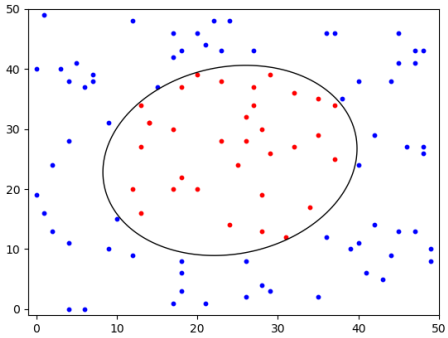
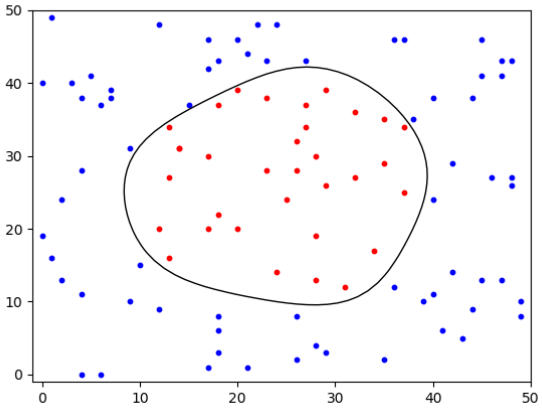


Figure 3: 多項式カーネル



最後にガウスカーネルのパラメータ $\sigma = 5$ として実行した結果が Figure 4 である．

Figure 4: ガウスカーネル（σ 変更）



4.2 交差検定結果

まず sample_linear.txt に関して交差検定を行ったところ，データ分割数をいかなる値，例えば最小の 2 にしても正解率 100%であった．sample_circle.txt

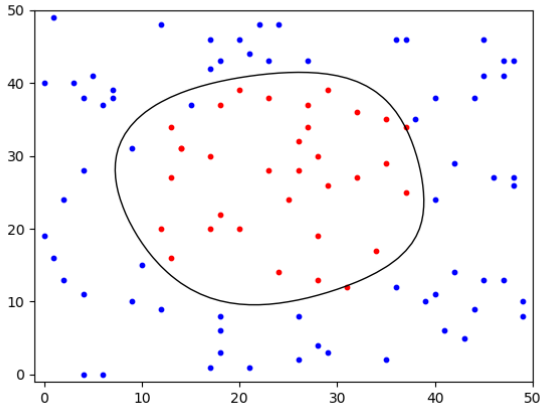
に対して検定を行った結果を下の Table 2 にまとめる．

Table 2: コマンド使用方法

カーネル	分割数	σ の値	結果 (平均)
ガウス	5	10	89%
ガウス	10	10	90%
ガウス	5	5	91%
ガウス	10	5	91%
多項式	5	-	96%
多項式	5	-	96%

参考までに，ガウスカーネルを用いて分割数 5 とした場合で正答率が最も悪い 80%であった時のグラフは，以下の Figure 5 のようになった．

Figure 5: 参考画像



5 考察

評価結果から自明なように，sample_linear.txt に対しては，非常に精度の高い分類が可能であった．分割数を変更しても正解率は 100%であったのは，サンプルデータが非常に綺麗に分類できるものであったということも理由と考

えられる．sample_circle.txt に対しても，交差検定を実装しても正解率が高く図を見ても正しく分類できたと言える．またガウスカーネルにおいて，σ の値を小さくすると境界線の凹凸が激しくなり，逆に大きくすれば滑らかな曲線が描写される．多項式カーネルにおいては，カーネルの定義式の性質上楕円のような境界線が描かれていることがわかった．こちらのサンプルデータもそもそもが綺麗なデータであったので，結果の精度も良かったと考えられる．

6 感想

シグモイドカーネルの実装でパラメータをどのような値にすれば良いのかよくわからなかったので，実装できなかったのが残念だった．またサンプルデータだけでなく，他のデータにも適用させたかったが，時間があまりなくて試すことができなかった．最終課題では満足のいくところまで実装をやりきりたい．