Statistical learning project: Report

Due on January 6, 2019

Hoda Mohammadzade

Amirhossein Yousefi

97206984

Project Description:
This project is classification of **emotions** using *EEG* signals recorded in *DEAP* data set.Here 40 video was shown to one person and he was asked to express his emotion as **Arousal** and **Valance** then give it some score from zero to nine as low to high respectively.
Here data is 3D tensor which its shape is $(40, 32, 315)$ which first dimension of this tensor demonstrates the number of films which was show to an individual task and the second dimension illustrates the number of channels was used to record the *EEG* signal and the last dimension related to features which were extracted in 60 seconds or in other words 21 features is extracted at each consecutive 4 seconds which overally lead to 315 samples.
We create three types of data as mentioned in project and then for each of them accuracy for classification and *MAE* error for regression is reported and also channel rejection according to activated part of the brain which is related to the mentioned emotions, considered and at the end window size is changed and then accuracy for each of mentioned modification is reported.
Here we used **SVM , Logostic regression , KNN, LDA, QDA, Adaboost, Random forest and neural network** as a classification method and for regression **Ridge, Lasso,Linear regression, Polynomial regression, Adaboost, Random forest,SVR and KNN** as regression method and the mentioned methods implemented by *scikit* in *google colab* environment.For feature selection *Kbest* is used by different methods such as *f value*.Note that somehow to or more results is shown for one method which results in changing hyper parameters or number of features to be selected.Different types of feature selection methods could be used in my code by uncomment them like *Genetic algorithms* and as we saw in results *PCA* did well in regression but not in classification.
Part a,b,c,d:
After preparing label and data set it's time to predict.
for data1:

project/data1clf/a.png

```
Accuracy of SVM(rbf) is = 0.900000
Best hyper-params for SVM(rbf) due to CV:
{'C': 100}

Accuracy of SVM(poly=3) is = 0.150000
Best hyper-params for SVM(poly=3) due to CV:
{'C': 0.001}

Accuracy of SVM(linear) is = 0.900000
Best hyper-params for SVM(linear) due to CV:
{'C': 1}

Accuracy of AdaBoost is = 0.450000
Best hyper-params for AdaBoost due to CV:
{'n_estimators': 10}
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':

Accuracy of RandomForest is = 0.200000
Best hyper-params for RandomForest due to CV:
{'max_depth': 2, 'n_estimators': 60}
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel_lau
  if sys.path[0] == '':

Accuracy of KNN is = 0.650000
Best hyper-params for KNN due to CV:
{'n_neighbors': 2}
```

Figure 1: classification result for data1

project/data1clf/b.png

```
Accuracy of MLP is = 0.900000
Best hyper-params for MLP due to CV:
{'hidden_layer_sizes': 15}
```

Figure 2: classification result for data1

project/data1clf/c.png

```
Accuracy of logistic regression is = 0.750000
Best hyper-params for logistic regression due to CV:
{'penalty': 'l2'}
```

Figure 3: classification result for data1

project/data1clf/d.png

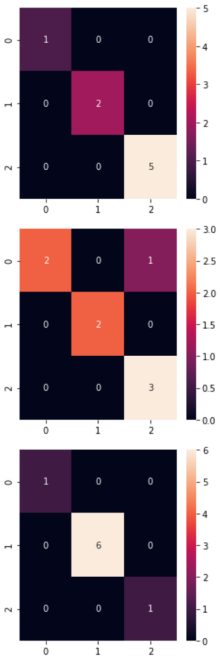
```
Accuracy of linear discriminant analysis is = 0.650000
Best hyper-params for linear discriminant analysis due to CV:
{'solver': 'svd'}
```

Figure 4: classification result for data1

project/data1clf/e.png

```
  if sys.path[0] == '':
Accuracy of quaadratic discriminant analysis is = 0.475000
```

Figure 5: classification result for data1



project/data1clf/f.png

Figure 6: confusion matrix for svm in some cross validation(3of5) for data1

for data2:

```
Accuracy of SVM(rbf) is = 0.150000
Best hyper-params for SVM due to CV:
{'C': 0.0001}

Accuracy of SVM(poly) is = 0.150000
Best hyper-params for SVM due to CV:
{'C': 0.0001}

Accuracy of SVM(linear) is = 0.150000
Best hyper-params for SVM due to CV:
{'C': 0.0001}
```

project/data2clf/a.png

Figure 7: classification result for data2

project/data2clf/b.png

```
Accuracy of AdaBoost is = 0.451667
Best hyper-params for AdaBoost due to CV:
{'n_estimators': 10}
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':

Accuracy of RandomForest is = 0.415000
Best hyper-params for RandomForest due to CV:
{'max_depth': 4, 'n_estimators': 20}
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':
/usr/local/lib/python3.6/dist-packages/ipykernel
  if sys.path[0] == '':

Accuracy of KNN is = 0.293333
Best hyper-params for KNN due to CV:
```

Figure 8: classification result for data2

project/data2clf/c.png

```
/usr/local/lib/python3.6/dist-packages/ip
  if sys.path[0] == '':

Accuracy of MLP is = 0.150000
Best hyper-params for MLP due to CV:
{'hidden_layer_sizes': 5}
```

Figure 9: classification result for data2

project/data2clf/d.png

```
Accuracy of SVM is = 0.150000
Best hyper-params for SVM due to CV:
{'C': 0.0001}
/usr/local/lib/python3.6/dist-packages/sk
  FutureWarning)
```

Figure 10: classification result for data2

project/data2clf/e.png

```
...
    Accuracy of SVM(rbf) is = 0.283333
    Best hyper-params for SVM due to CV:
    {'C': 0.0001, 'gamma': 1e-07}

    Accuracy of SVM(poly) is = 0.203333
    Best hyper-params for SVM due to CV:
    {'C': 0.0001, 'gamma': 0.0001}

    Accuracy of SVM(linear) is = 0.161667
    Best hyper-params for SVM due to CV:
    {'C': 0.0001, 'gamma': 1e-07}
```

Figure 11: classification result for data2

project/data2clf/f.png

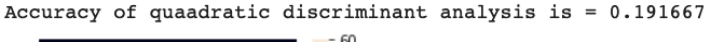


Figure 12: classification result for data2

project/data2clf/g.png

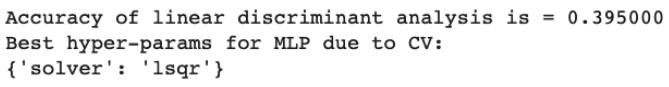


Figure 13: classification result for data2

project/data2clf/h.png

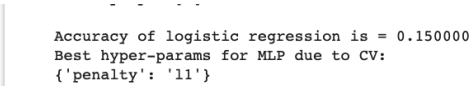


Figure 14: classification result for data2

project/data2clf/i.png

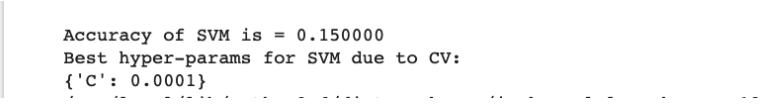
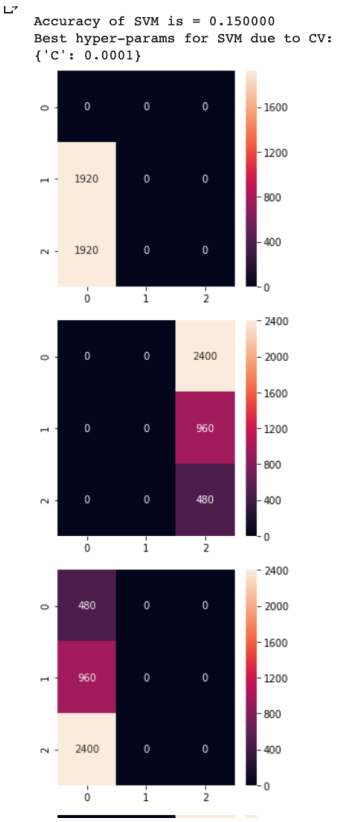


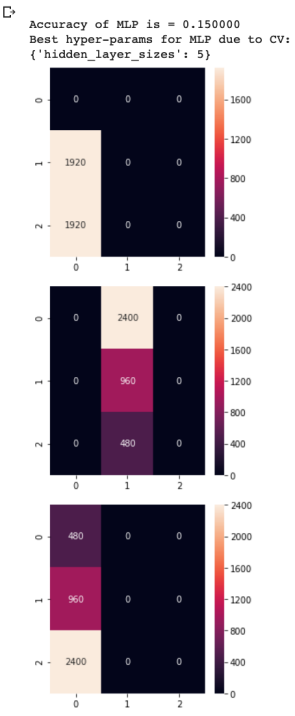
Figure 15: classification result(svm and its kernel is sigmoid) for data2

for data3:



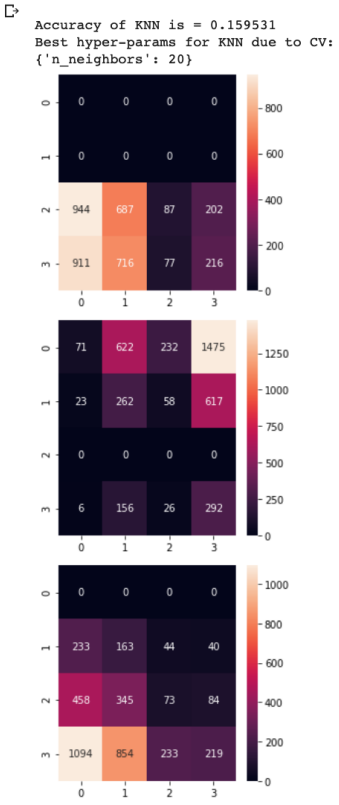
project/data3clf/a.png

Figure 16: classification result for data3



project/data3clf/b.png

Figure 17: classification result for data3

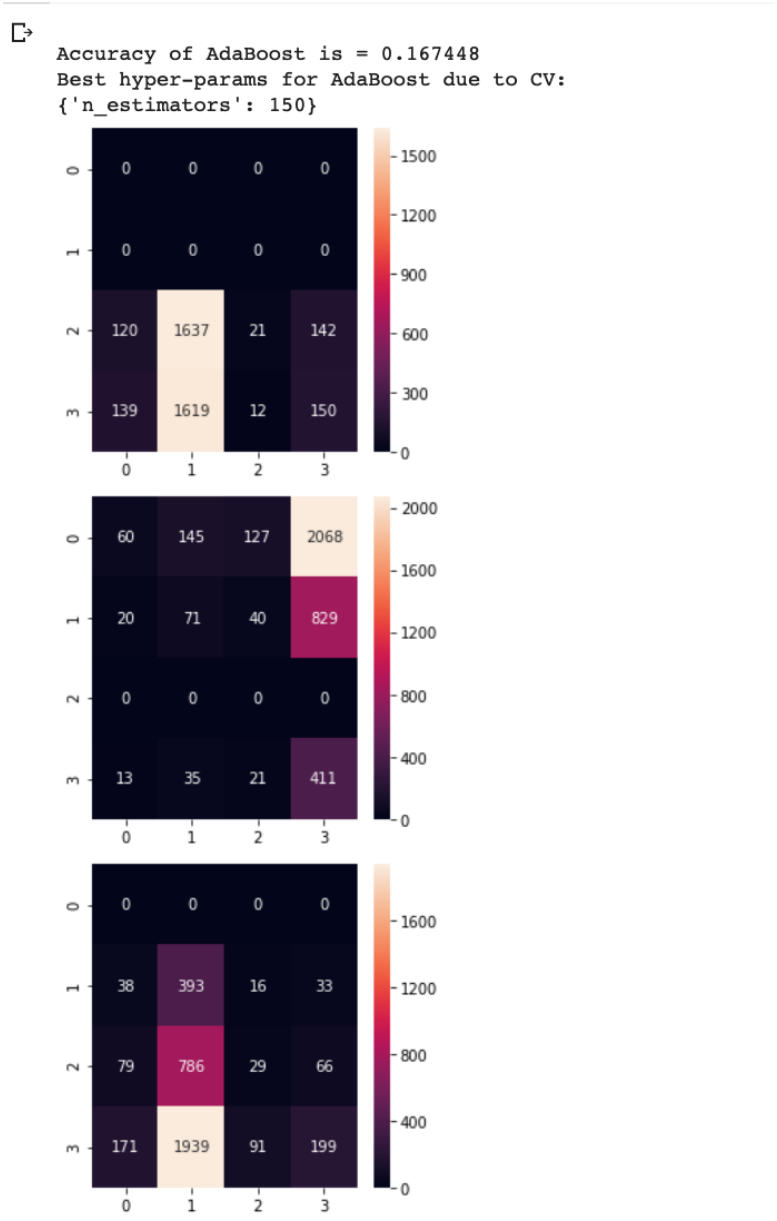


project/data3clf/c.png

Figure 18: classification result for data3

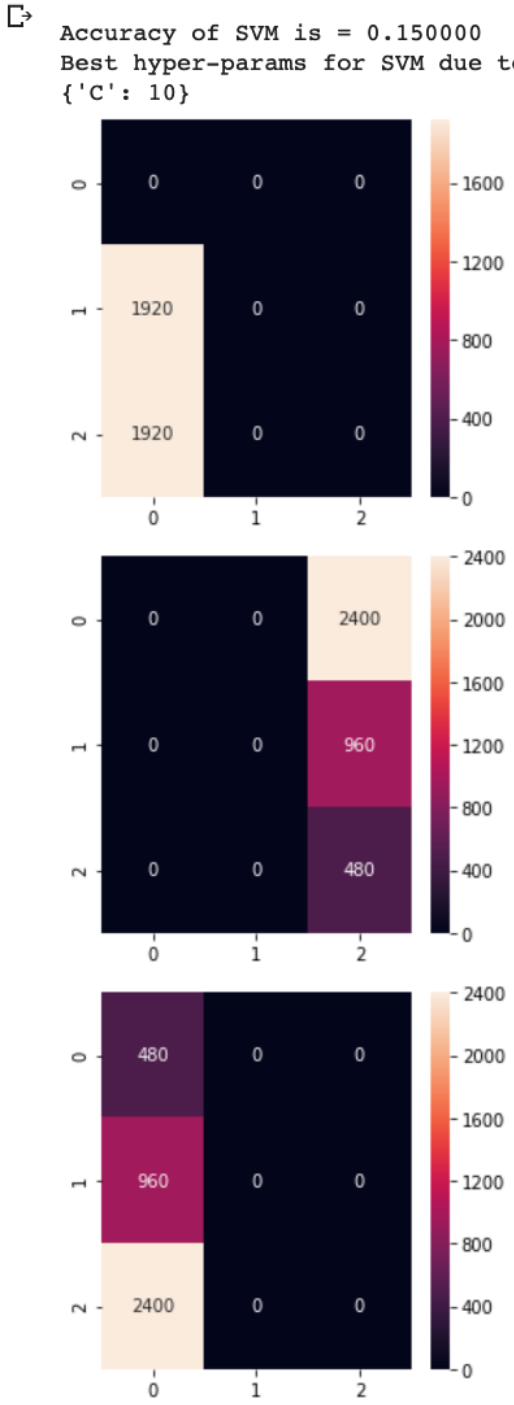
project/data3clf/d.png

Figure 19: classification result for data3



project/data3clf/e.png

Figure 20: classification result for data3



project/data3clf/f.png

Figure 21: classification result for data3

project/data3clf/g.png

```
Accuracy of logistic regression is = 0.150000
Best hyper-params for MLP due to CV:
{'penalty': 'l1'}
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear.")

Accuracy of linear discriminant analysis is = 0.395000
Best hyper-params for MLP due to CV:
{'solver': 'lsqr'}
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")

Accuracy of quaadratic discriminant analysis is = 0.191667
--
```

Figure 22: classification result for data3

Part e:
Now classification results for different window size is plotted.
for window size=12s:

```
Accuracy of SVM is = 0.575000
Best hyper-params for SVM due to CV:
{'C': 10, 'gamma': 1}

Accuracy of SVM is = 0.525000
Best hyper-params for SVM due to CV:
{'C': 0.001, 'gamma': 101}

Accuracy of SVM is = 0.625000
Best hyper-params for SVM due to CV:
{'C': 100, 'gamma': 0}

Accuracy of AdaBoost is = 0.350000
Best hyper-params for AdaBoost due to CV:
{'n_estimators': 70}
/usr/local/lib/python3.6/dist-packages/ipykern
/usr/local/lib/python3.6/dist-packages/ipykern
/usr/local/lib/python3.6/dist-packages/ipykern
/usr/local/lib/python3.6/dist-packages/ipykern
/usr/local/lib/python3.6/dist-packages/ipykern
/usr/local/lib/python3.6/dist-packages/ipykern

Accuracy of RandomForest is = 0.525000
Best hyper-params for RandomForest due to CV:
{'max_depth': 4, 'n_estimators': 20}
/usr/local/lib/python3.6/dist-packages/ipykern
/usr/local/lib/python3.6/dist-packages/ipykern
/usr/local/lib/python3.6/dist-packages/ipykern
/usr/local/lib/python3.6/dist-packages/ipykern

Accuracy of KNN is = 0.550000
Best hyper-params for KNN due to CV:
{'n_neighbors': 2}
/usr/local/lib/python3.6/dist-packages/sklearn
```

project/w1/a.png

Figure 23: classification result for data1 and window size=12

```
Accuracy of MLP is = 0.425000
Best hyper-params for MLP due to CV:
{'hidden_layer_sizes': 15}
```

project/w1/b.png

Figure 24: classification result for data1 and window size=12

project/w1/c.png

```
Accuracy of logistic regression is = 0.325000
Best hyper-params for logistic regression due to CV:
{'penalty': 'l1'}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/sklearn/covariance/empir
  warnings.warn("Only one sample available. "
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/sklearn/covariance/empir
  warnings.warn("Only one sample available. "
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28

Accuracy of linear discriminant analysis is = 0.475000
Best hyper-params for linear discriminant analysis due to CV:
{'solver': 'lsqr'}
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_ana
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28

Accuracy of quaadratic discriminant analysis is = 0.150000
```

Figure 25: classification result for data1 and window size=12

for window size=20s:

project/w2/a.png

```
Accuracy of SVM is = 0.575000
Best hyper-params for SVM due to CV:
{'C': 10, 'gamma': 1}

Accuracy of SVM is = 0.525000
Best hyper-params for SVM due to CV:
{'C': 0.001, 'gamma': 101}

Accuracy of SVM is = 0.625000
Best hyper-params for SVM due to CV:
{'C': 100, 'gamma': 0}

Accuracy of AdaBoost is = 0.375000
Best hyper-params for AdaBoost due to CV:
{'n_estimators': 50}
/usr/local/lib/python3.6/dist-packages/ipykernel_
/usr/local/lib/python3.6/dist-packages/ipykernel_
/usr/local/lib/python3.6/dist-packages/ipykernel_
/usr/local/lib/python3.6/dist-packages/ipykernel_
/usr/local/lib/python3.6/dist-packages/ipykernel_
/usr/local/lib/python3.6/dist-packages/ipykernel_

Accuracy of RandomForest is = 0.500000
Best hyper-params for RandomForest due to CV:
{'max_depth': 4, 'n_estimators': 20}
/usr/local/lib/python3.6/dist-packages/ipykernel_
/usr/local/lib/python3.6/dist-packages/ipykernel_
/usr/local/lib/python3.6/dist-packages/ipykernel_
/usr/local/lib/python3.6/dist-packages/ipykernel_

Accuracy of KNN is = 0.550000
Best hyper-params for KNN due to CV:
{'n_neighbors': 2}
```

Figure 26: classification result for data1 and window size=20

project/w2/b.png

```
/usr/local/lib/python3.6/dist-packages/ipykern

Accuracy of MLP is = 0.375000
Best hyper-params for MLP due to CV:
{'hidden_layer_sizes': 30}
/usr/local/lib/python3.6/dist-packages/sklearn
```

Figure 27: classification result for data1 and window size=20

project/w2/c.png

```
  "this warning.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher

Accuracy of logistic regression is = 0.325000
Best hyper-params for logistic regression due to CV:
{'penalty': 'l1'}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher
```

Figure 28: classification result for data1 and window size=20

project/w2/c.png

```
  "this warning.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher

Accuracy of logistic regression is = 0.325000
Best hyper-params for logistic regression due to CV:
{'penalty': 'l1'}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher
```

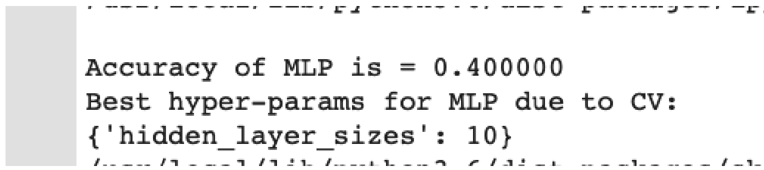
Figure 29: classification result for data1 and window size=20

for window size=8s:

project/w3/a.png

```
Accuracy of SVM is = 0.650000
Best hyper-params for SVM due to CV:
{'C': 100, 'gamma': 1}

Accuracy of SVM is = 0.550000
Best hyper-params for SVM due to CV:
{'C': 0.1, 'gamma': 101}

Accuracy of SVM is = 0.550000
Best hyper-params for SVM due to CV:
{'C': 100, 'gamma': 0}

Accuracy of AdaBoost is = 0.425000
Best hyper-params for AdaBoost due to CV:
{'n_estimators': 20}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru

Accuracy of RandomForest is = 0.550000
Best hyper-params for RandomForest due to CV:
{'max_depth': 4, 'n_estimators': 10}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: Ru

Accuracy of KNN is = 0.575000
Best hyper-params for KNN due to CV:
{'n_neighbors': 2}
```

Figure 30: classification result for data1 and window size=8

project/w3/b.png 

```
Accuracy of MLP is = 0.400000
Best hyper-params for MLP due to CV:
{'hidden_layer_sizes': 10}
```

Figure 31: classification result for data1 and window size=8

project/w3/c.png 

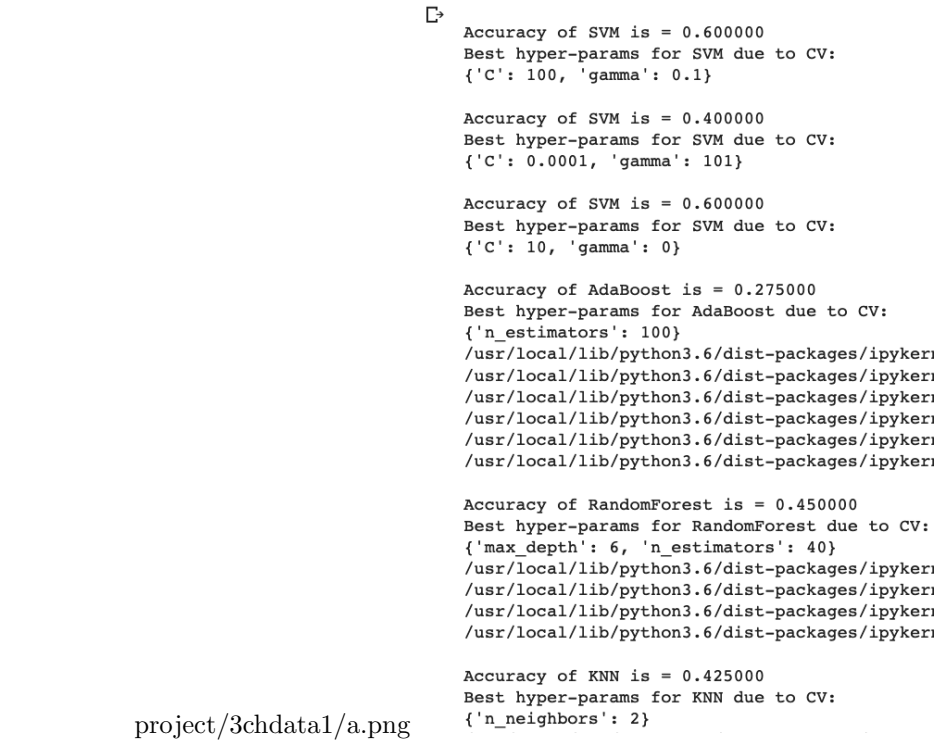
```
Accuracy of logistic regression is = 0.350000
Best hyper-params for logistic regression due to CV:
{'penalty': 'l2'}
/usr/local/lib/python3.6/dist-packages/sklearn/covariance/empi
  warnings.warn("Only one sample available. "
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/sklearn/covariance/empi
  warnings.warn("Only one sample available. "
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2

Accuracy of linear discriminant analysis is = 0.525000
Best hyper-params for linear discriminant analysis due to CV:
{'solver': 'svd'}
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_an
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_an
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_an
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_an
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_an
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2

Accuracy of quaadratic discriminant analysis is = 0.450000
```

Figure 32: classification result for data1 and window size=8

Part f:

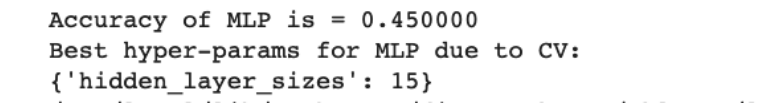
Now it's time to channel selection.Channel *O1,O2,Oz* is selected and in some case there is improvement in accuracy.

Channel selection for data1 :



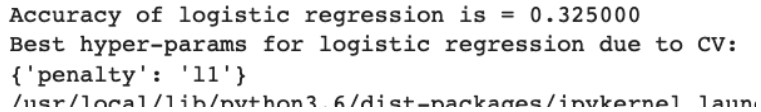
project/3chdata1/a.png

Figure 33: classification result for data1 with channel selection



project/3chdata1/b.png

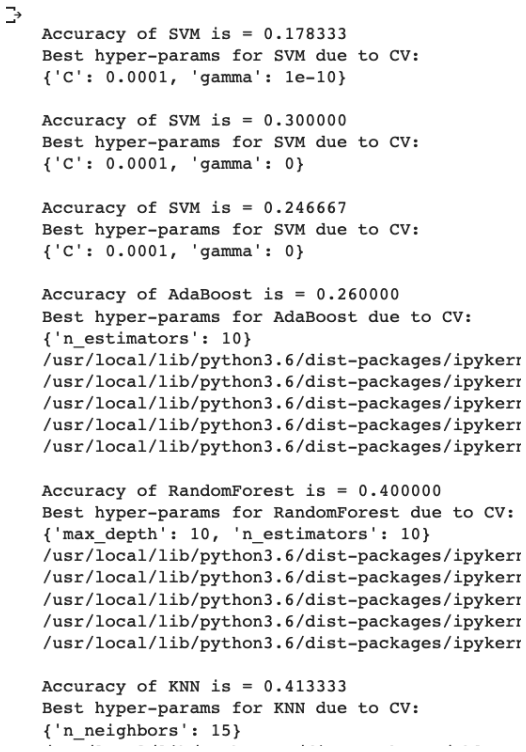
Figure 34: classification result for data1 with channel selection



project/3chdata1/c.png

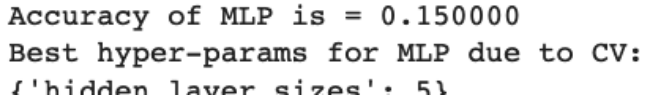
Figure 35: classification result for data1 with channel selection

Channel selection for data2 :



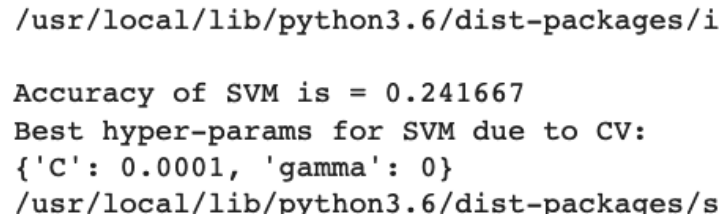
project/3chdata2/a.png

Figure 36: classification result for data2 with channel selection



project/3chdata2/b.png

Figure 37: classification result for data2 with channel selection



project/3chdata2/c.png

Figure 38: classification result for data2 with channel selection

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R

Accuracy of logistic regression is = 0.248333
Best hyper-params for MLP due to CV:
{'penalty': 'l1'}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R

Accuracy of linear discriminant analysis is = 0.426667
Best hyper-params for MLP due to CV:
{'solver': 'svd'}
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_analys
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_analys
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_analys
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_analys
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R
/usr/local/lib/python3.6/dist-packages/sklearn/discriminant_analys
  warnings.warn("Variables are collinear")
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28: R

Accuracy of quaadratic discriminant analysis is = 0.391667
```

project/3chdata2/d.png

Figure 39: classification result for data2 with channel selection

Channel selection for data3 :

```
Accuracy of SVM is = 0.144444
Best hyper-params for SVM due to CV:
{'C': 0.001, 'gamma': 0}

Accuracy of SVM is = 0.149444
Best hyper-params for SVM due to CV:
{'C': 0.0001, 'gamma': 0}

Accuracy of SVM is = 0.150000
Best hyper-params for SVM due to CV:
{'C': 0.0001, 'gamma': 0}

Accuracy of AdaBoost is = 0.179444
Best hyper-params for AdaBoost due to CV:
{'n_estimators': 150}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2

Accuracy of RandomForest is = 0.160556
Best hyper-params for RandomForest due to CV:
{'max_depth': 2, 'n_estimators': 20}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2

Accuracy of KNN is = 0.175000
Best hyper-params for KNN due to CV:
{'n_neighbors': 20}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2

Accuracy of MLP is = 0.150000
Best hyper-params for MLP due to CV:
{'hidden_layer_sizes': 5}
```

project/3chdata3/a.png

Figure 40: classification result for data3 with channel selection

project/3chdata3/b.png

```
this warning. , FutureWarning)

Accuracy of SVM is = 0.156667
Best hyper-params for SVM due to CV:
{'C': 0.0001, 'gamma': 0}
```

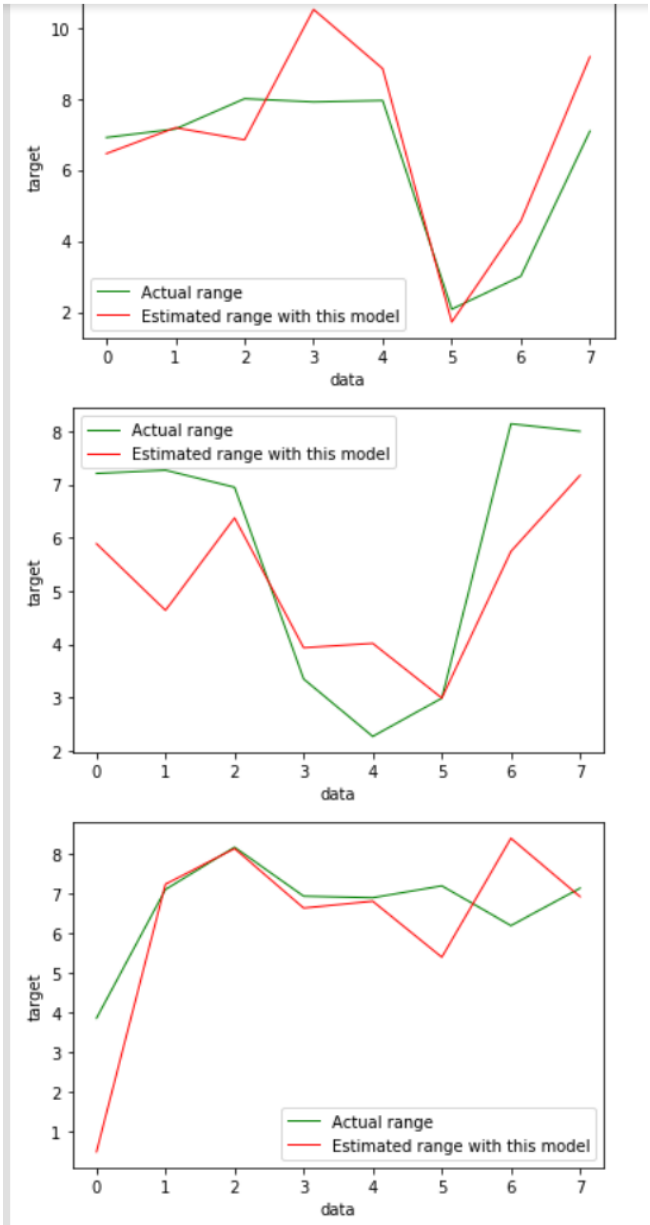
Figure 41: classification result for data3 with channel selection

project/3chdata3/c.png

```
Accuracy of logistic regression is = 0.150000
Best hyper-params for MLP due to CV:
{'penalty': 'l2'}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28

Accuracy of linear discriminant analysis is = 0.158333
Best hyper-params for MLP due to CV:
{'solver': 'svd'}
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:28

Accuracy of quaadratic discriminant analysis is = 0.186667
```

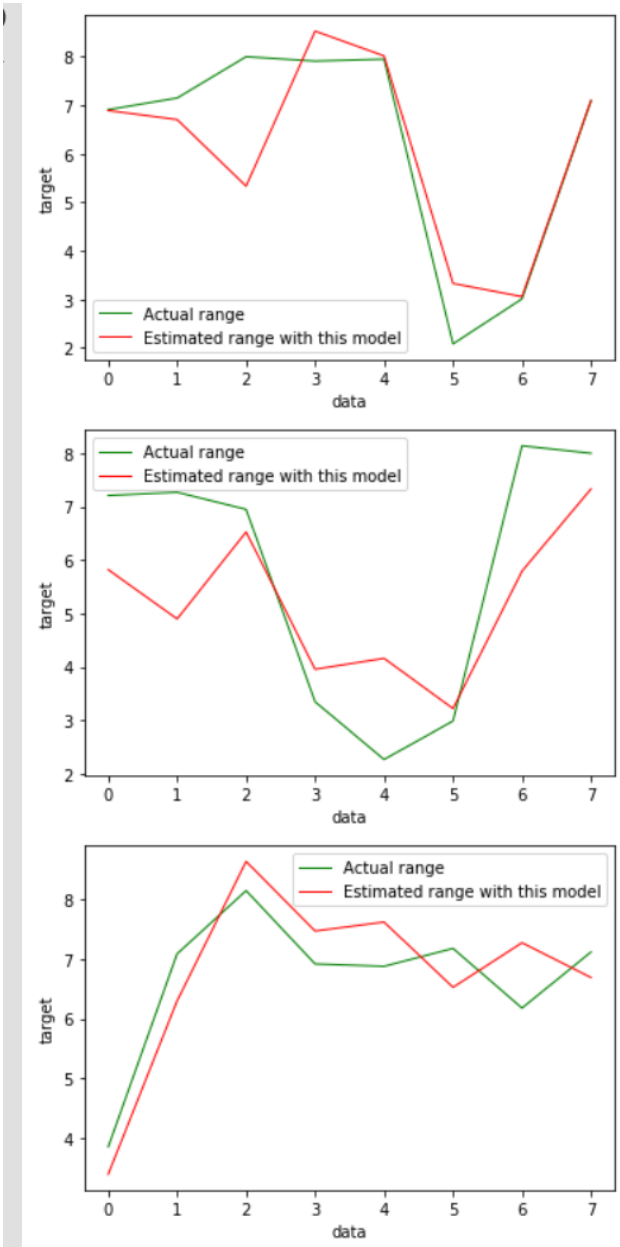
Figure 42: classification result for data3 with channel selection

Part g:
Now prediction is done due to mentioned regression methods.For each of *Arousal* and *Valance* we fit regression methods.
Regression for data1:
Arousal:



project/data1rega/a.png `MAE of LinReg is = 1.002672`

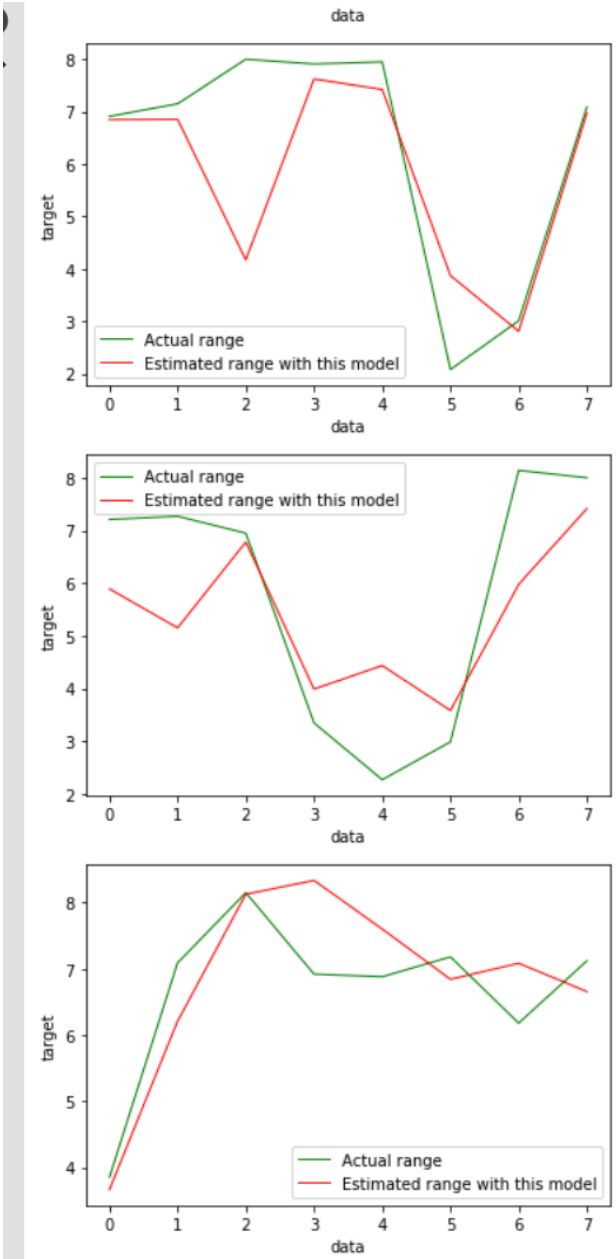
Figure 43: regression result(Arousal) for data1




```
MAE of Ridge is = 0.859211
Best hyper-params for Ridge due to CV:
{'alpha': 1}
```

project/data1rega/b.png

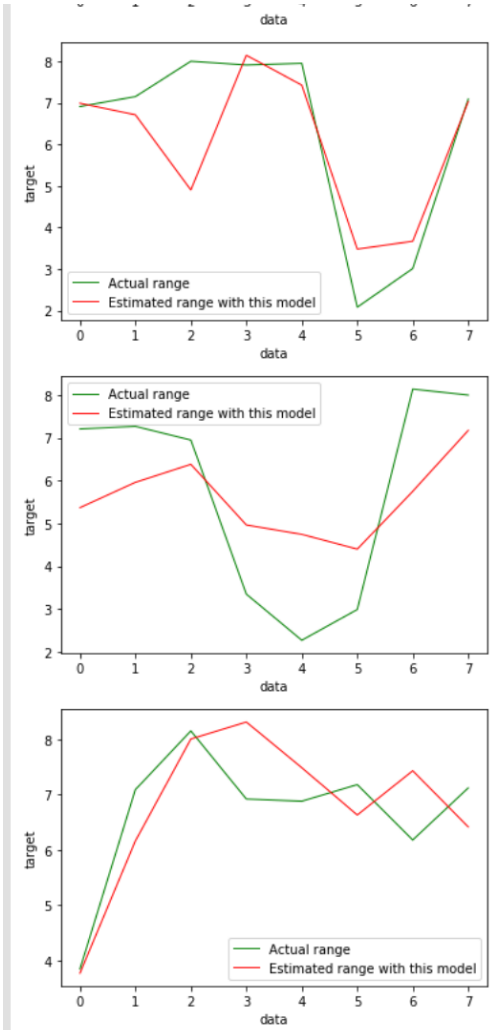
Figure 44: regression result(Arousal) for data1




```
MAE of Lasso is = 0.955645
Best hyper-params for Lasso due to CV:
{'alpha': 0.1}
```

project/data1rega/c.png

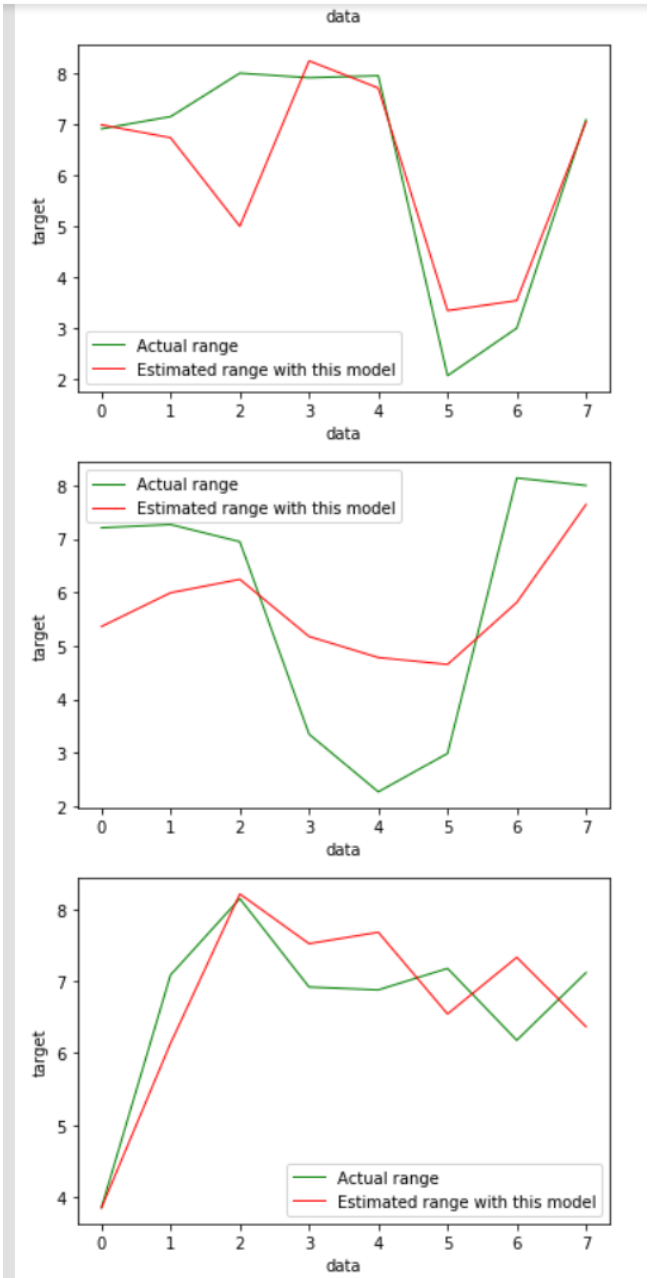
Figure 45: regression result(Arousal) for data1



project/data1rega/d.png

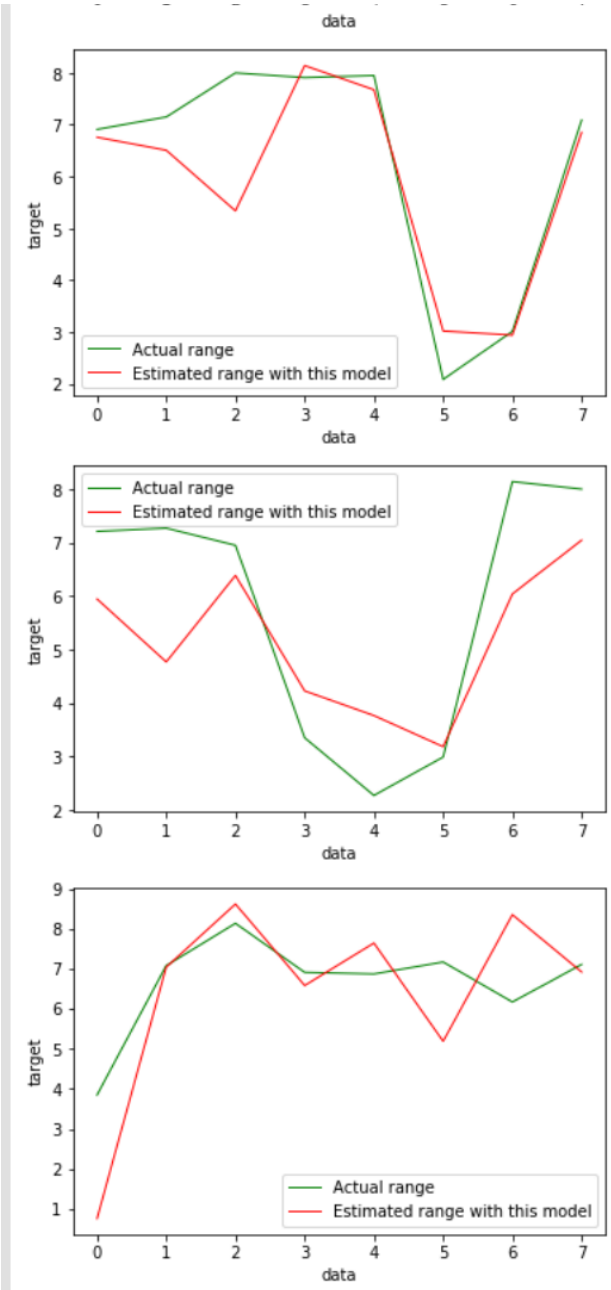
```
MAE of polynomial reg2 is = 1.014250
```

Figure 46: regression result(Arousal) for data1



project/data1rega/e.png

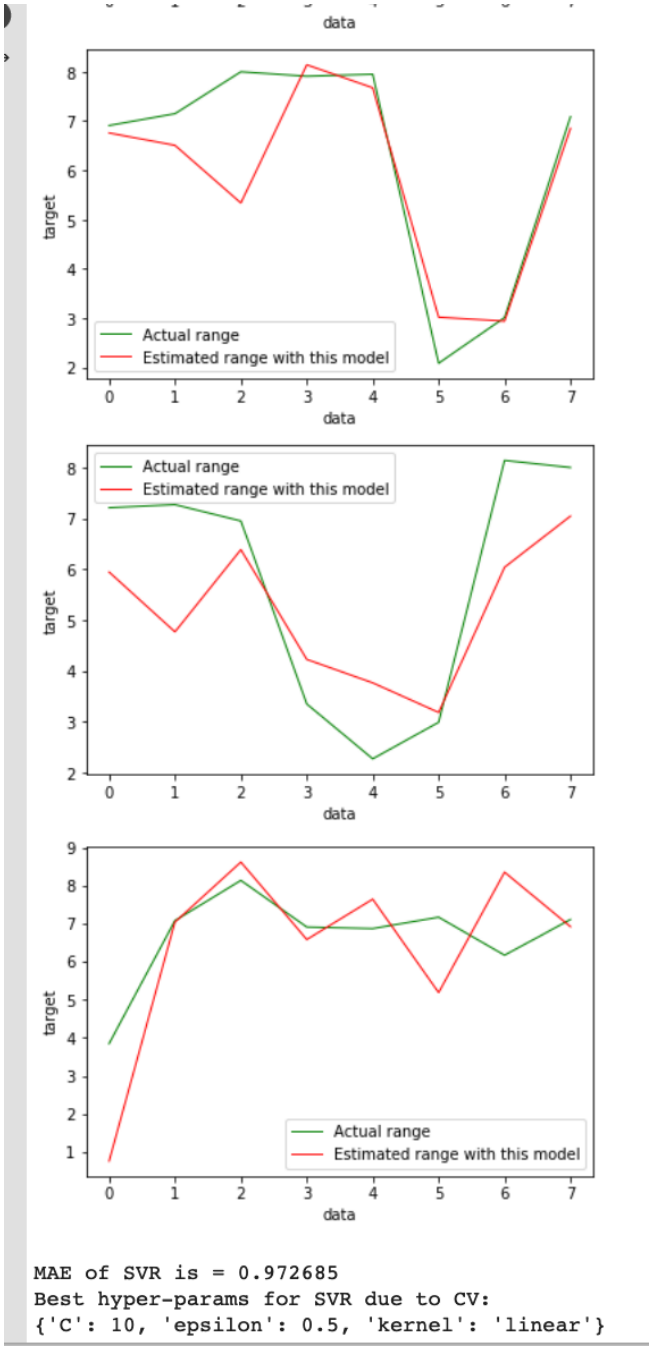
Figure 47: regression result(Arousal) for data1



project/data1rega/f.png

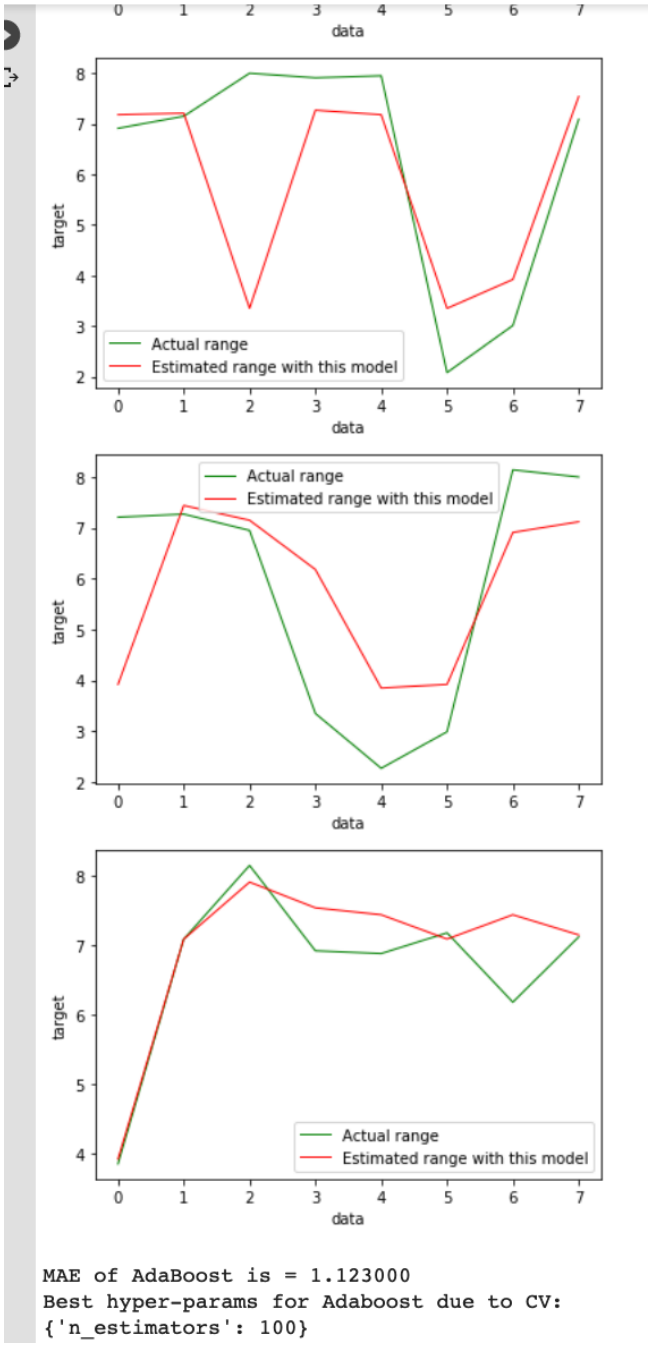
```
MAE of SVR is = 0.972685
```

Figure 48: regression result(Arousal) for data1



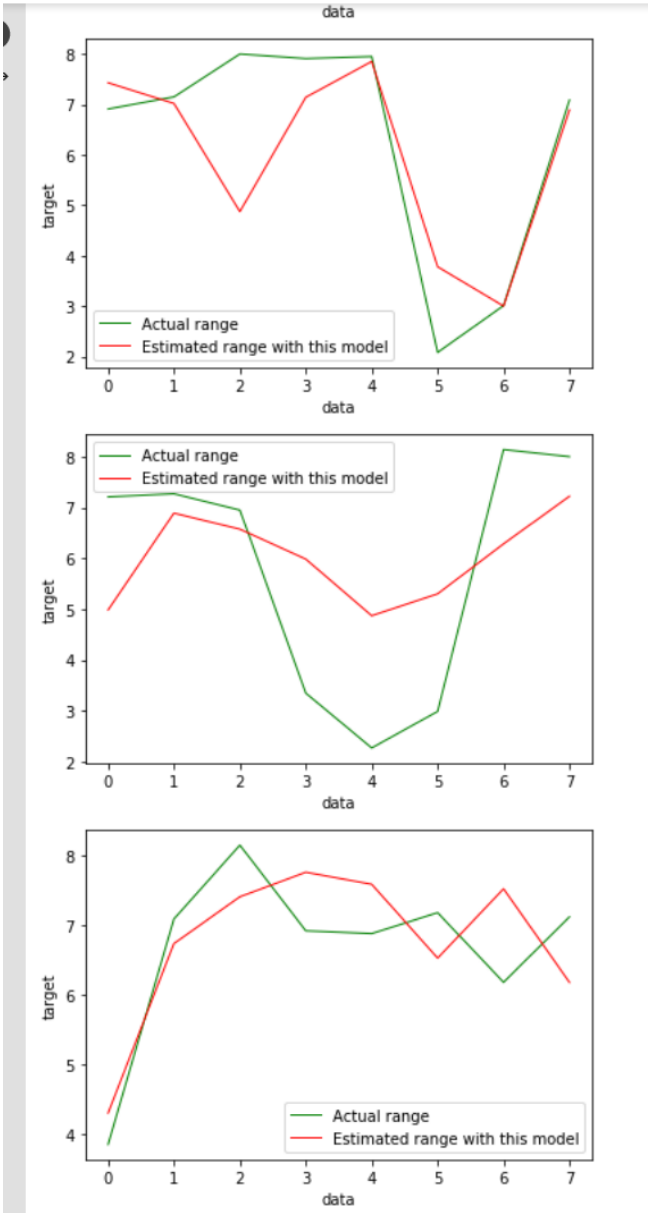
project/data1rega/g.png

Figure 49: regression result(Arousal) for data1



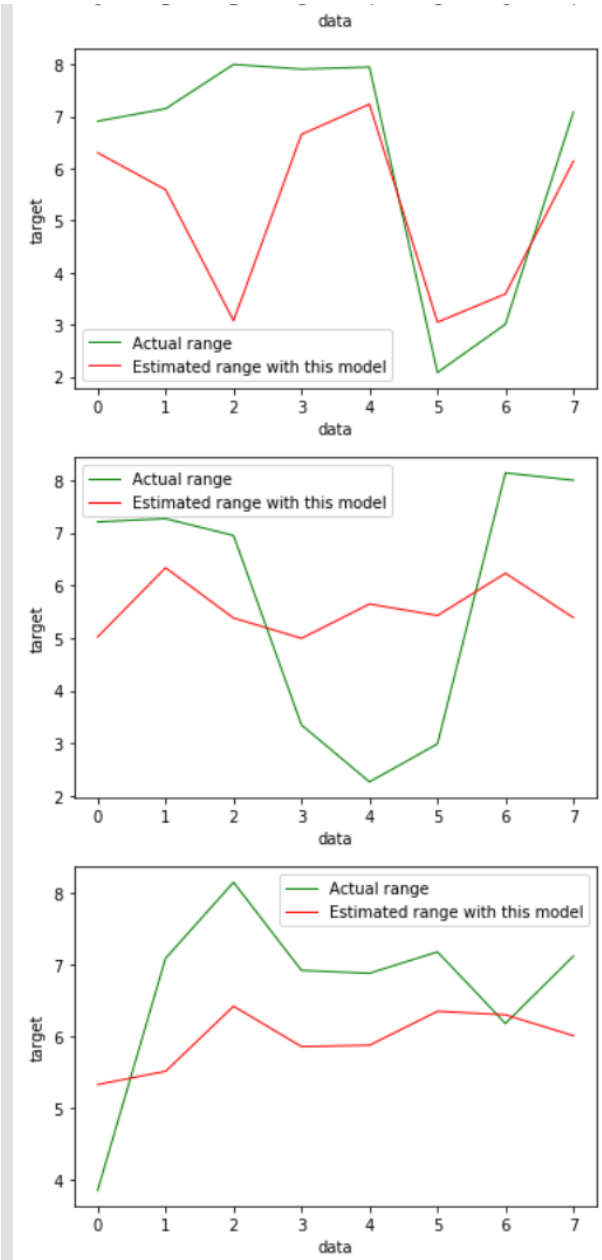
project/data1rega/h.png

Figure 50: regression result(Arousal) for data1



project/data1rega/i.png

Figure 51: regression result(Arousal) for data1

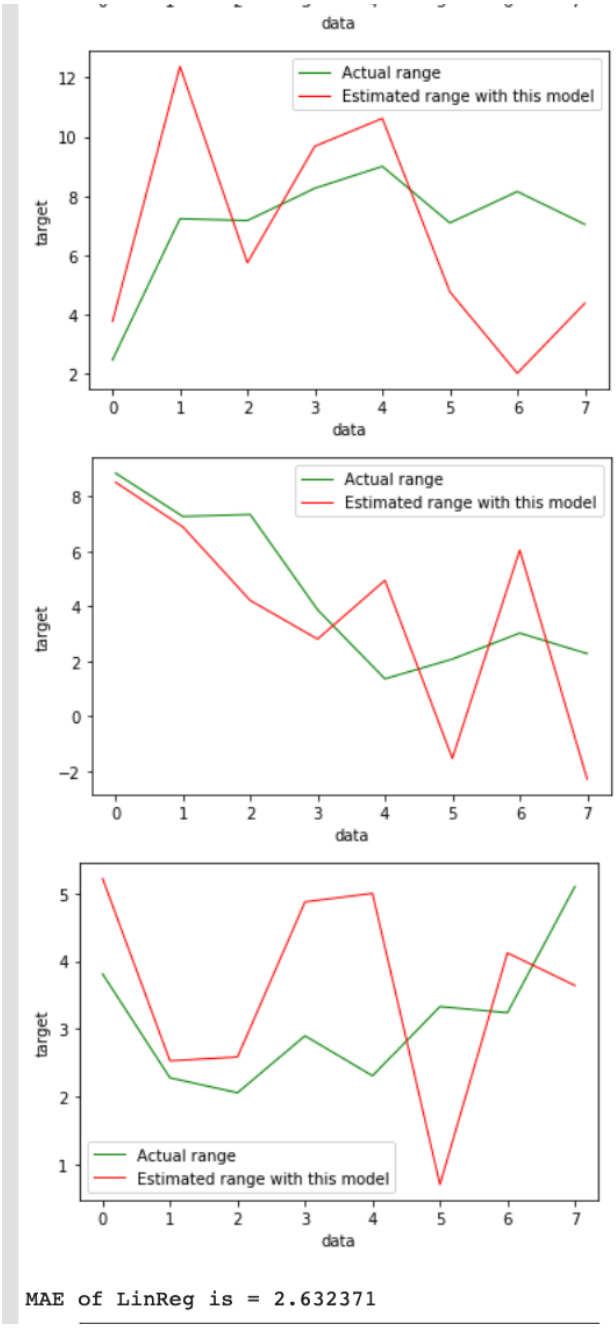



```
MAE of RandomForest is = 1.643630
Best hyper-params for RandomForest due to CV:
{'max_depth': 12, 'n_estimators': 100}
```

project/data1rega/j.png

Figure 52: regression result(Arousal) for data1

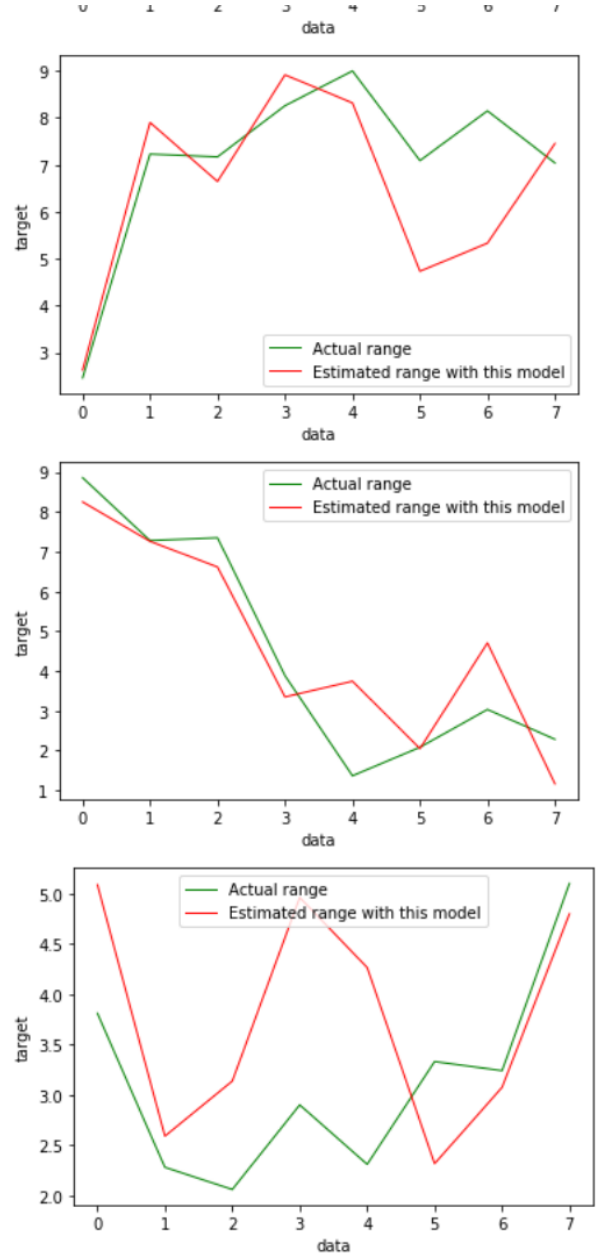
Valance:



project/data1regv/a.png

```
MAE of LinReg is = 2.632371
```

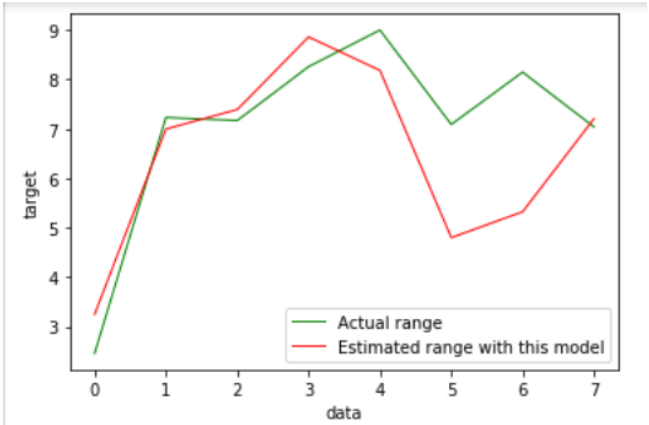
Figure 53: regression result(Valance) for data1



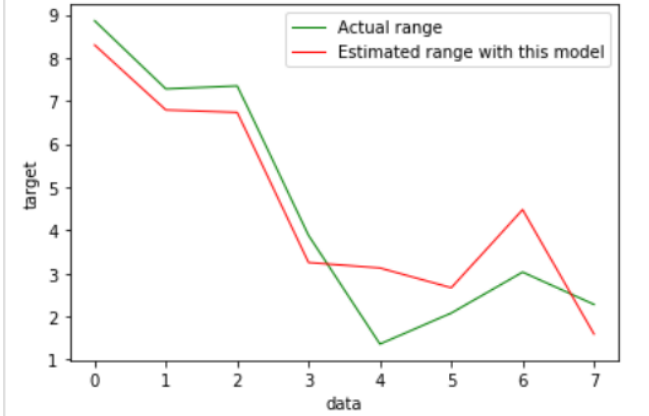

```
MAE of Ridge is = 1.192307
Best hyper-params for Ridge due to CV:
{'alpha': 1}
```

project/data1regv/b.png

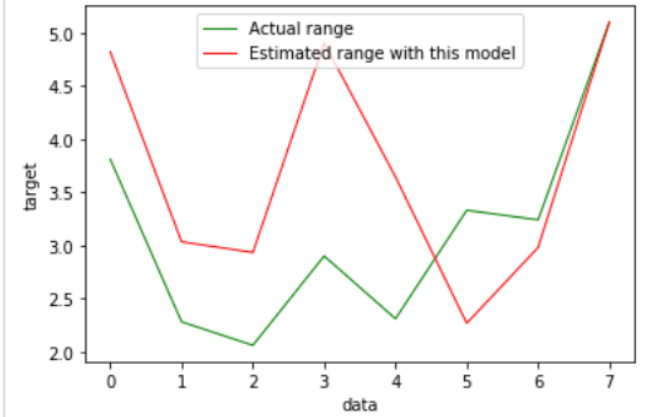
Figure 54: regression result(Valance) for data1




```
/usr/local/lib/python3.6/dist-packages/sklearn
  positive)
/usr/local/lib/python3.6/dist-packages/sklearn
  positive)
```



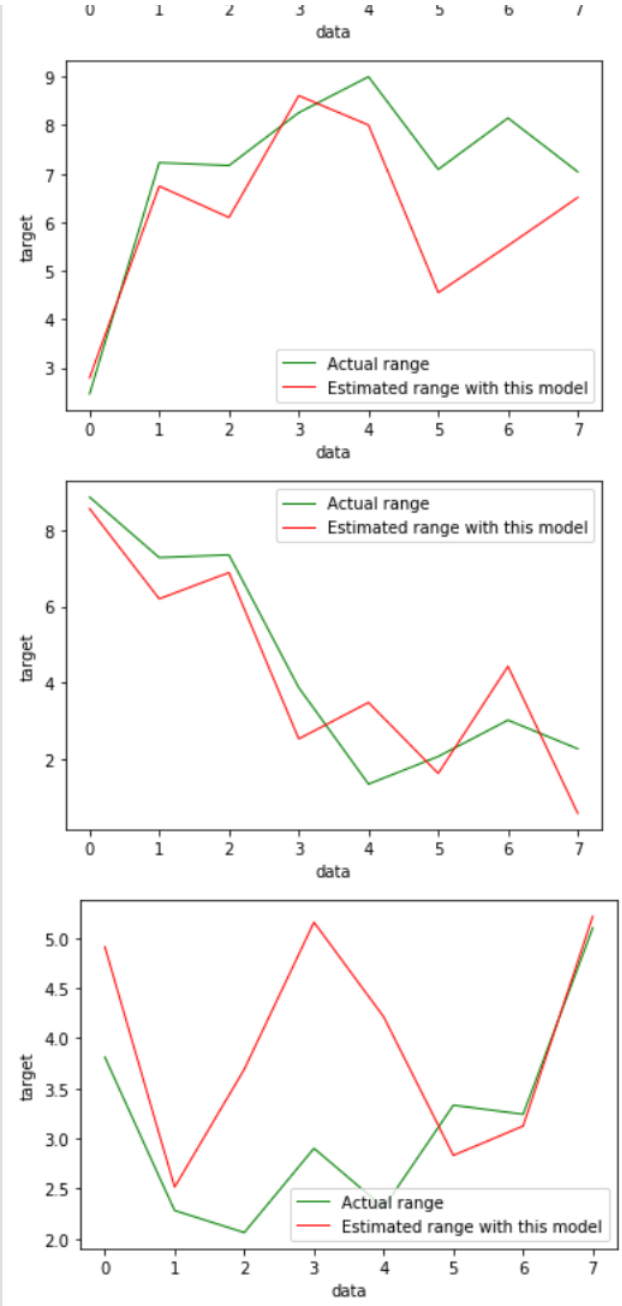

```
/usr/local/lib/python3.6/dist-packages/sklearn
  positive)
```




```
MAE of Lasso is = 1.207148
Best hyper-params for Lasso due to CV:
{'alpha': 0.1}
```

project/data1regv/c.png

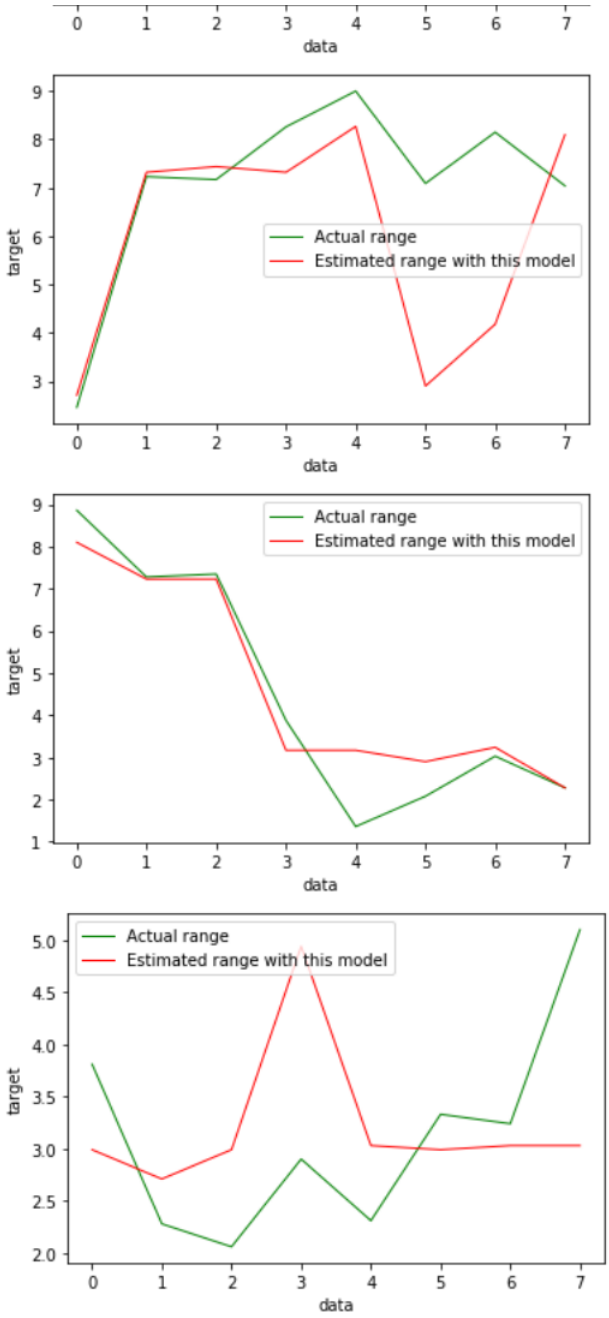
Figure 55: regression result(Valance) for data1




```
MAE of SVR is = 1.313911
Best hyper-params for SVR due to CV:
{'C': 1, 'epsilon': 0.1, 'kernel': 'linear'}
```

project/data1regv/d.png

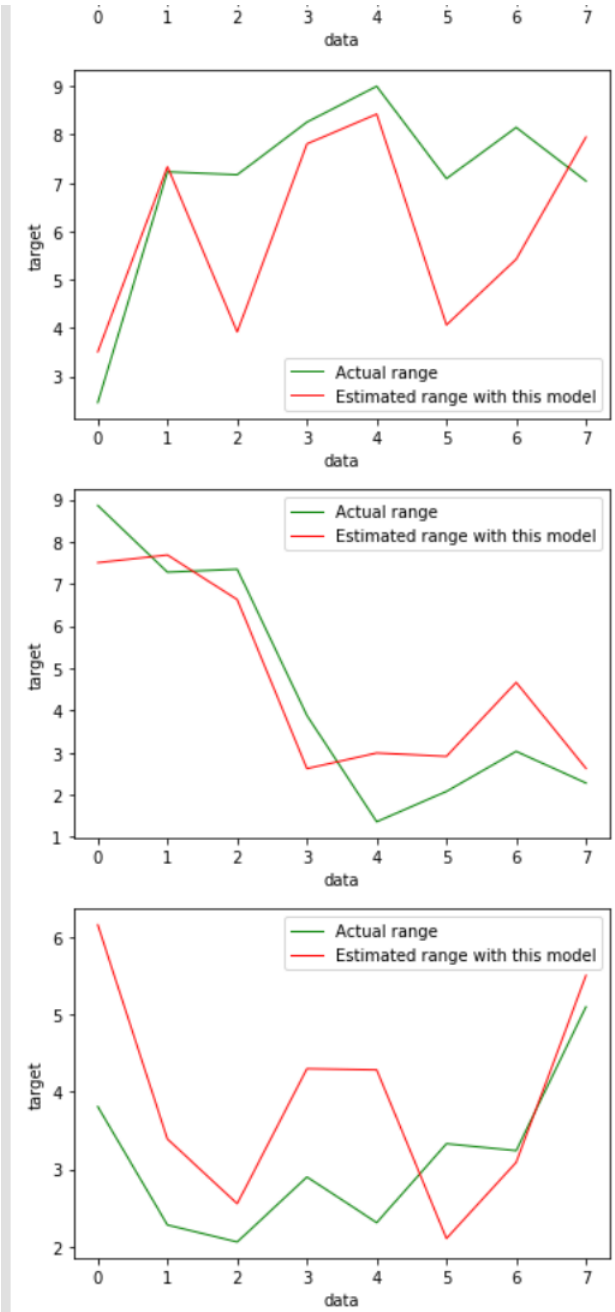
Figure 56: regression result(Valance) for data1




```
MAE of AdaBoost is = 1.095750
Best hyper-params for Adaboost due to CV:
{'n_estimators': 70}
```

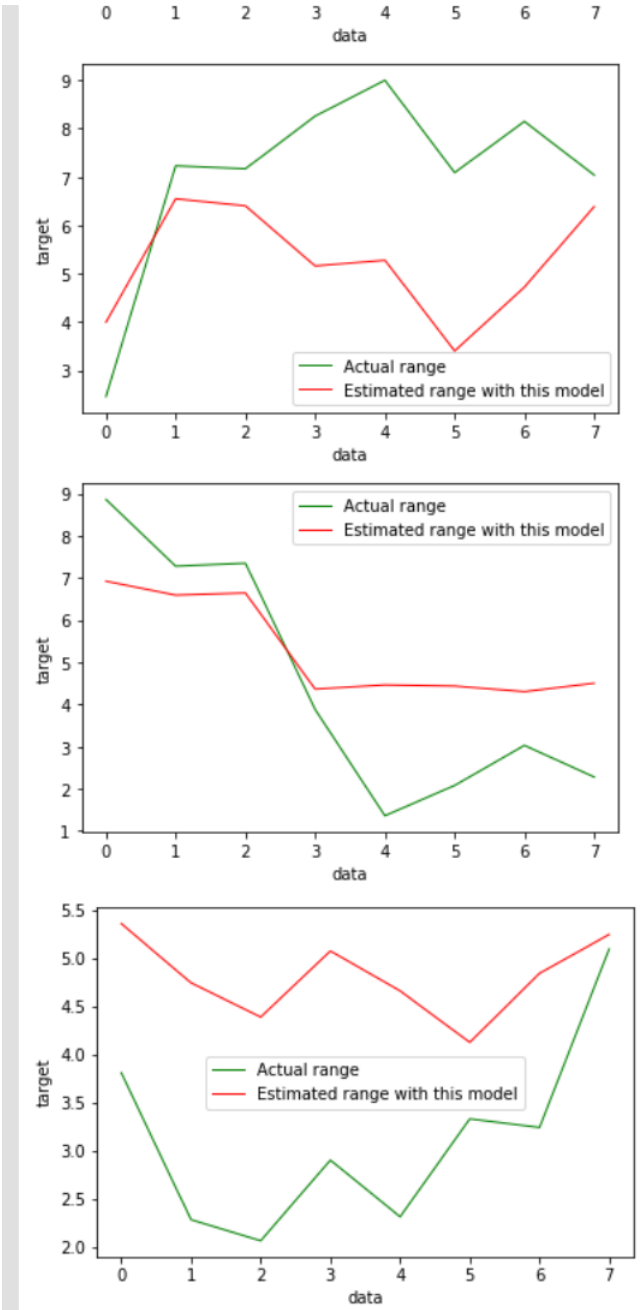
project/data1regv/e.png

Figure 57: regression result(Valance) for data1



project/data1regv/f.png

Figure 58: regression result(Valance) for data1

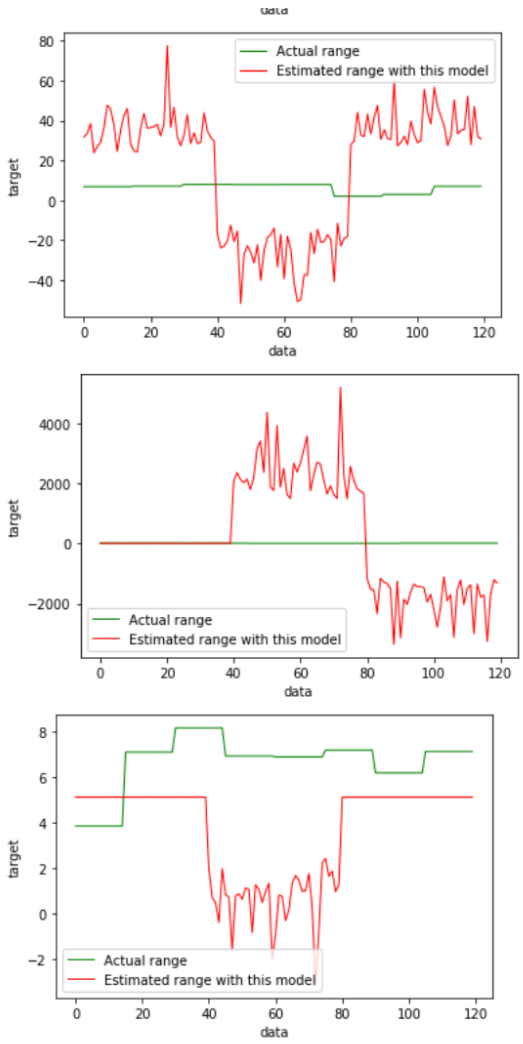



```
MAE of RandomForest is = 1.883518
Best hyper-params for RandomForest due to CV:
{'max_depth': 10, 'n_estimators': 40}
```

project/data1regv/g.png

Figure 59: regression result(Valance) for data1

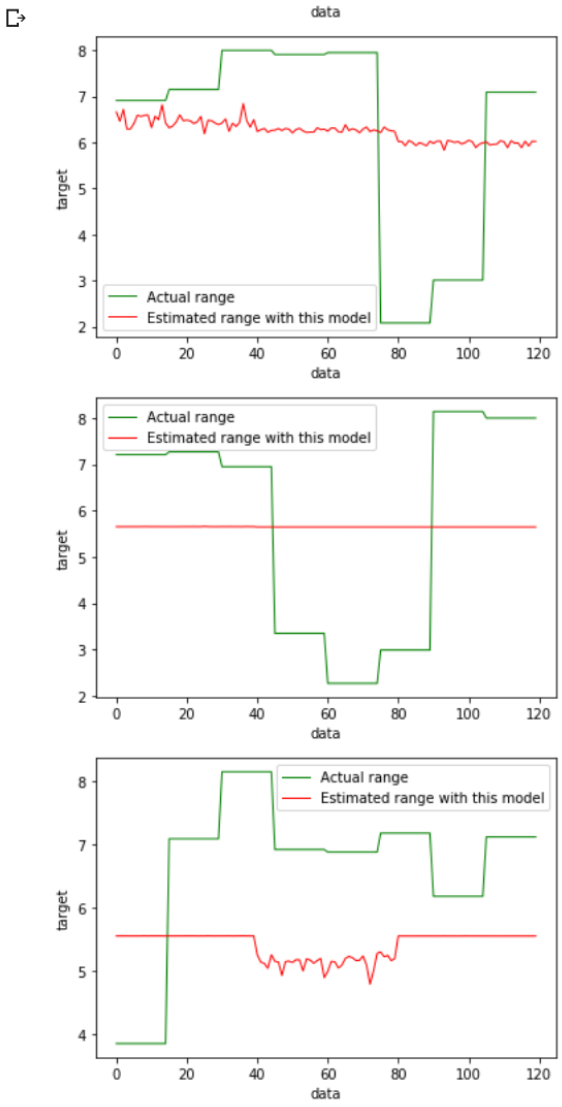
Regression for data2:
Arousal:



project/data2rega/a.png

```
MAE of LinReg is = 295.048801
```

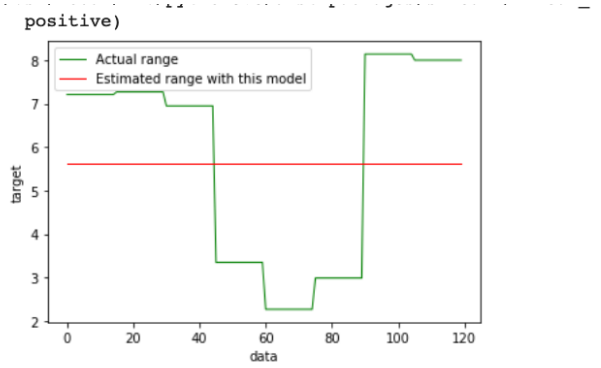
Figure 60: regression result(Arousal) for data2



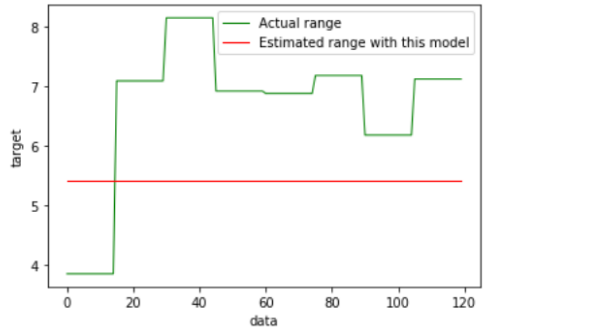
project/data2rega/b.png

```
MAE of Ridge is = 2.013063
Best hyper-params for Ridge due to CV:
```

Figure 61: regression result(Arousal) for data2



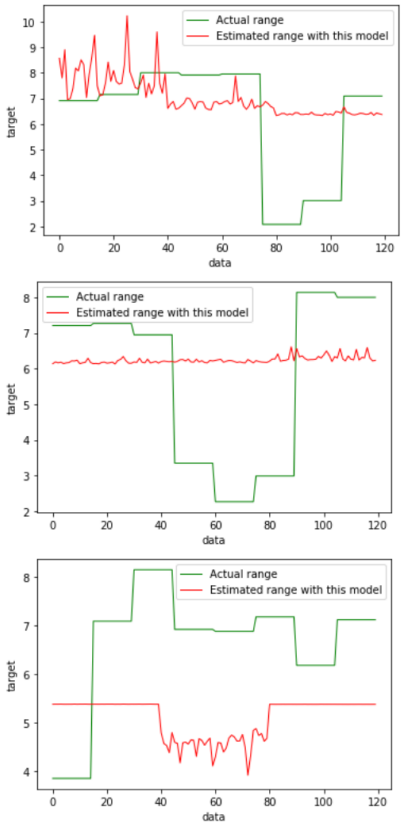

```
positive)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_m
  positive)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_m
  positive)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_m
  positive)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_m
  positive)
```




```
MAE of Lasso is = 2.019728
Best hyper-params for Lasso due to CV:
{'alpha': 1}
```

project/data2rega/c.png

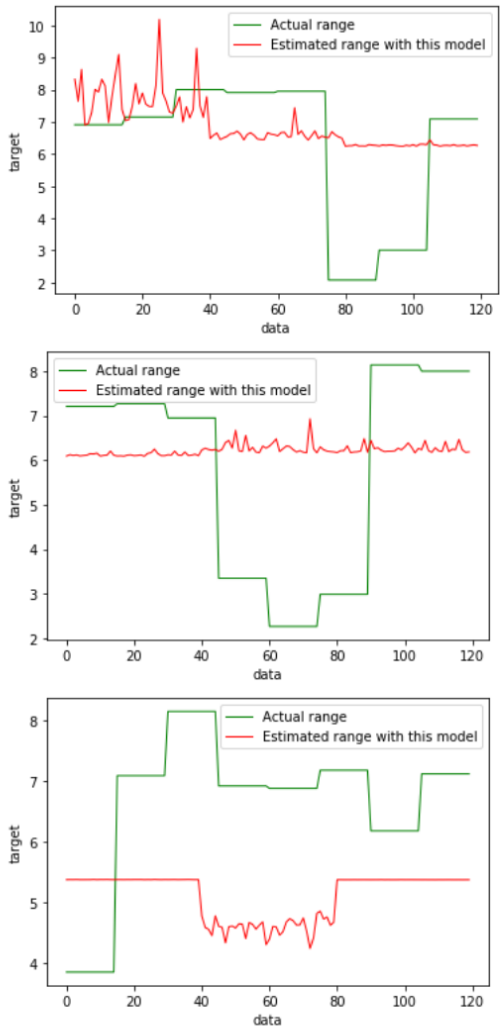
Figure 62: regression result(Arousal) for data2



project/data2rega/d.png `MAE of polynomial reg2 is = 2.004249`

Figure 63: regression result(Arousal) for data2

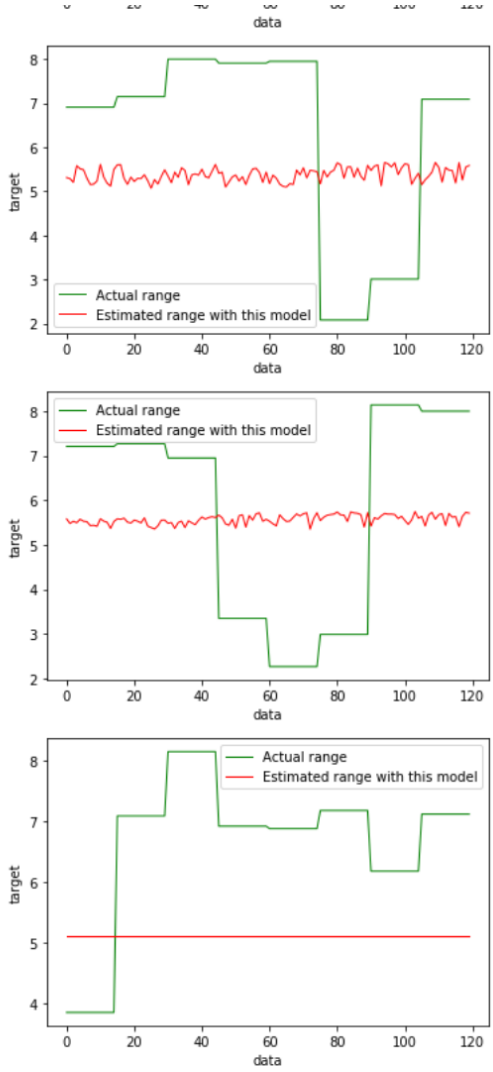
17]



project/data2rega/e.png

```
MAE of polynomial reg3 is = 2.014105
```

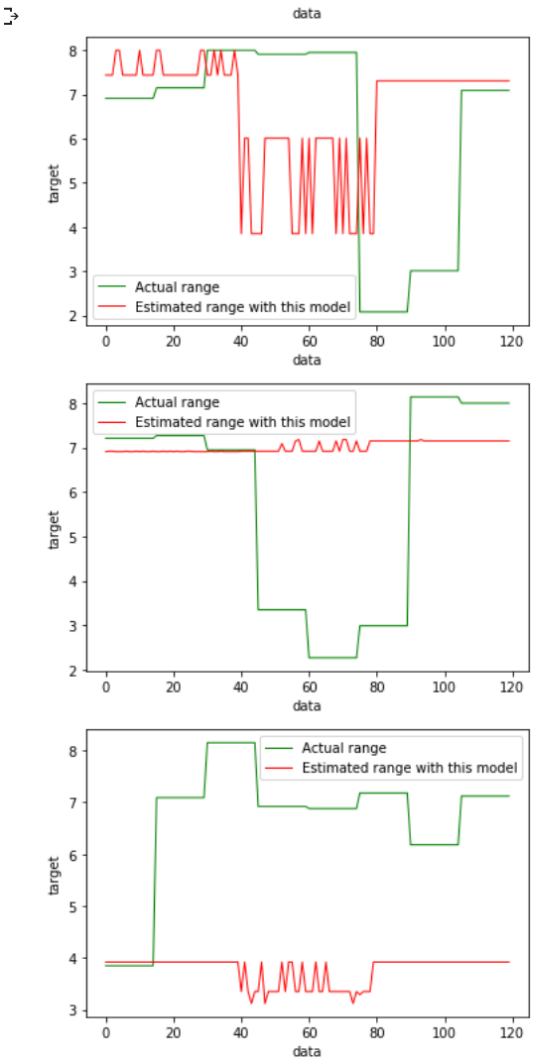
Figure 64: regression result(Arousal) for data2




```
MAE of SVR is = 2.198706
Best hyper-params for SVR due to CV:
{'C': 0.0001, 'epsilon': 5, 'kernel': 'rbf'}
```

project/data2rega/f.png

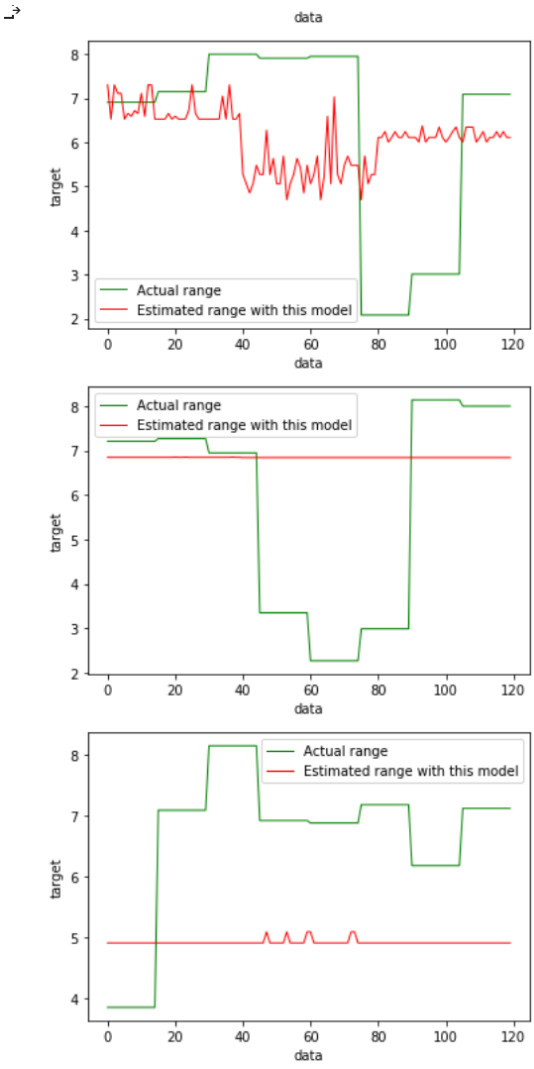
Figure 65: regression result(Arousal) for data2




```
MAE of AdaBoost is = 2.648850
Best hyper-params for Adaboost due to CV:
{'n_estimators': 100}
```

project/data2rega/g.png

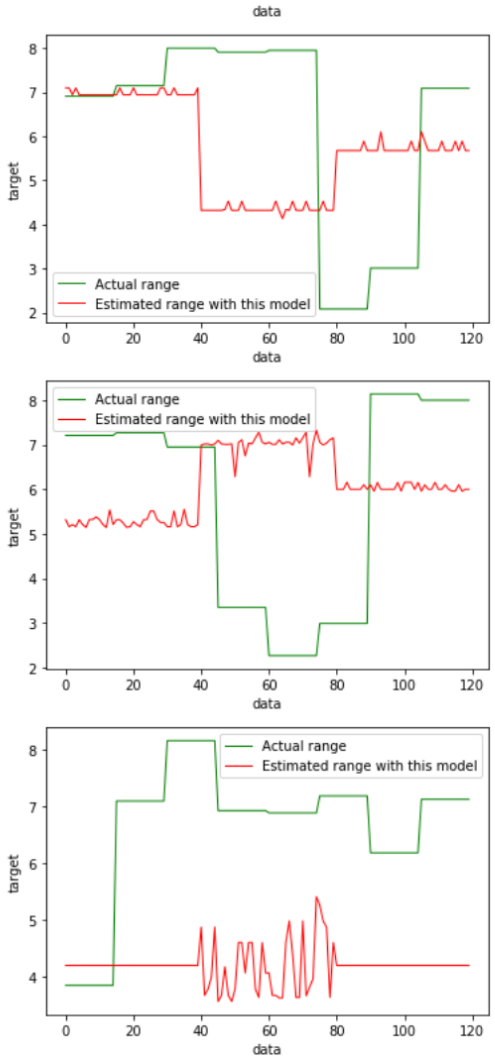
Figure 66: regression result(Arousal) for data2




```
MAE of KNN is = 1.969928
Best hyper-params for KNN due to CV:
{'n_neighbors': 20}
```

project/data2rega/h.png

Figure 67: regression result(Arousal) for data2

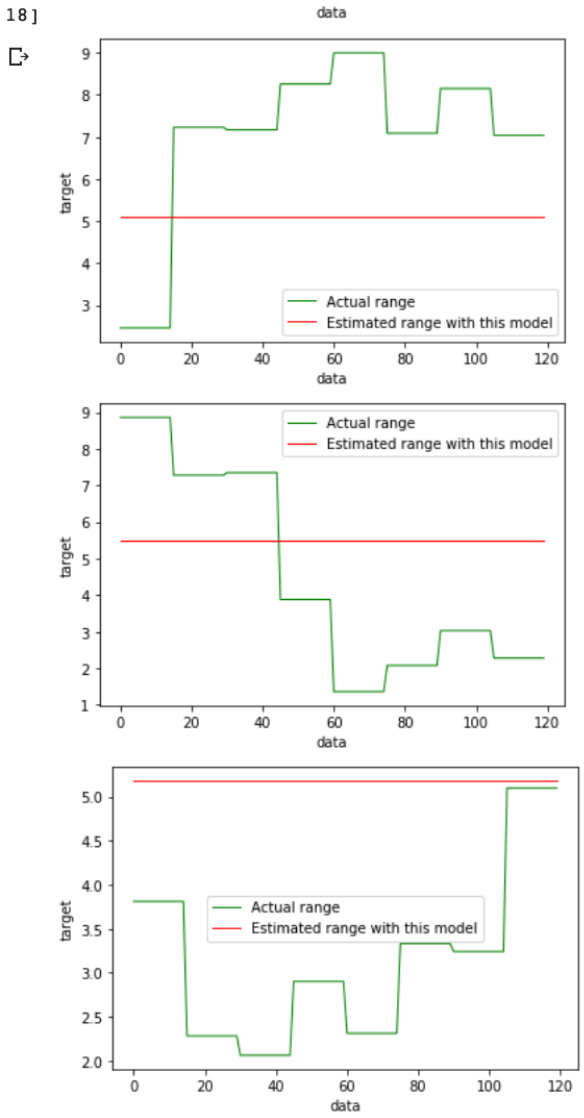



```
MAE of RandomForest is = 2.471902
Best hyper-params for RandomForest due to CV:
{'max_depth': 12, 'n_estimators': 10}
```

project/data2rega/i.png

Figure 68: regression result(Arousal) for data2

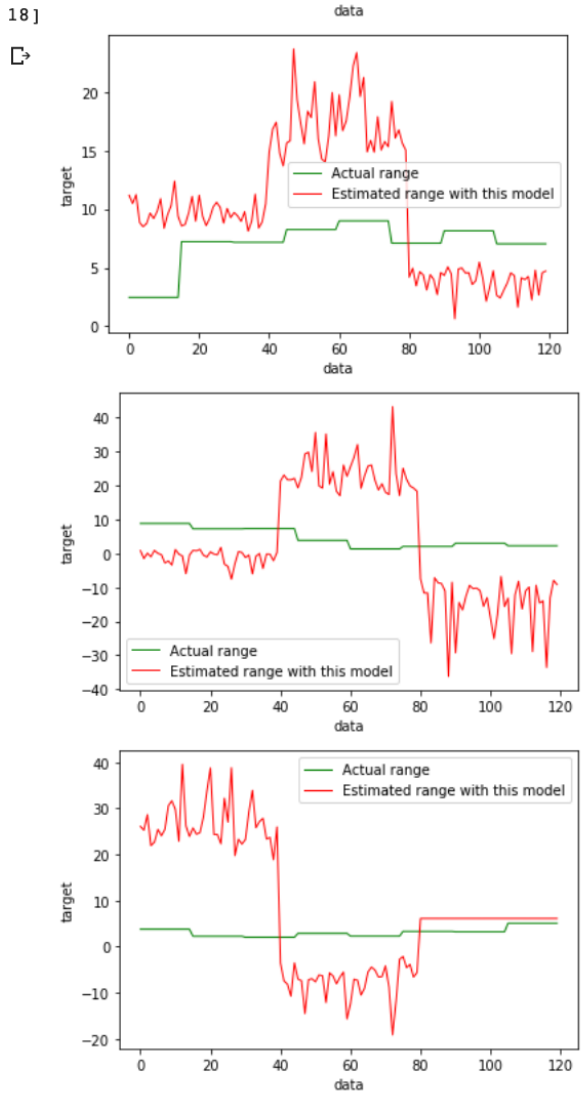
Valance:




```
MAE of SVR is = 3.019175
Best hyper-params for SVR due to CV:
{'C': 0.0001, 'epsilon': 5, 'kernel': 'rbf'}
```

project/data2regv/a.png

Figure 69: regression result(Valance) for data2

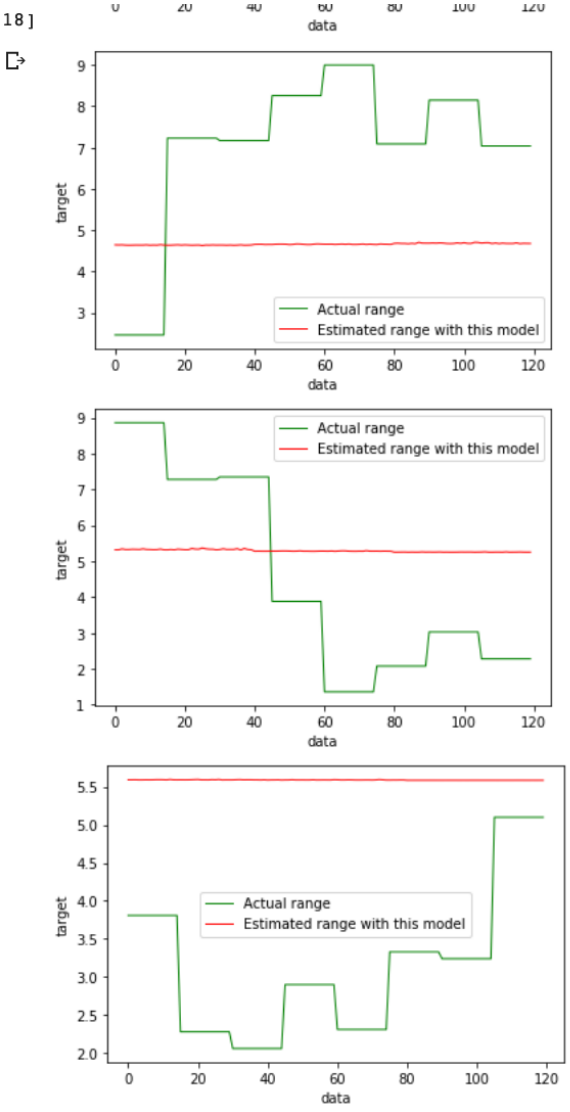


project/data2regv/b.png

```
MAE of LinReg is = 13.298979
```

Figure 70: regression result(Valance) for data2

18]

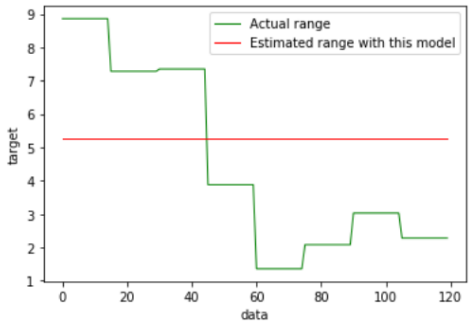



```
MAE of Ridge is = 2.785953
Best hyper-params for Ridge due to CV:
{'alpha': 1000}
```

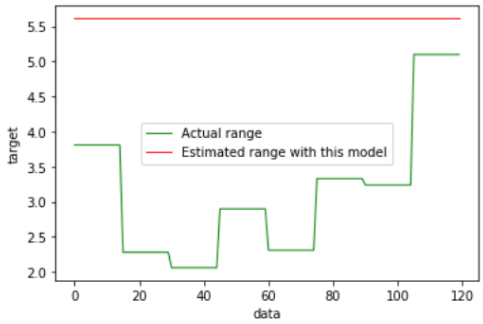
project/data2regv/c.png

Figure 71: regression result(Valance) for data2

```
positive)
```



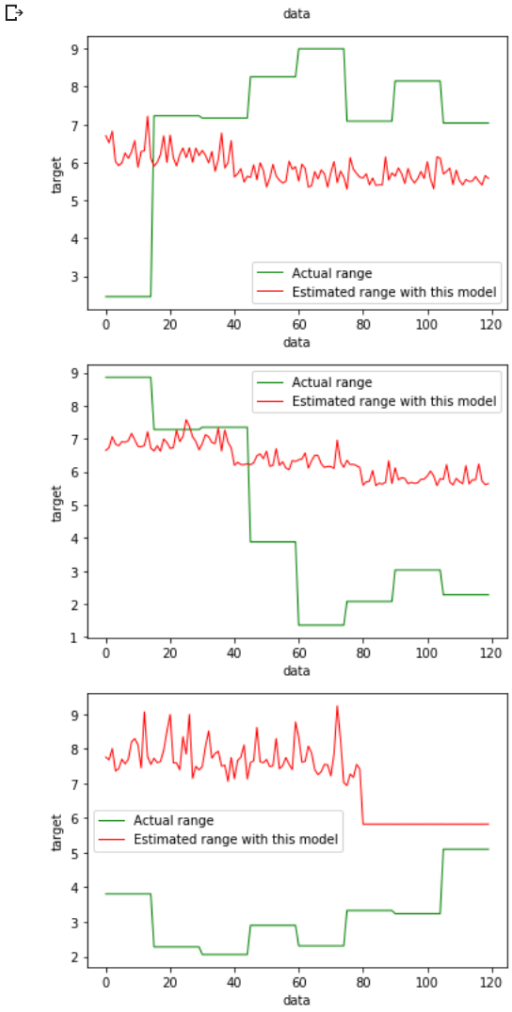

```
/usr/local/lib/python3.6/dist-packages/sklearn/linear_mo
  positive)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_mo
  positive)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_mo
  positive)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_mo
  positive)
```




```
MAE of Lasso is = 2.942555
Best hyper-params for Lasso due to CV:
{'alpha': 1}
```

project/data2regv/d.png

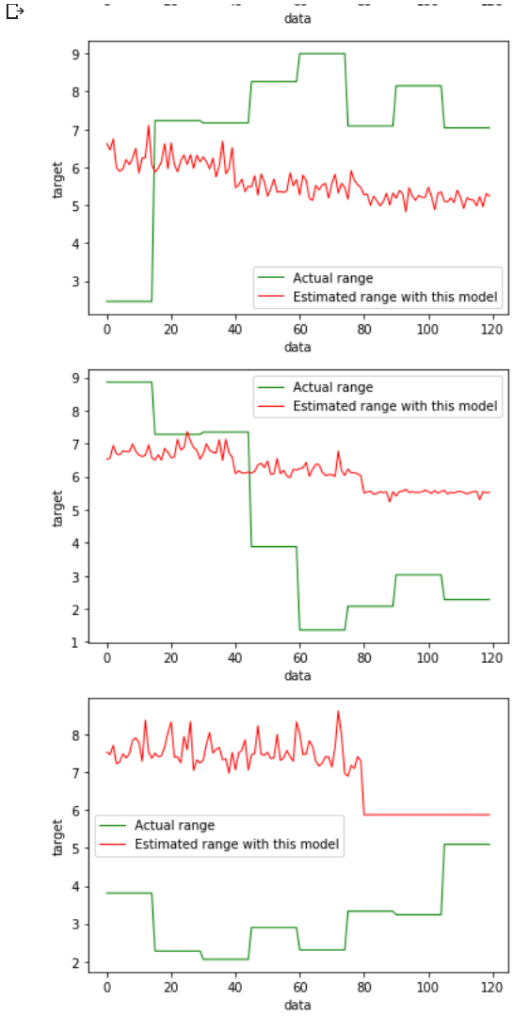
Figure 72: regression result(Valance) for data2



project/data2regv/e.png

```
MAE of polynomial reg2 is = 3.048826
```

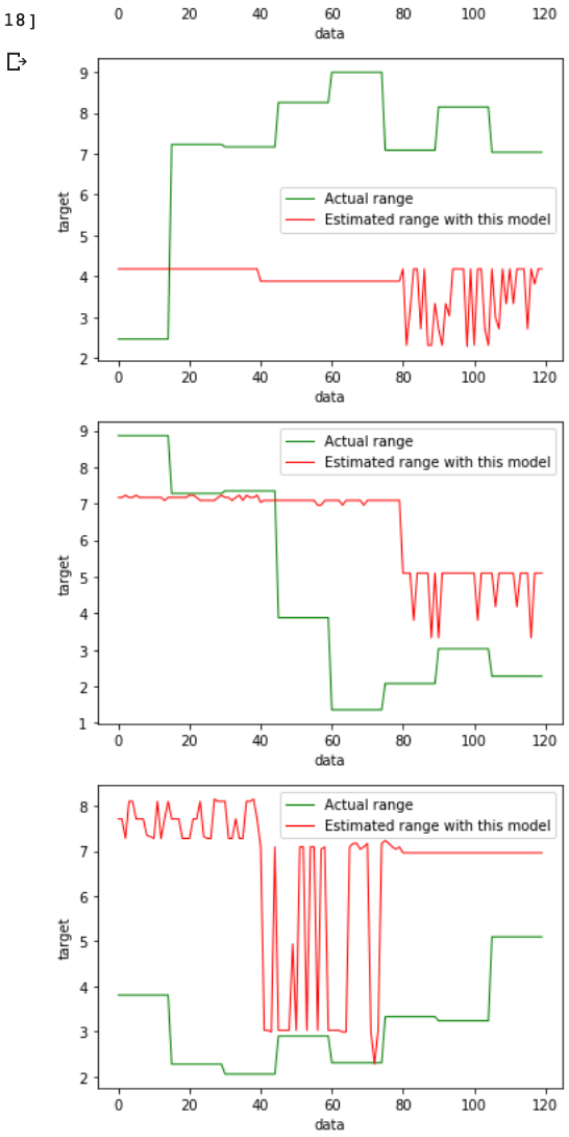
Figure 73: regression result(Valance) for data2



project/data2regv/f.png

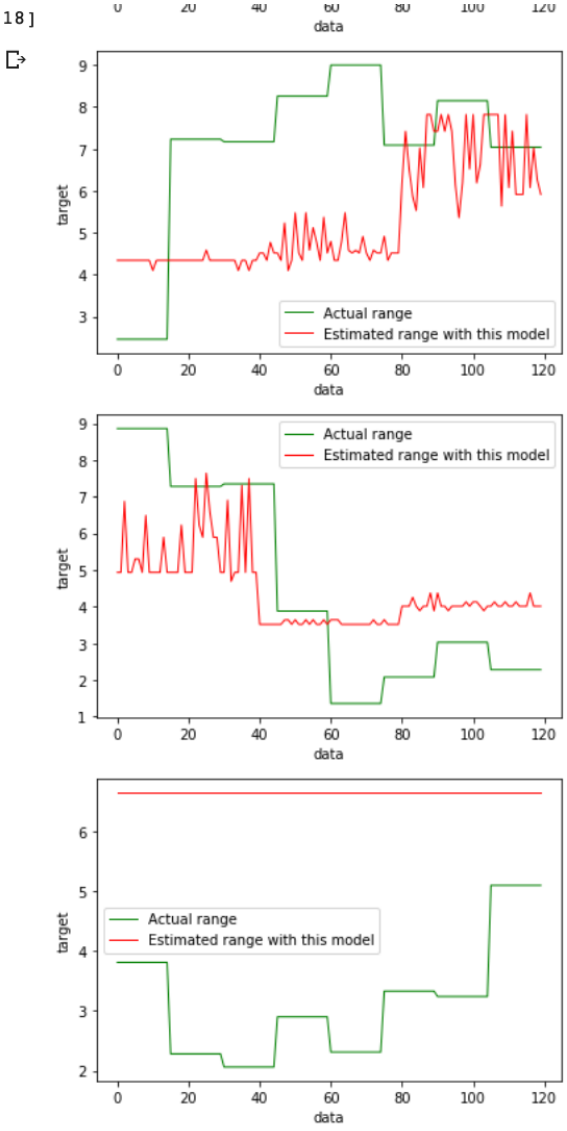
```
MAE of polynomial reg3 is = 3.083717
```

Figure 74: regression result(Valance) for data2



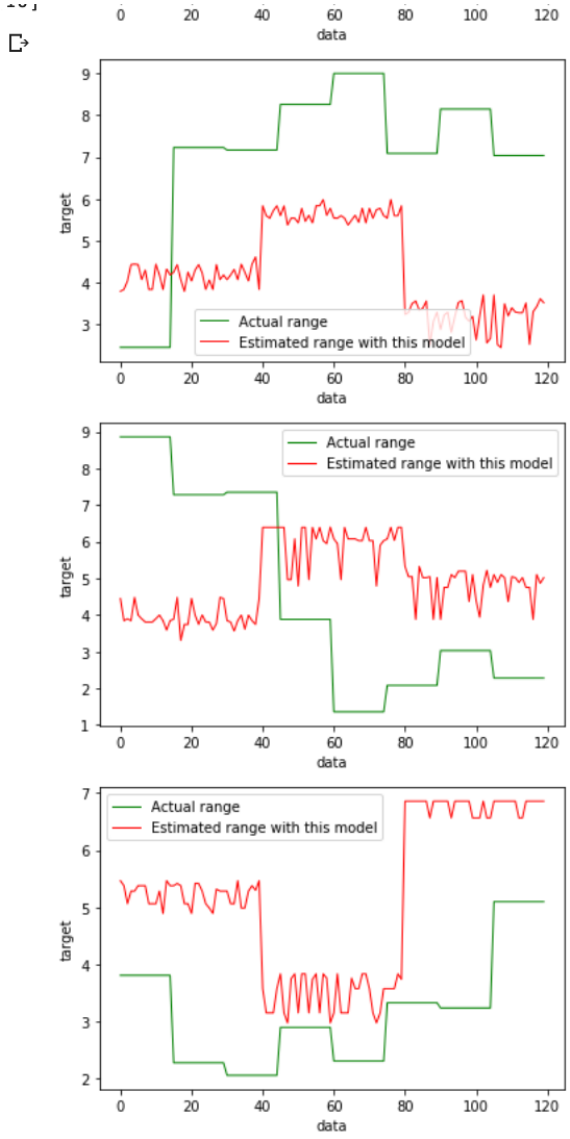
project/data2regv/g.png

Figure 75: regression result(Valance) for data2



project/data2regv/h.png

Figure 76: regression result(Valance) for data2

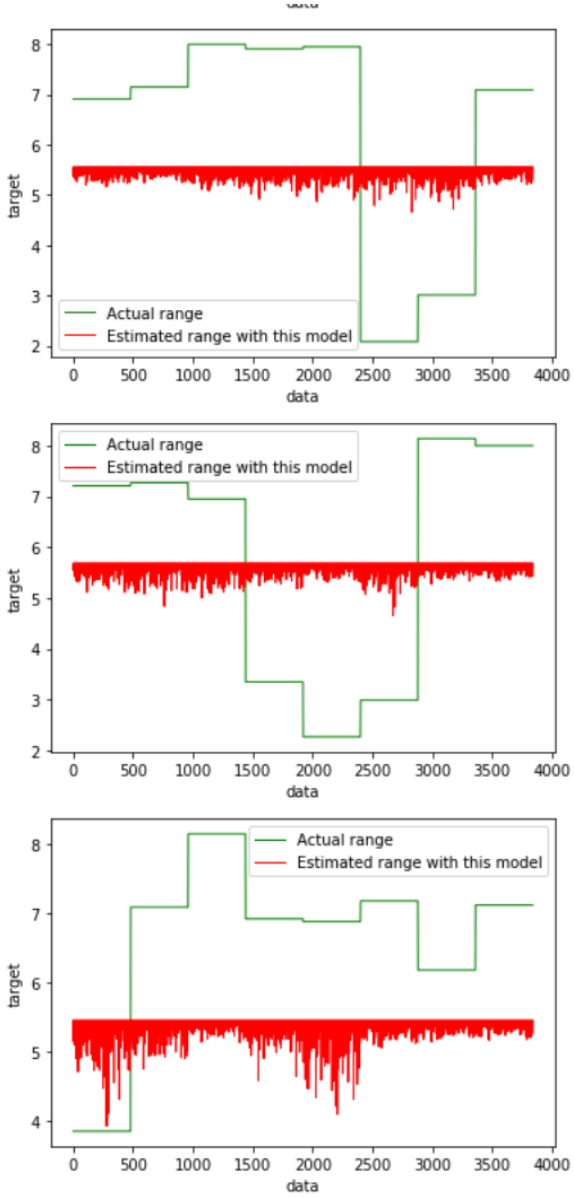



```
MAE of RandomForest is = 3.062123
Best hyper-params for RandomForest due to CV:
{'max_depth': 20, 'n_estimators': 10}
```

project/data2regv/i.png

Figure 77: regression result(Valance) for data2

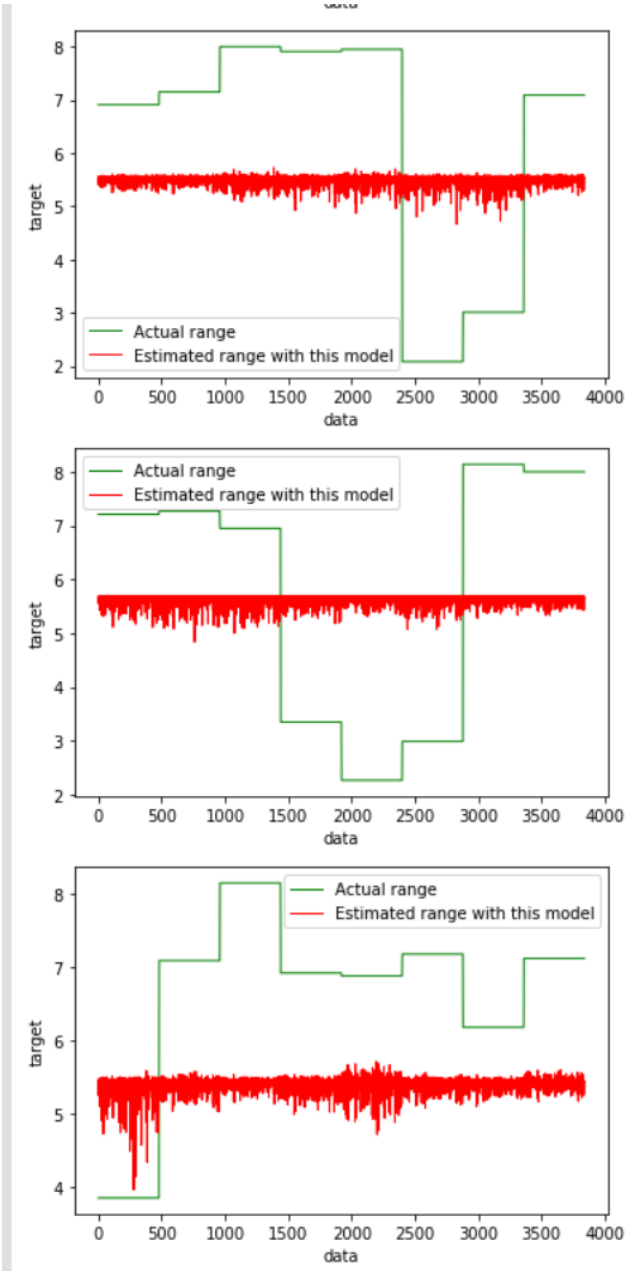
Regression for data3:
Arousal:



project/data3rega/a.png

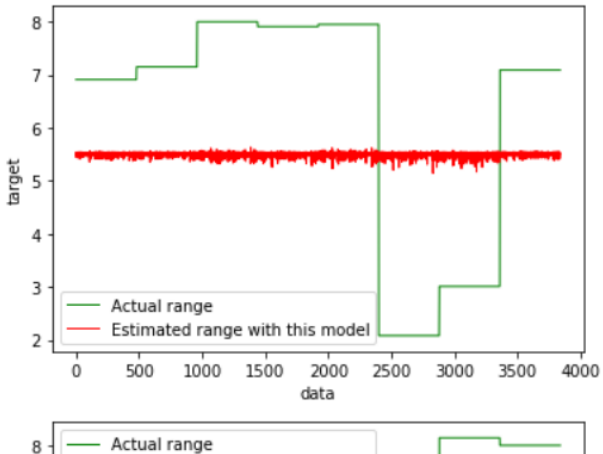
```
MAE of LinReg is = 2.097927
```

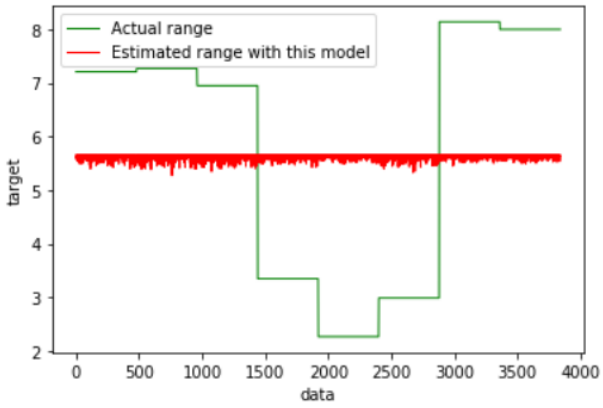
Figure 78: regression result(Arousal) for data3

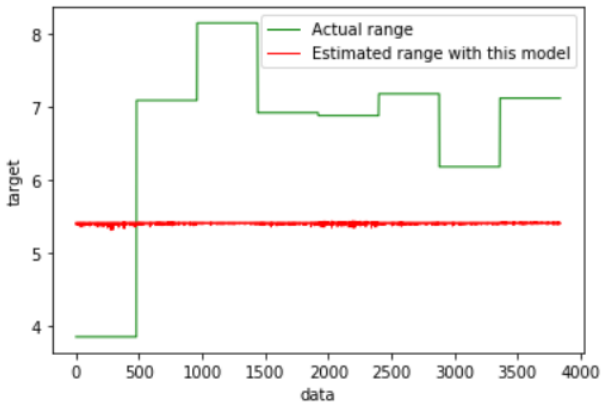


project/data3rega/b.png `MAE of LinReg is = 2.099364`

Figure 79: regression result(Arousal) for data3



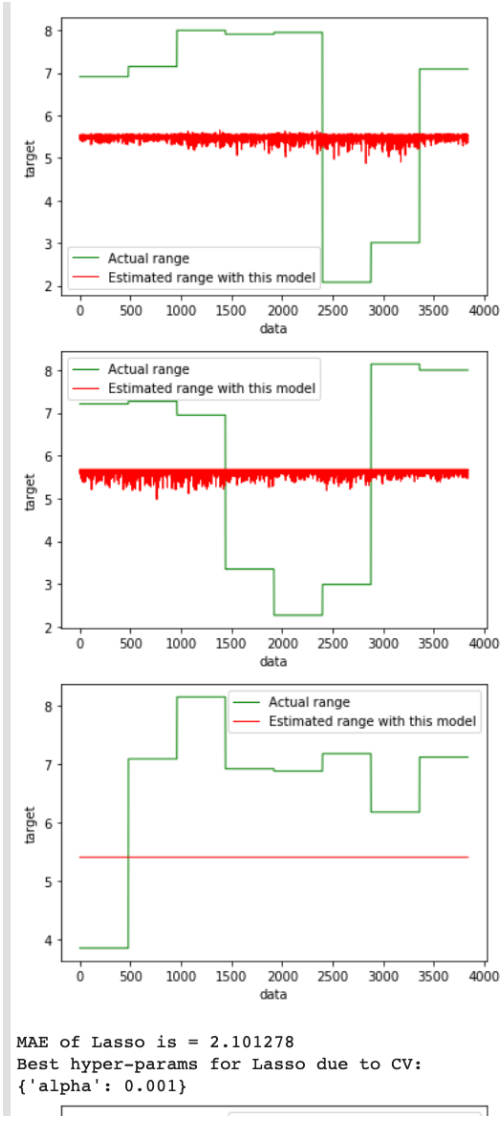





```
MAE of Ridge is = 2.101195
Best hyper-params for Ridge due to CV:
{'alpha': 10}
```

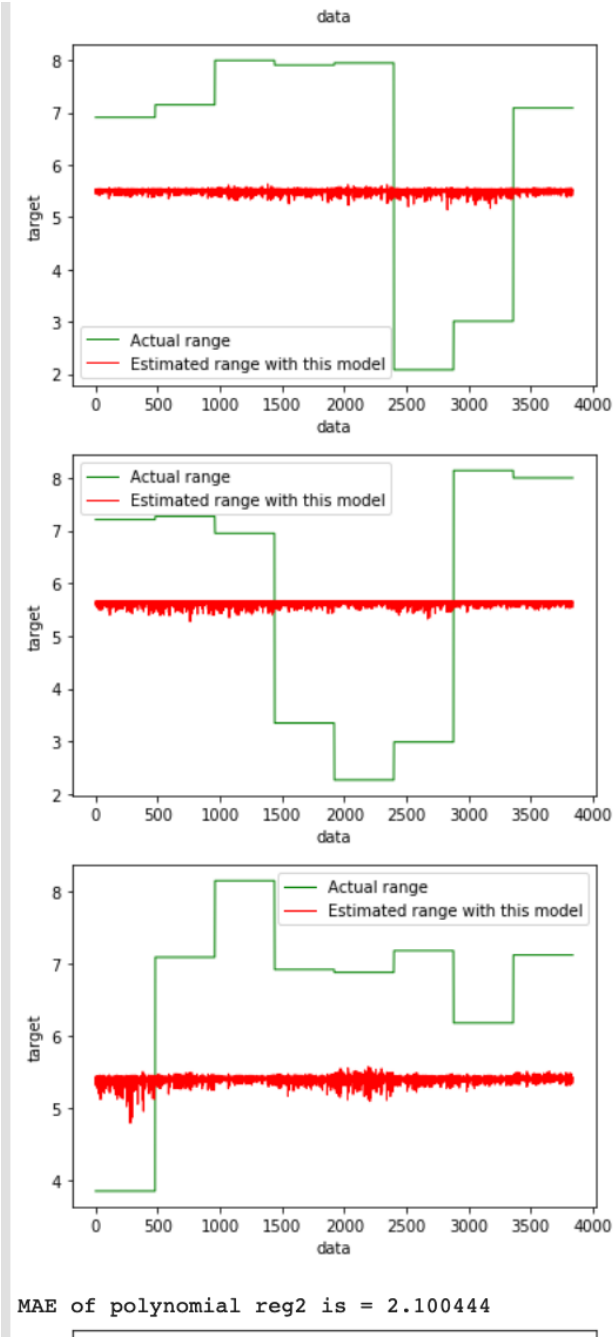
project/data3rega/c.png

Figure 80: regression result(Arousal) for data3



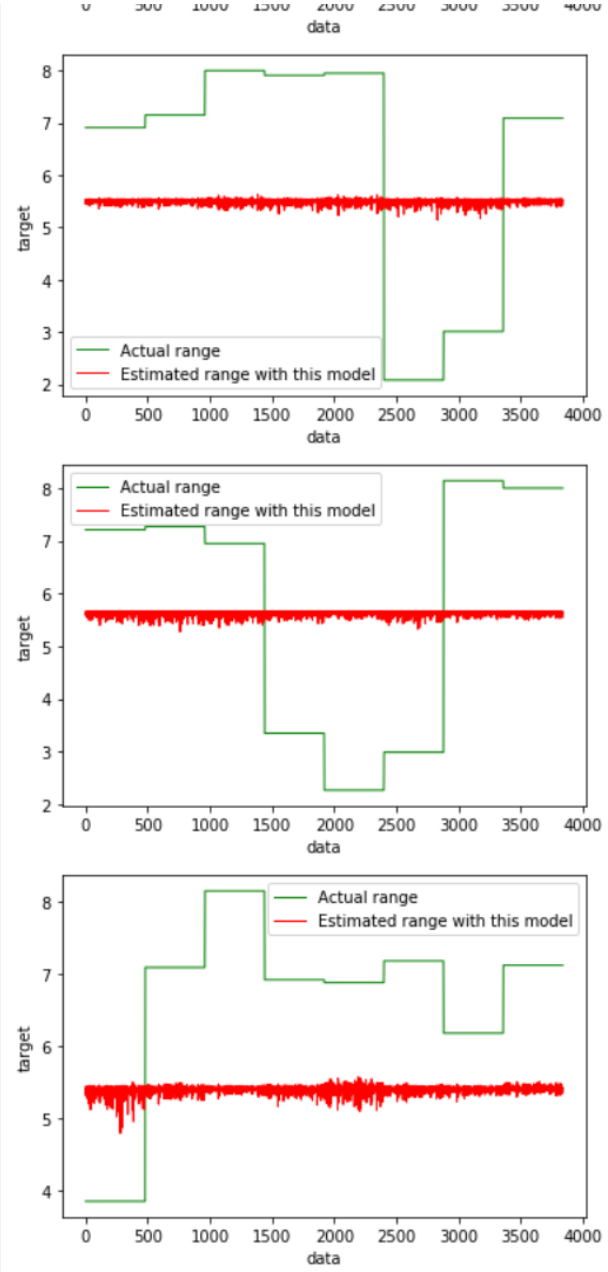
project/data3rega/d.png

Figure 81: regression result(Arousal) for data3



project/data3rega/e.png

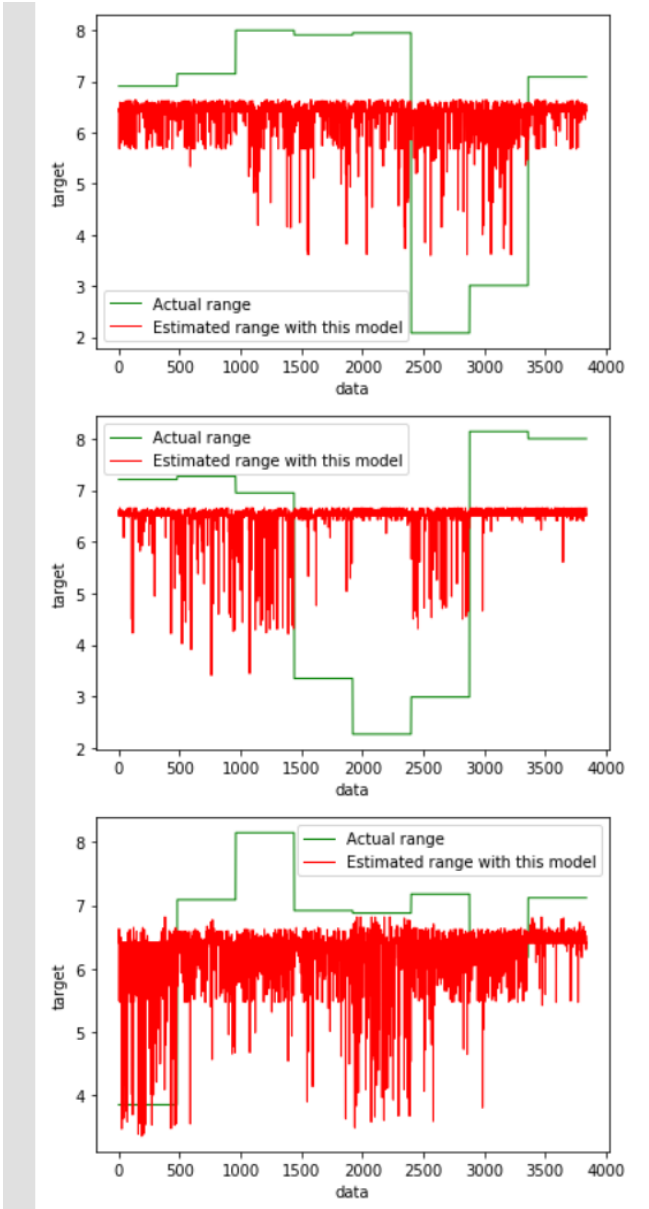
Figure 82: regression result(Arousal) for data3



project/data3rega/f.png

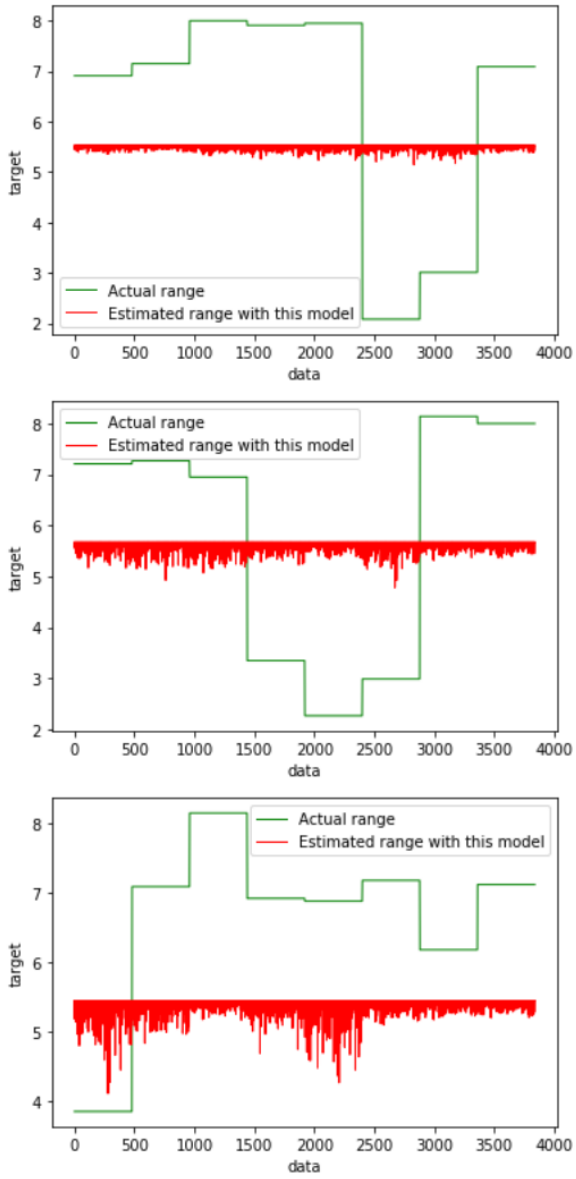
```
MAE of polynomial reg3 is = 2.100444
```

Figure 83: regression result(Arousal) for data3



project/data3rega/g.png

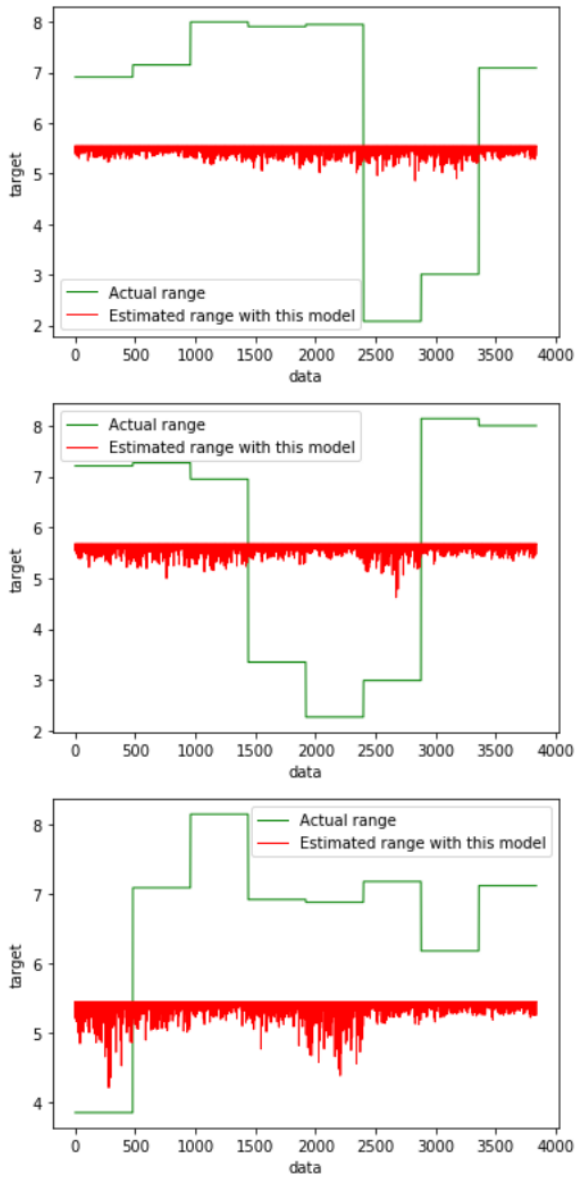
Figure 84: regression result(Arousal) for data3




```
MAE of Ridge is = 2.099481
Best hyper-params for Ridge due to CV:
{'alpha': 0.1}
```

project/data3rega/h.png

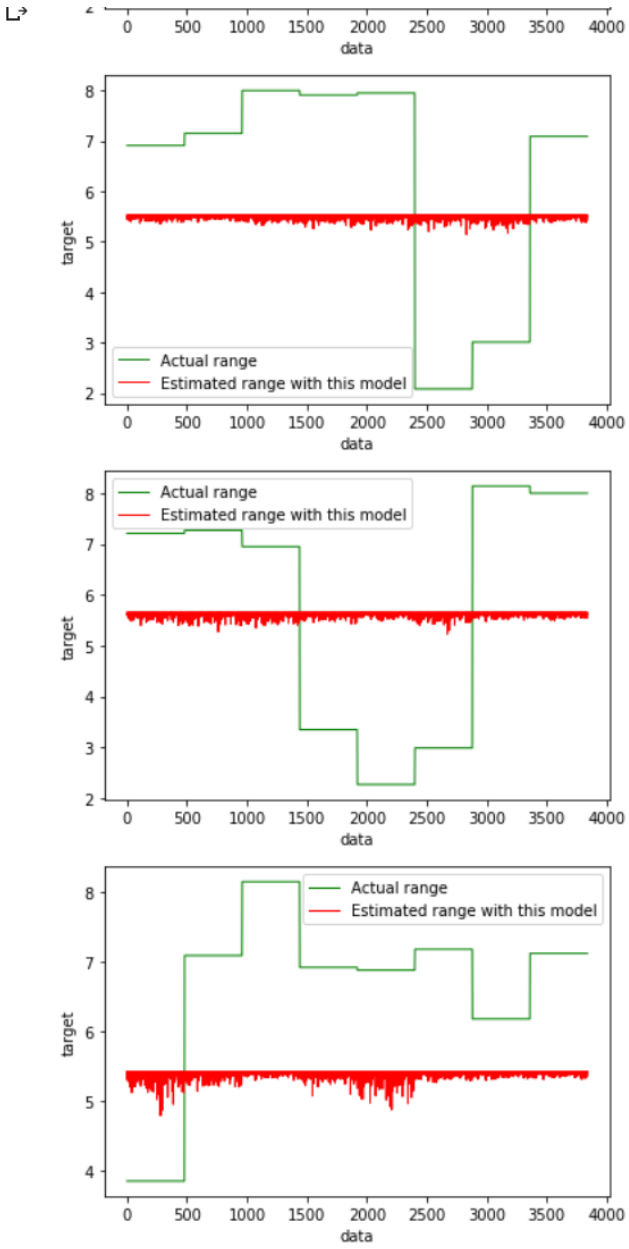
Figure 85: regression result(Arousal) for data3




```
MAE of Lasso is = 2.098493
Best hyper-params for Lasso due to CV:
{'alpha': 0.0001}
```

project/data3rega/i.png

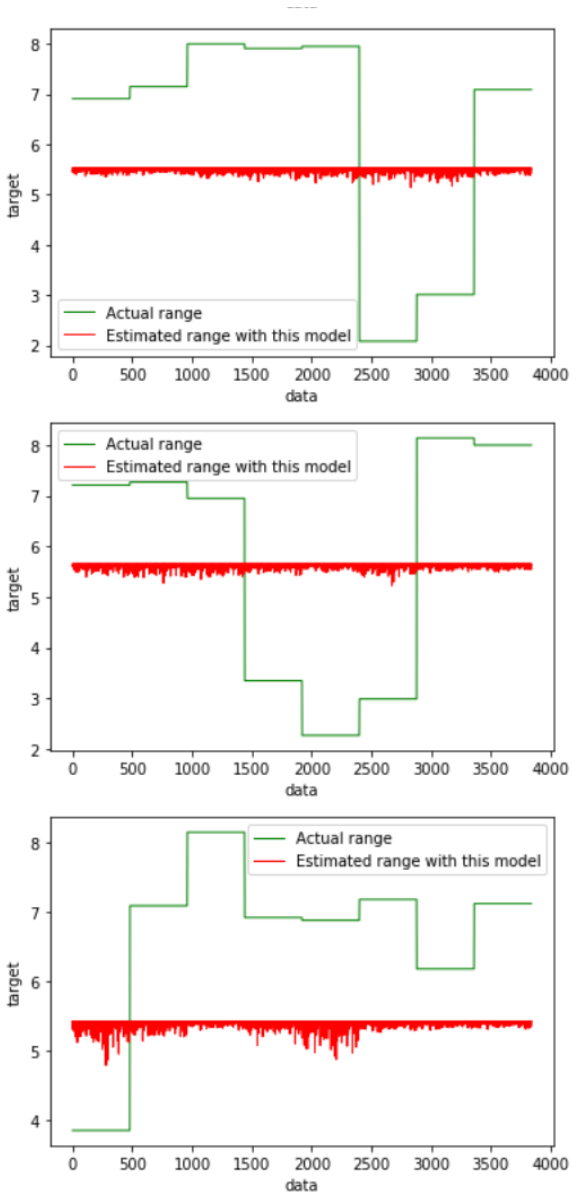
Figure 86: regression result(Arousal) for data3



project/data3rega/j.png

```
MAE of polynomial reg2 is = 2.100018
```

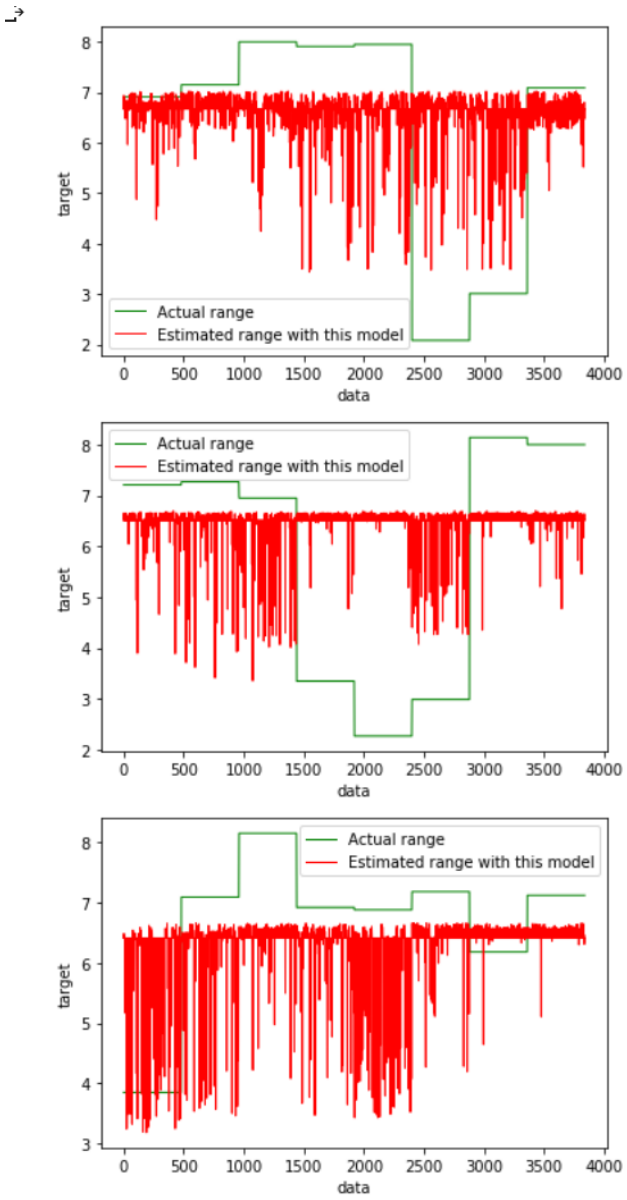
Figure 87: regression result(Arousal) for data3



project/data3rega/k.png

```
MAE of polynomial reg3 is = 2.100018
```

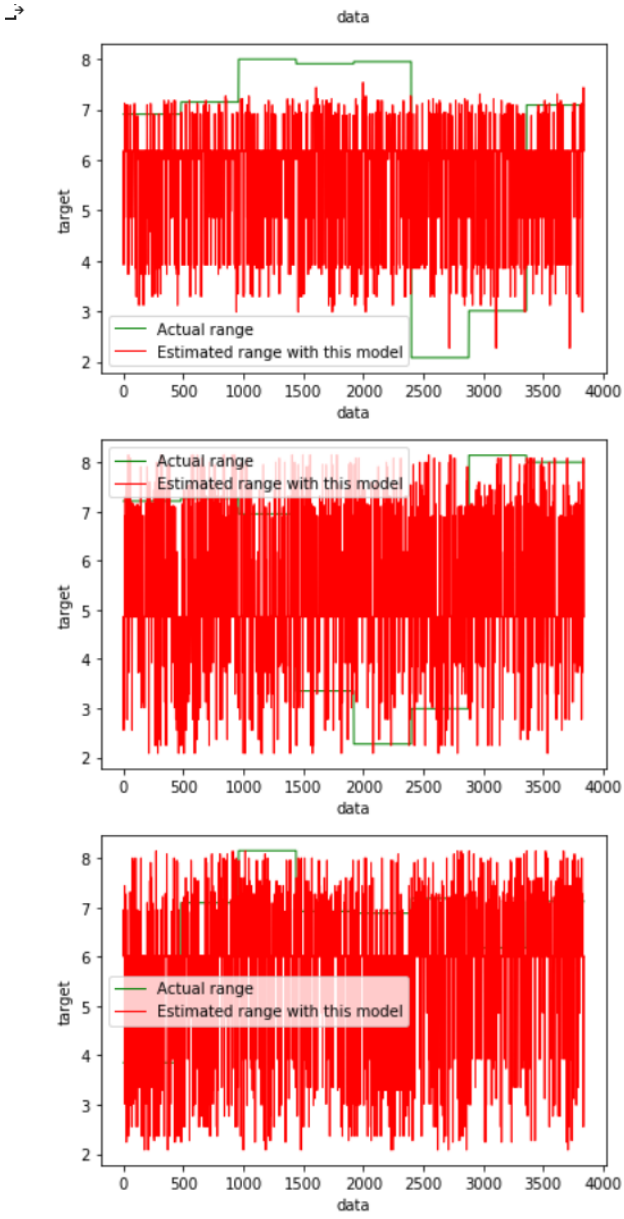
Figure 88: regression result(Arousal) for data3




```
MAE of SVR is = 1.908484
Best hyper-params for SVR due to CV:
{'C': 0.1, 'epsilon': 0.5, 'kernel': 'rbf'}
```

project/data3rega/l.png

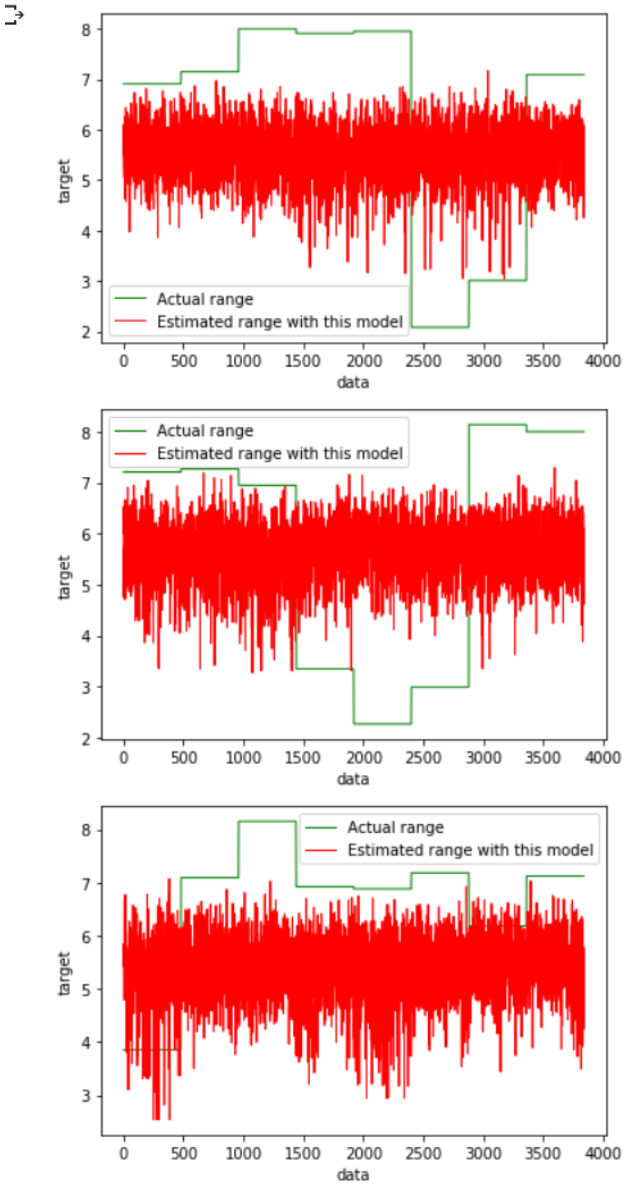
Figure 89: regression result(Arousal) for data3




```
MAE of AdaBoost is = 2.074185
Best hyper-params for Adaboost due to CV:
{'n_estimators': 10}
```

project/data3rega/m.png

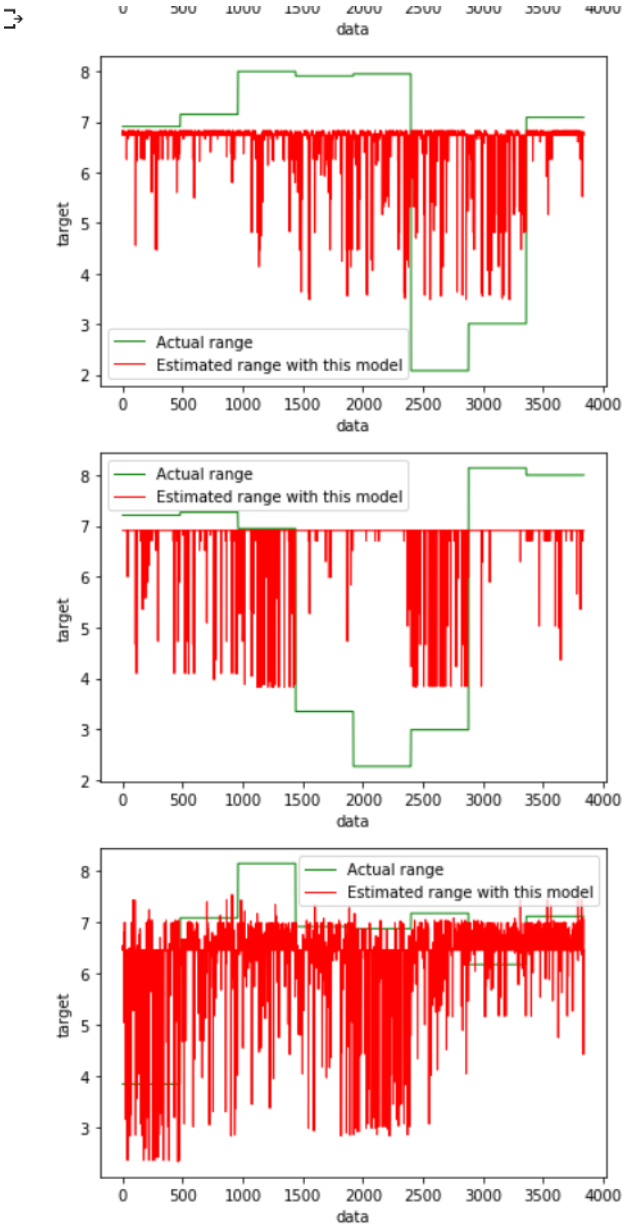
Figure 90: regression result(Arousal) for data3




```
MAE of KNN is = 2.115779
Best hyper-params for KNN due to CV:
{'n_neighbors': 20}
```

project/data3rega/n.png

Figure 91: regression result(Arousal) for data3

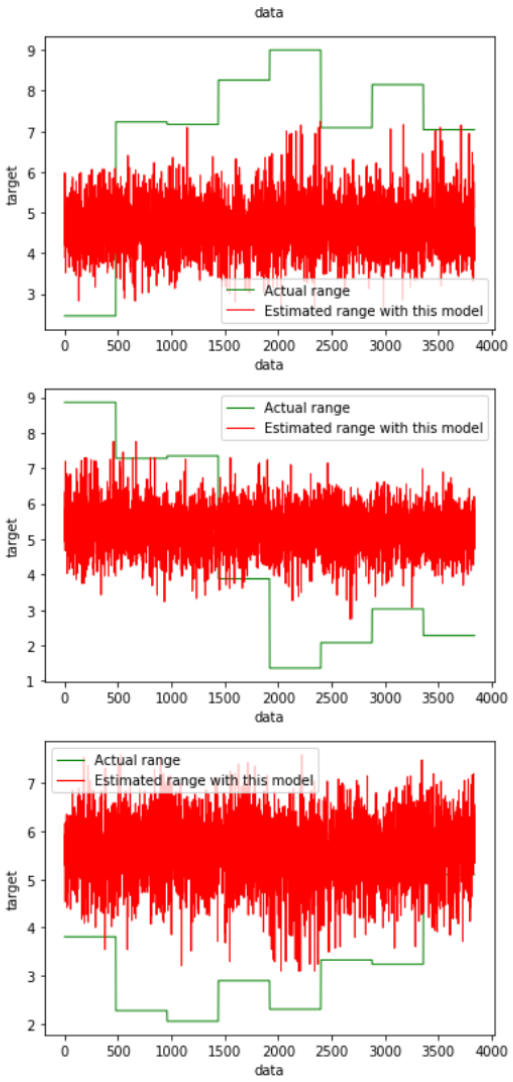



```
MAE of RandomForest is = 1.924109
Best hyper-params for RandomForest due to CV:
{'max_depth': 8, 'n_estimators': 10}
```

project/data3rega/o.png

Figure 92: regression result(Arousal) for data3

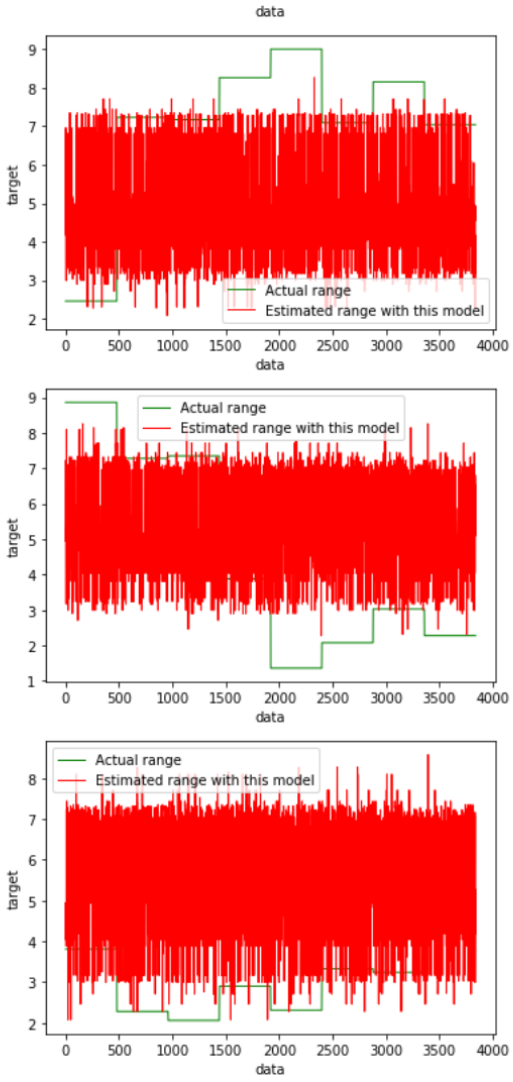
Valance:




```
MAE of KNN is = 2.668883
Best hyper-params for KNN due to CV:
{'n_neighbors': 20}
```

project/data3regv/a.png

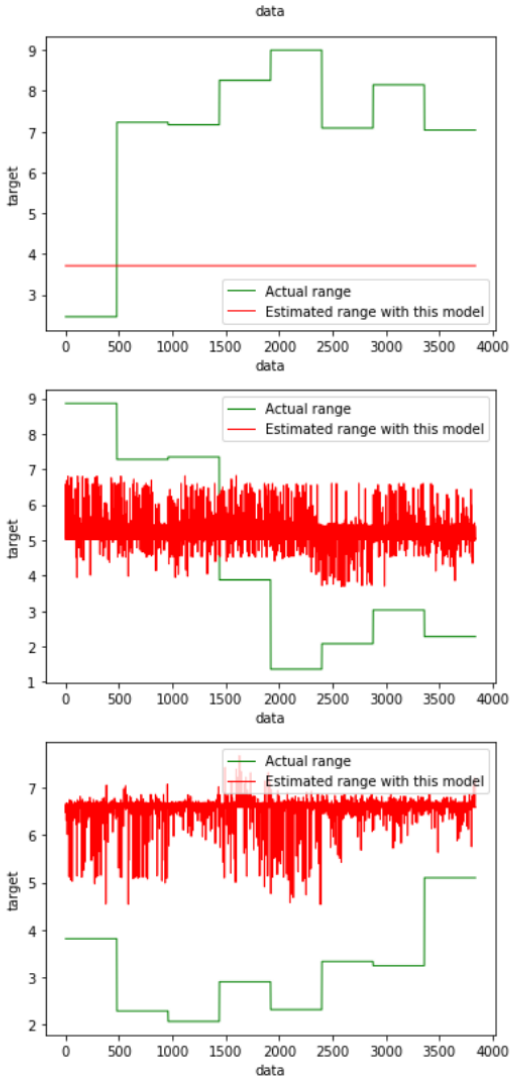
Figure 93: regression result(Valance) for data3




```
MAE of AdaBoost is = 2.647558
Best hyper-params for Adaboost due to CV:
{'n_estimators': 100}
```

project/data3regv/b.png

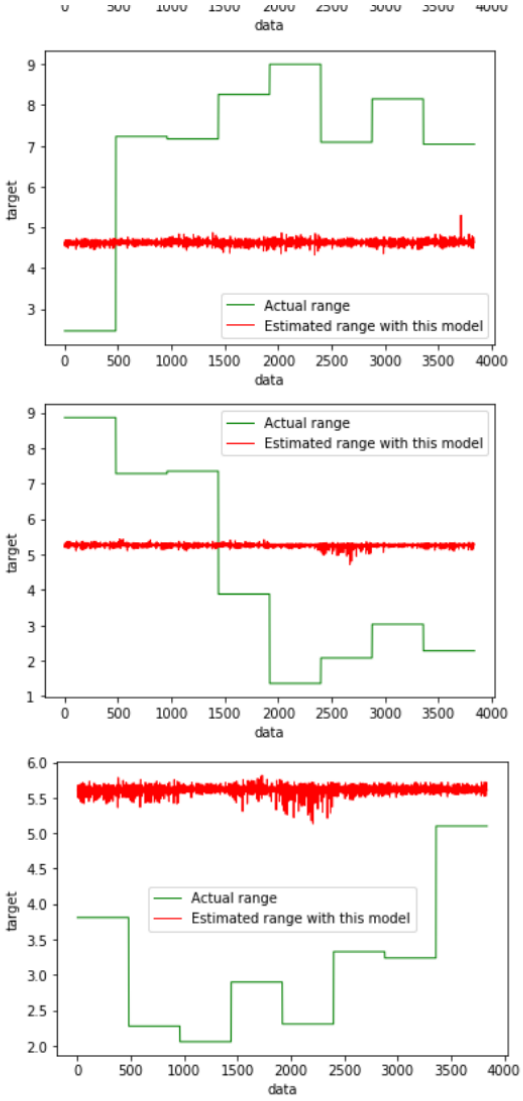
Figure 94: regression result(Valance) for data3




```
MAE of SVR is = 3.336379
Best hyper-params for SVR due to CV:
{'C': 0.1, 'epsilon': 0.5, 'kernel': 'rbf'}
```

project/data3regv/c.png

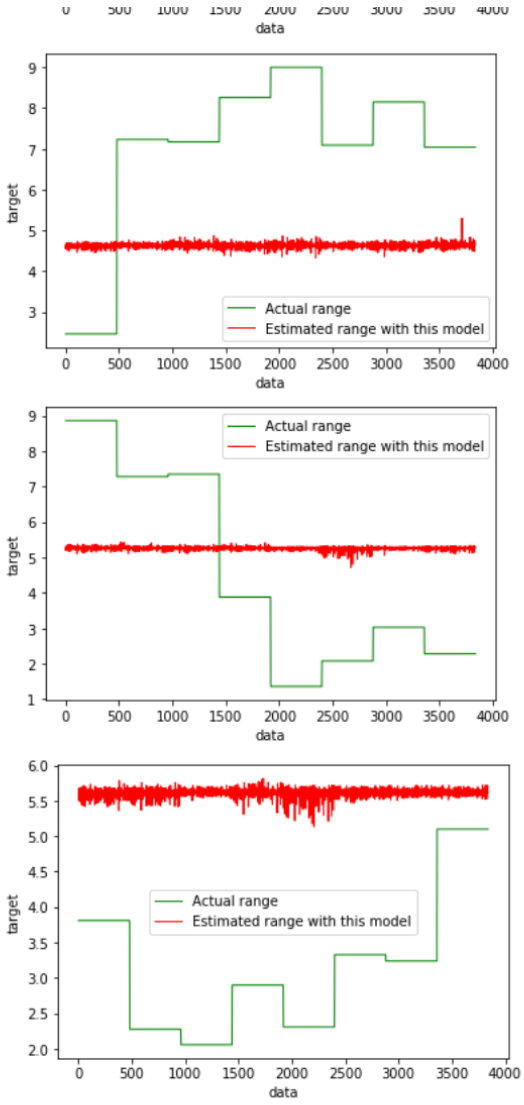
Figure 95: regression result(Valance) for data3



project/data3regv/d.png

```
MAE of polynomial reg3 is = 2.670917
```

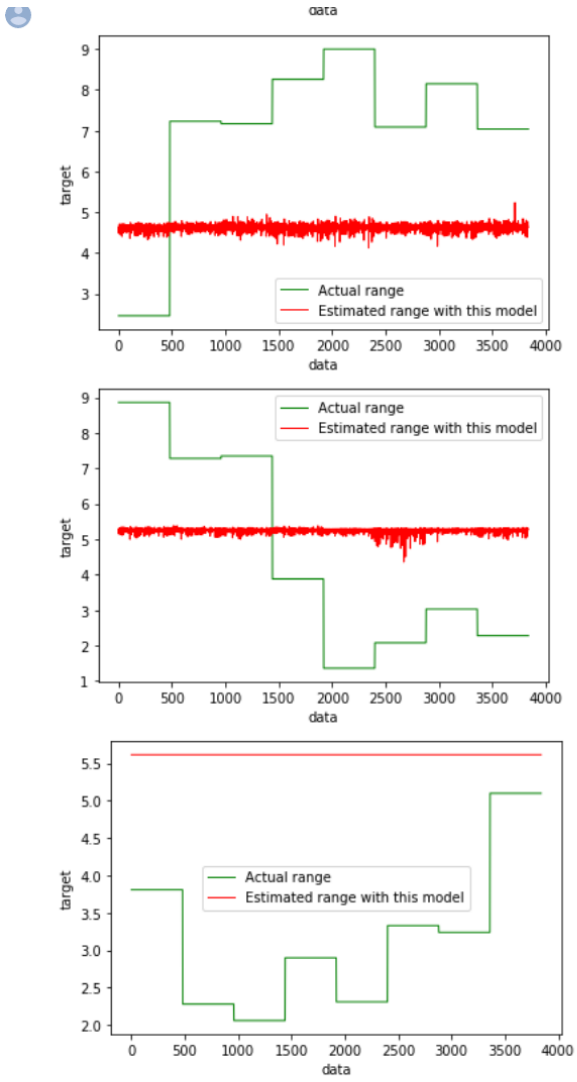
Figure 96: regression result(Valance) for data3



project/data3regv/e.png

```
MAE of polynomial reg2 is = 2.670917
```

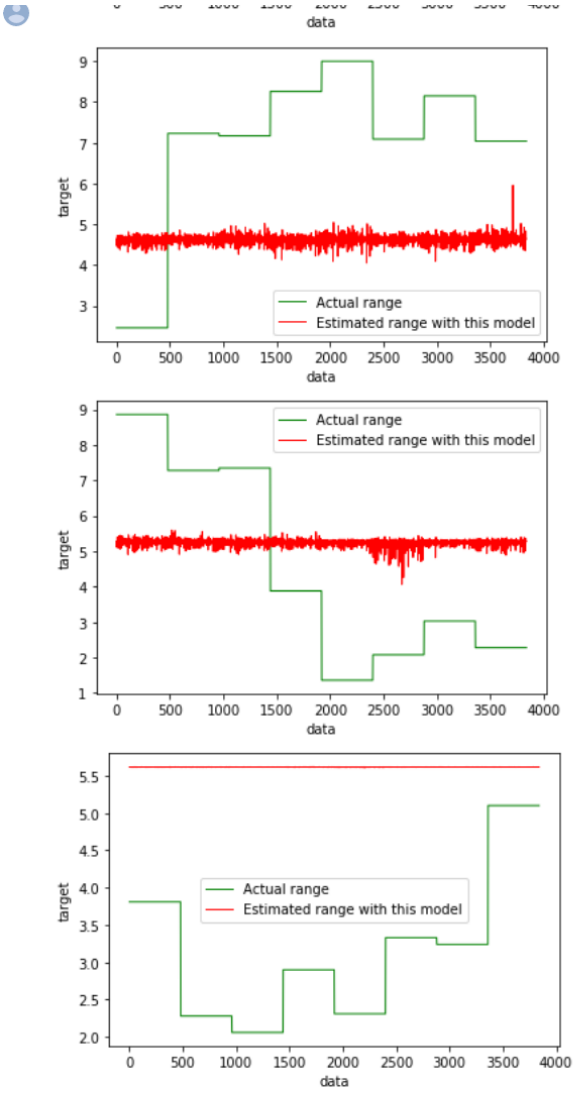
Figure 97: regression result(Valance) for data3



project/data3regv/f.png

```
MAE of Lasso is = 2.671053
Best hyper-params for Lasso due to CV:
{'alpha': 0.001}
```

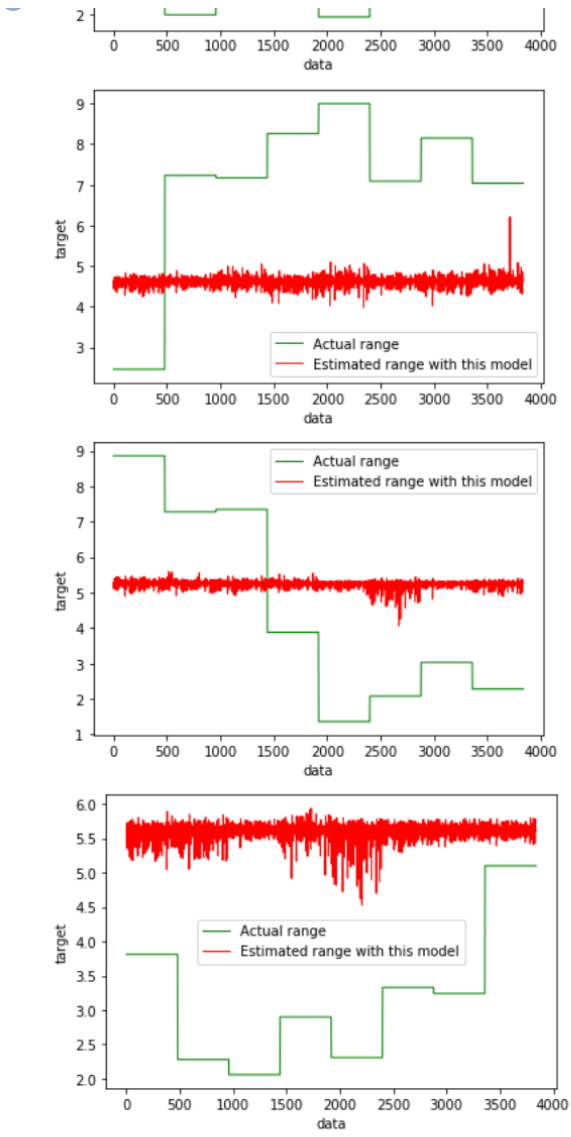
Figure 98: regression result(Valance) for data3




```
MAE of Ridge is = 2.670709
Best hyper-params for Ridge due to CV:
{'alpha': 100}
```

project/data3regv/g.png

Figure 99: regression result(Valance) for data3



project/data3regv/h.png

```
MAE of LinReg is = 2.669920
```

Figure 100: regression result(Valance) for data3