新人課題

津嶋 佑旗

2019 年 4 月 22 日

1 配列情報処理

1.1 ヒト 21 番染色体の先頭コンティグの DNA 配列を秋山研サーバから取得し，A/T/G/C の各塩基数を出力せよ．

以下のように実行する．実行環境は Python2 で，ghostgw 上での動作を確認している．

```
python 1_1.py /mnt/fs/ohue/newcomer/NT_113952.1.fasta
```

ソースコード 1　実行方法

プログラム本体は 1_1.py である．fasta ファイルの塩基配列について，行ごとに Python の機能で各文字の出現回数をカウントする．実行した結果が以下の通りである．

```
A:59200
T:56195
G:34714
C:34246
```

ソースコード 2　実行結果

1.2 NT_113952.1.fasta の逆相補鎖を生成するプログラムを作成せよ．

以下のように実行する．実行環境は Python2 で，ghostgw 上での動作を確認している．

```
python 1_2.py /mnt/fs/ohue/newcomer/NT_113952.1.fasta
```

ソースコード 3　実行方法

プログラム本体は 1_2.py と rev.py，combine.py である．実行結果は result_1_2.txt として別添する．

1.3 ウィンドウ幅 w，ステップ幅 s で，ウィンドウ内の GC 含量を出力するプログラムを作成し，NT_113952.1.fasta に $w = 1000$，$s = 300$ で適用した結果を gnuplot 等 Excel 以外のツールでプロットせよ．

以下のように実行する．実行環境は Python2 で，ghostgw 上での動作を確認している．

```
python 1_3.py /mnt/fs/ohue/newcomer/NT_113952.1.fasta 1000 300 > data.txt
```

ソースコード 4　実行方法

プログラム本体は 1_3.py である．また，combine.py も使用する．実行結果を gnuplot で描画した結果以下のようになった．

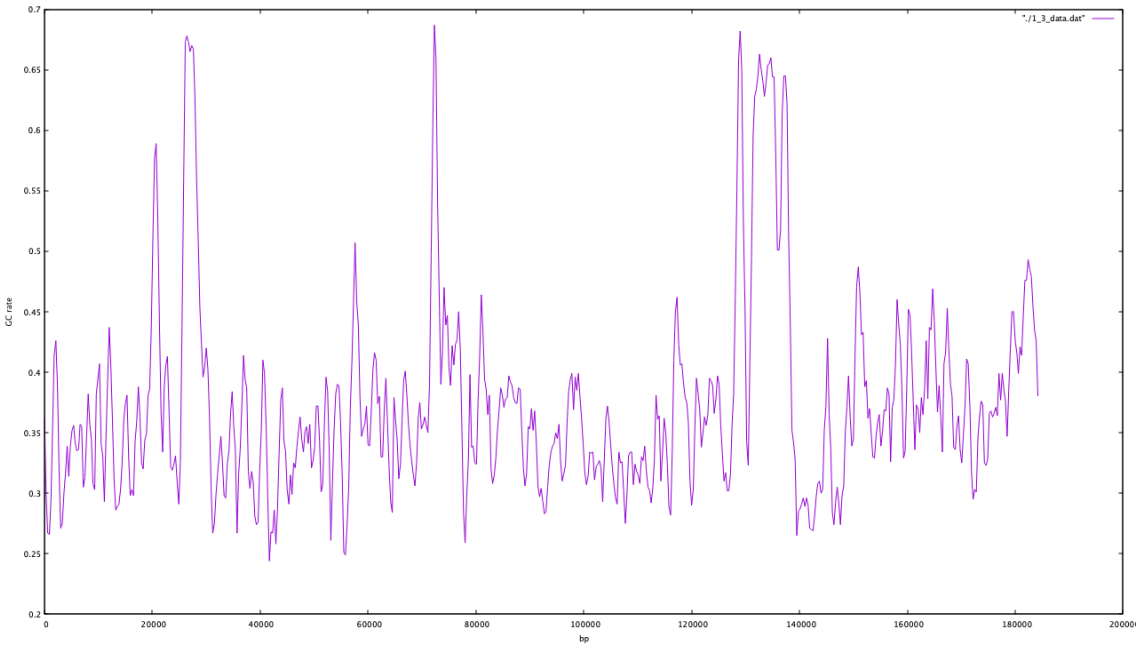

図 1　グラフ

1.4　引数で与えられた部分配列を検索し，何文字目に現れたかを表示するプログラムを作成せよ (逆相補鎖上も検索すること)．部分配列を GAATTC および ATG とし，NT_113952.1.fasta に適用せよ．

以下のように実行する．実行環境は Python2 で，ghostgw 上での動作を確認している．

```
1    python 1_4.py /mnt/fs/ohue/newcomer/NT_113952.1.fasta GAATTC
2    python 1_4.py /mnt/fs/ohue/newcomer/NT_113952.1.fasta ATG
```

ソースコード 5　実行方法

プログラム本体は 1_4.py と findall.py である．また，rev.py と combine.py も使用する．実行した結果が以下の通りである．ただし，ATG の結果については極端に長いため result_1_4_ATG.txt として別添する．

```
1  GAATTC in STRING is below:
2  [395, 4319, 5056, 5334, 10567, 14092, 14296, 15029, 19284, 22535, 25293, 25897, 30824, 34194,
     34625, 37333, 56600, 57464, 58202, 60553, 65241, 66725, 79939, 80725, 82012, 86322, 92914,
     94554, 96877, 97628, 99894, 102214, 103133, 120291, 121414, 121612, 124893, 124901, 125389,
     151452, 151874, 156137, 158738, 166310, 175826, 177983, 180004, 182341]
3  GAATTC in REVERSE STRING is below:
4  [2008, 4345, 6366, 8523, 18039, 25611, 28212, 32475, 32897, 58960, 59448, 59456, 62737, 62935,
     64058, 81216, 82135, 84455, 86721, 87472, 89795, 91435, 98027, 102337, 103624, 104410,
     117624, 119108, 123796, 126147, 126885, 127749, 147016, 149724, 150155, 153525, 158452,
     159056, 161814, 165065, 169320, 170053, 170257, 173782, 179015, 179293, 180030, 183954]
```

ソースコード 6　実行結果 (GAATTC)

1.5　NT_113952.1.fasta を 6 つの読み枠でアミノ酸配列に変換せよ．Stop コドンはアンダースコアで表示すること．

以下のように実行する．実行環境は Python2 で，ghostgw 上での動作を確認している．

```
python 1_5.py /mnt/fs/ohue/newcomer/NT_113952.1.fasta
```

ソースコード 7　実行方法

プログラム本体は 1_5.py と decode.py である．また，rev.py と combine.py も使用する．実行した結果が以下の通りである．

```
PEPTIDE 1
Met Ile Val Met Asn Ser Asn Cys Cys Leu Cys Arg Pro Thr Arg Phe Leu Thr Ser Leu Ser Tyr His Phe
   Leu Leu Ser Tyr Leu Leu Ser Lys Cys Ile Gln Met Lys Gly Cys Gly Glu Cys _
PEPTIDE 2
Met Pro Arg Glu Ile Ser Arg Ser Ser Val Pro Cys _
PEPTIDE 3
Met Pro _
PEPTIDE 4
Met Leu Arg Thr Leu Leu Leu Ile Val _
PEPTIDE 5
Met Gln Asn Lys Phe Ile Arg His _
PEPTIDE 6
Met Gly Arg Lys Asp Lys Ala Ala Ile _
```

ソースコード 8　実行結果

2　タンパク質構造情報処理

2.1　PyMOL ソフトウェアを利用し，ヒトのヘモグロビンの構造をチェインごとに異なる色で表示し，4 量体であることを確認せよ．また，A チェインだけを表示し，タンパク質鎖を cartoon 表示して 2 次構造に従って色付けし，結合するヘムを stick 表示して原子ごとに色分けせよ．

チェインごとに色分けしたヘモグロビンが画像 2 である．

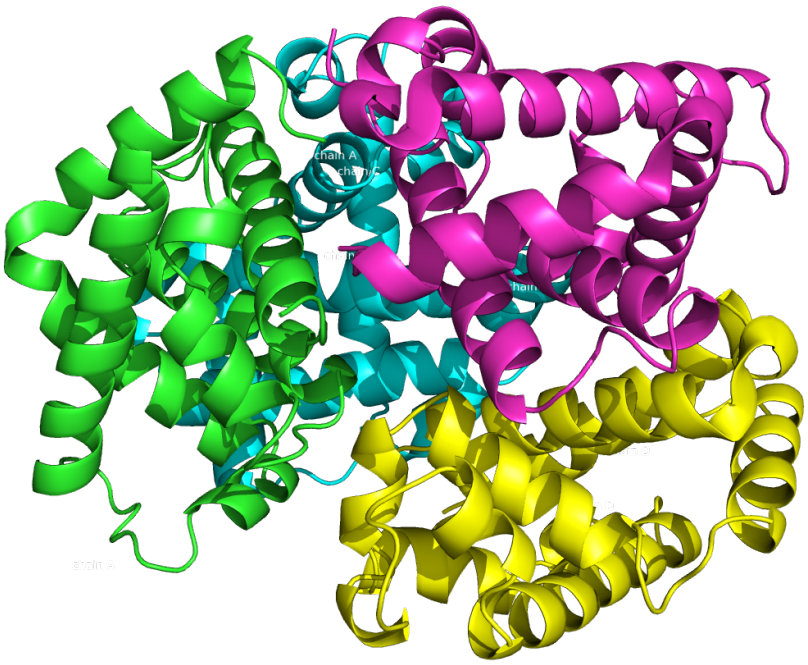

図 2　ヘモグロビン

また，A チェインとヘムを表示したものが画像 3 である．

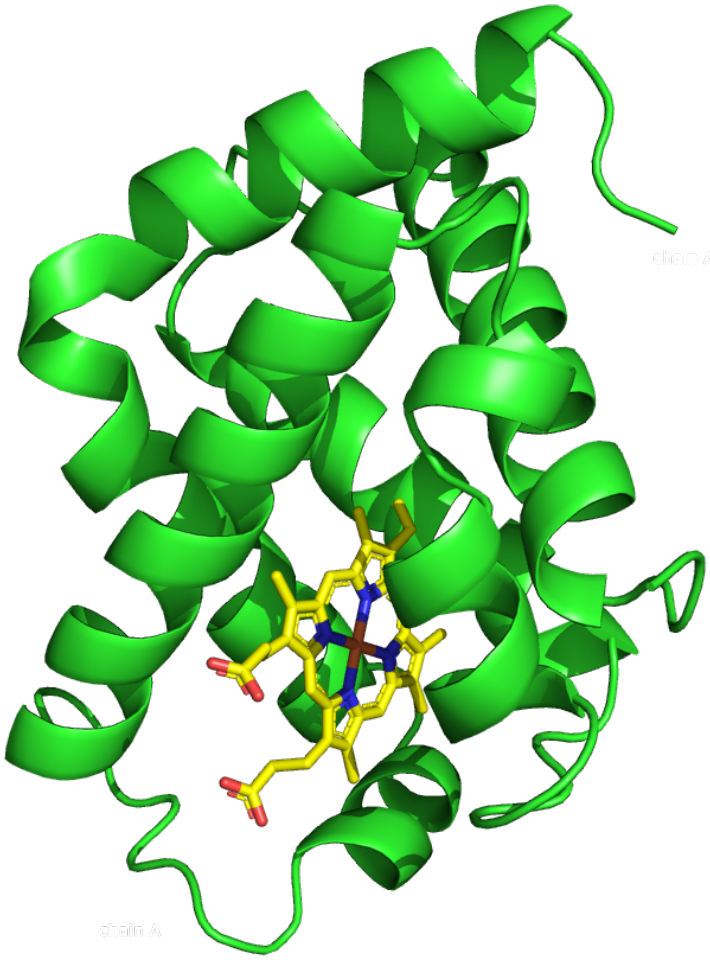

図 3　A チェインとヘム

2.2　PDB ファイル名とチェイン名を引数にして，その回転半径を計算するプログラムを作成せよ．

2.3　上記のプログラムの結果を利用し，PyMOL で重心から回転半径の範囲内にある原子を赤で，範囲外にある原子を青で色付けせよ．

この 2 つは一緒に行い，プログラム 2_7.py を作成した．実行環境は Anaconda3 の Python 2.7 環境である．このプログラムを次のように実行する．

```
1    python 2_7.py ./1BUW.pdb A
```

ソースコード 9　実行方法

結果は次の通りである．

```
Center: (47.0059686237, 33.4827138428, 35.7021762506)
Radius: 14.0781227416
```

ソースコード 10　実行結果

また，描画結果として以下の画像を得た．

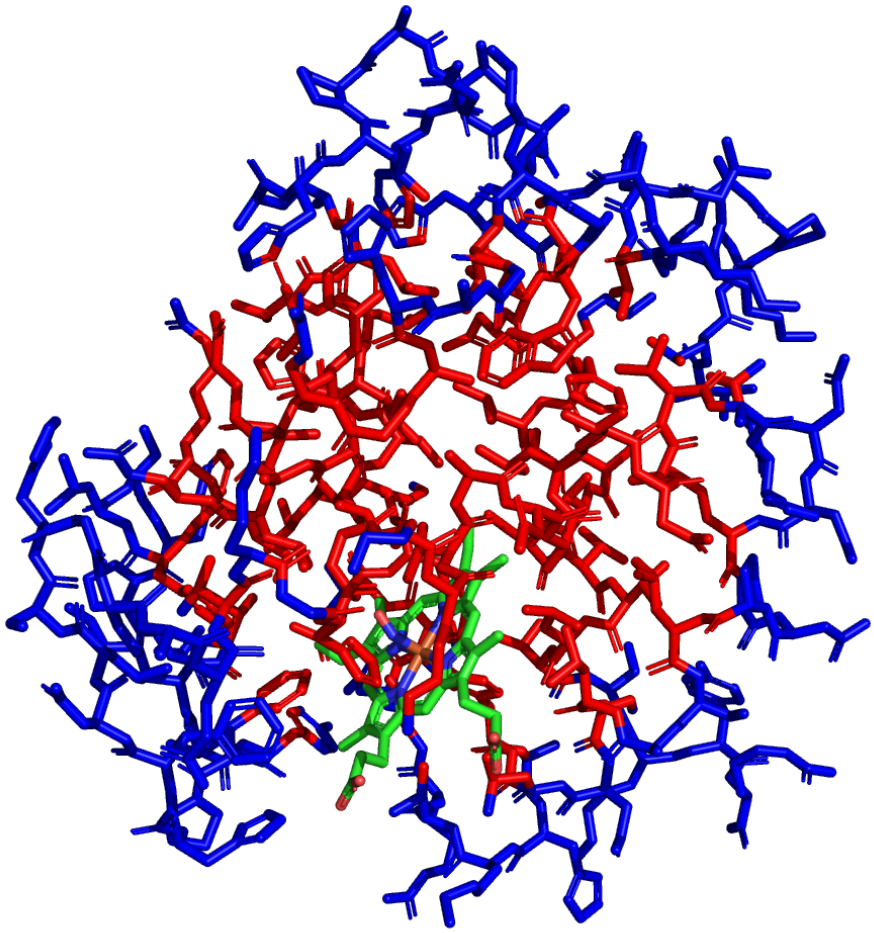

図 4　回転半径

3　機械学習

3.1　Python の scikit-learn の SVM で ionosphere のデータの 10-fold Cross Validation を実施せよ．予測性能として precision, recall, MCC, ROC 曲線の AUC 値 (AUROC), F-score を求めよ．カーネルは RBF カーネルとし，パラメータは適宜定めよ．

プログラム 3_9.py を作成した．Anaconda3 の Python2.7 環境で動作する．また，ionosphere.scale の読み込みにあたって，欠番のパラメータを 0 で補完するなどの処理を行うため，読み込み部分は別のプログラム load_ionosphere.py として分離した．実行した結果を 11 に示す．

```
Set     Preci.  Recall  MCC     AUROC   F-score
1       1.0     1.0     1.0     1.0     1.0
2       1.0     0.95454 0.94146 0.97727 0.97674
3       0.9     1.0     0.89113 0.94117 0.94736
4       0.84    0.95454 0.69186 0.82342 0.89361
5       0.92    1.0     0.87559 0.91666 0.95833
6       1.0     1.0     1.0     1.0     1.0
7       0.91666 0.95652 0.80760 0.89492 0.93617
8       1.0     0.9375  0.75    0.96875 0.96774
9       1.0     0.95238 0.94280 0.97619 0.97560
10      0.90476 0.95    0.82495 0.90833 0.92682
```

ソースコード 11　実行結果

3.2　ionosphere データにおいて，より良い RBF カーネルパラメータ γ とコストパラメータ C の値を探索せよ．評価方法は 10-fold Cross Validation とし，評価基準は AUROC と F-score の 2 通りを試すこと．

プログラム 3_10.py を作成した．Anaconda3 の Python2.7 環境で動作する．実行した結果を 12 に示す．

```
Evaluate with AUROC
{'C': 10, 'gamma': 1}
0.9860481909394953
Evaluate with F-score
{'C': 10, 'gamma': 0.1}
0.9616653503831382
```

ソースコード 12　実行結果

4　創薬情報処理・機械学習

4.1　(準備 1)Python に RDKit をインストールし，化合物ファイル (SDF ファイル) を読み込んで構造式が出力できることを確認せよ．

省略する．

4.2　(準備 2) 論文 Leung SSF, et al. J Chem Inf Model 56: 924-020, 2016. の Supporting Information より，3D SDF ファイル (TXT) と PDF ファイルをダウンロードせよ．Table S1 の実験値「RRCK Log P_{app}」の値をパース (転記) し，CSV ファイル等で準備せよ．

省略する．

4.3　(準備 3) 所望の化合物に対し，RDKit の ECFP4 fingerprint を計算できるようにせよ．

calc_fgprint 関数を作成した．コードは calc_fgprint.py の通りである．

4.4 RRCK Log P_{app} を目的関数，ECFP4 fingerprint を特徴ベクトルとして，Data Set 3(医薬品 104 化合物) に関して回帰 (10-fold Cross Validation) を行う機械学習プログラムを作成せよ．学習器は Support Vector Regression とし，カーネルは RBF カーネルとすること．

スクリプト 4_14.py を作成した．実行結果は 13 の通りである．

```
0.2630136419201868
```

ソースコード 13　実行結果

4.5 10-fold Cross Validation によって，RRCK Log P_{app} の予測値との平均 2 乗誤差 (RMSE) が最も小さくなるパラメータを探索せよ．またそのときの RMSE と R^2 値を求めよ．

スクリプト 4_15.py を作成した．実行結果は 14 の通りである．

```
R2_score : 0.9747505807591622
RMSE : 0.10158867737298335
best_params : {'C': 100, 'gamma': 0.01}
```

ソースコード 14　実行結果

4.6 (15) で決めたパラメータの学習器で，Data Set 1(環状ペプチド 7 化合物)，Data Set 4(環状ペプチド 16 化合物)，Data Set 8(環状ペプチド 22 化合物) の RRCK Log P_{app} の予測値を求めよ．それぞれの Data Set ごとに RMSE と R^2 値を求め，予測値と実験値の散布図を描け．

スクリプト 4_16.py を作成した．実行結果は 15 の通りである．

```
Dataset 1
R2_score : -2.314801681170682
RMSE : 0.6643257009105027
Dataset 3
R2_score : 0.9747505807591622
RMSE : 0.10158867737298335
Dataset 4
R2_score : -2.1150984682038985
RMSE : 0.9198261864774264
Dataset 8
R2_score : 0.12606435549538175
RMSE : 0.4296164950959734
```

ソースコード 15　実行結果

また，実験値と予測値の散布図は出力したファイルから Gnuplot で描画した．図 5，6，7，8 を得た．

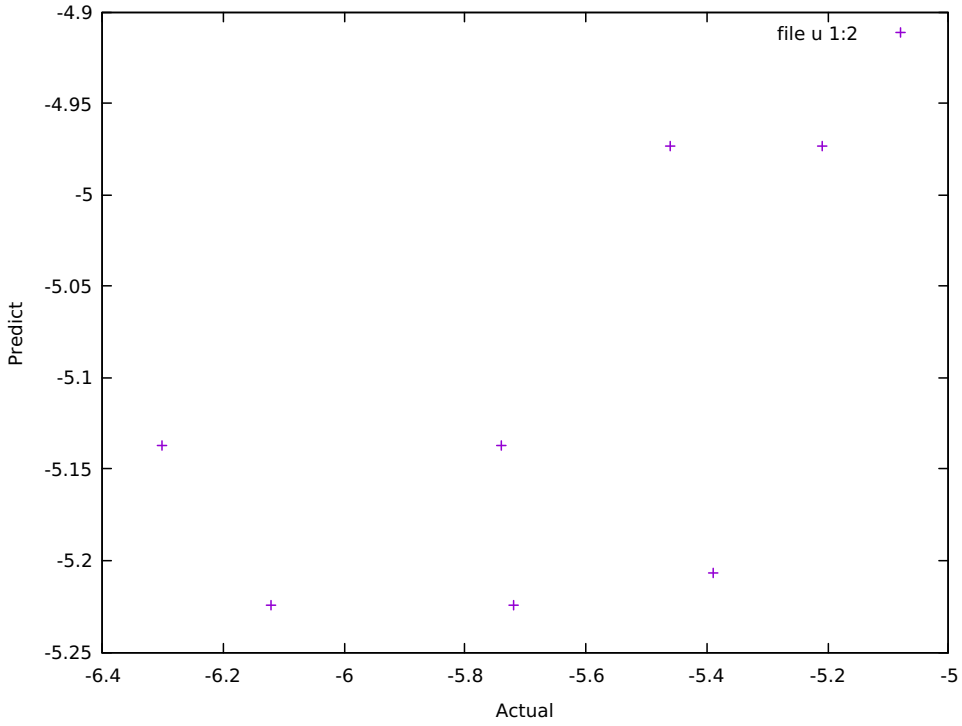

図 5　Data Set 1

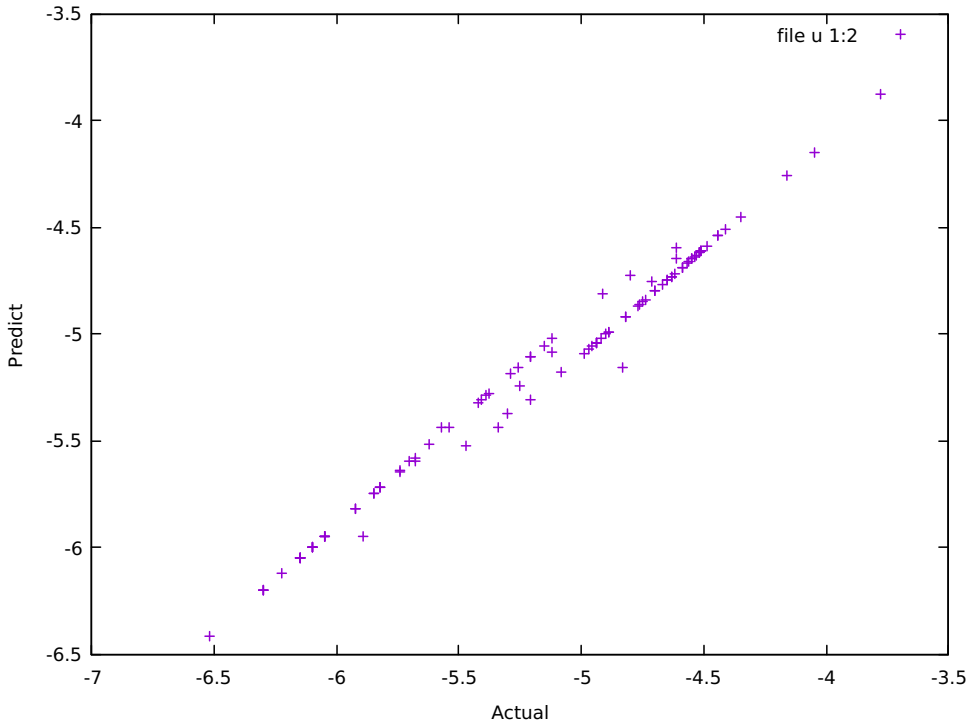

図 6　Data Set 3

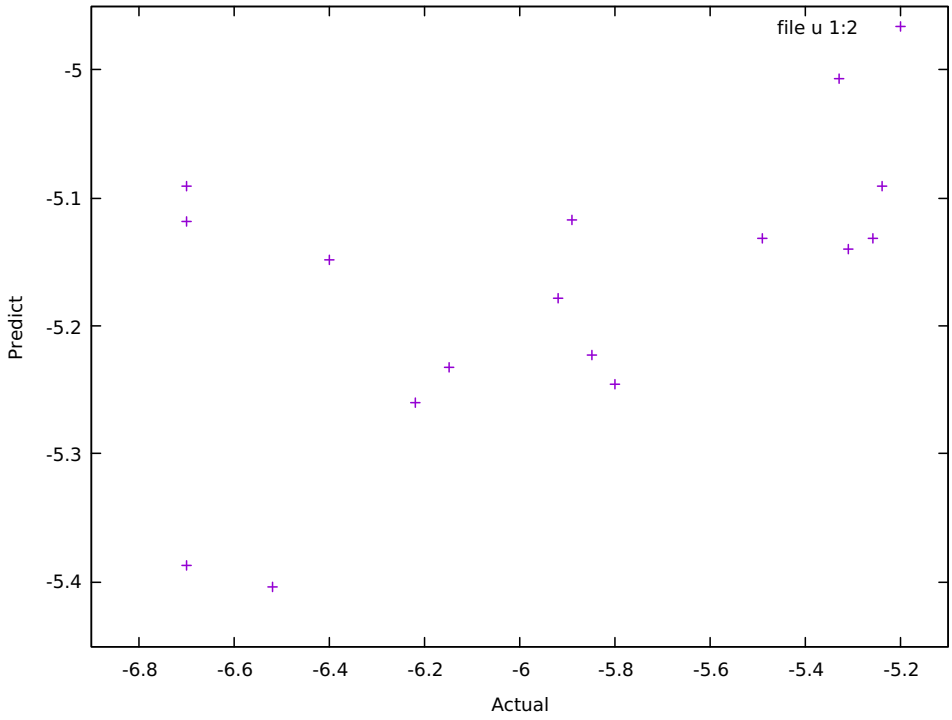

図 7　Data Set 4

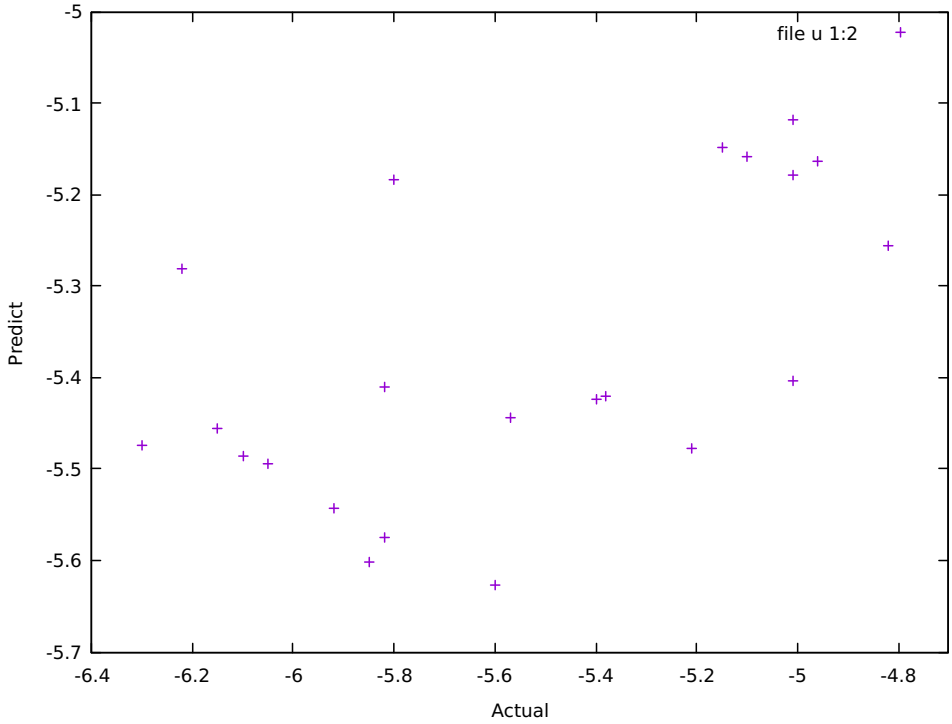

図 8　Data Set 8