A2-3_svm

February 12, 2021

[20]:
```python
# python
import numpy as np
import matplotlib.pyplot as plt

# sklearn
from sklearn.svm import SVC
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error

# stats
import statsmodels.api as sm
```

[21]:
```python
## Load Test/Train Data
DATA = {
    "A": {
        "train": {
            "X": np.loadtxt(open("a2-files/X_train_A.csv"), delimiter=","),
            "Y": np.loadtxt(open("a2-files/Y_train_A.csv"), delimiter=","),
        }
    },
    "B": {
        "train": {
            "X": np.loadtxt(open("a2-files/X_train_B.csv"), delimiter=","),
            "Y": np.loadtxt(open("a2-files/Y_train_B.csv"), delimiter=","),
        },
        "test": {
            "X": np.loadtxt(open("a2-files/X_test_B.csv"), delimiter=","),
            "Y": np.loadtxt(open("a2-files/Y_test_B.csv"), delimiter=","),
        }
    },
}
```

[22]:
```python
# SVC-Soft
svc_regr_soft = SVC(
    C=1.0, # regularization
    kernel='linear'
```

```python
)
svc_regr_soft.fit(DATA["A"]["train"]["X"], DATA["A"]["train"]["Y"])

# SVC-Hard
svc_regr_hard = SVC(
    C=float('inf'), # regularization
    kernel='linear'
)
svc_regr_hard.fit(DATA["A"]["train"]["X"], DATA["A"]["train"]["Y"])
```

[22]:
```
SVC(C=inf, kernel='linear')
```

[23]:
```python
np.shape(DATA["A"]["train"]["X"])
```

[23]:
```
(2000, 50)
```

[24]:
```python
# SM-Soft : PerfectSeparationError: Perfect separation detected, results not␣
↪available
# sm_logit = sm.Logit(DATA["A"]["train"]["Y"],DATA["A"]["train"]["X"]).fit()
print("[SM]: PerfectSeparationError: Perfect separation detected, results not␣
↪available")
```

```
[SM]: PerfectSeparationError: Perfect separation detected, results not available
```

[25]:
```python
print('w_hard = ',svc_regr_hard.coef_)
print('b_hard = ',svc_regr_hard.intercept_)
print('w_soft = ',svc_regr_soft.coef_)
print('b_soft = ',svc_regr_soft.intercept_)

print('|w_hard|_2 - |w_soft|_2 = ', (np.linalg.norm(svc_regr_hard.coef_) - np.
↪linalg.norm(svc_regr_soft.coef_)))
```

```
w_hard =  [[-1.21665426e-01  3.62093715e-04 -8.14547914e-02 -1.30461544e-01
   1.03679744e-01 -1.52182440e-01  1.04733167e-01  2.37085565e-02
   1.88904300e-01  4.14259231e-01 -5.74918926e-02  2.53091455e-02
  -9.57967331e-02  2.78213946e-01  1.26402871e-02  4.24952941e-02
  -1.08058239e-01 -3.17898534e-02  3.32288614e-03  1.71401564e-01
  -6.93185355e-02 -2.85727264e-01 -1.11431511e-01 -2.81551300e-02
   7.14262962e-02  1.98803054e-01 -1.89996218e-01  9.63349060e-02
   1.20110591e-01 -5.53355480e-02 -4.29452659e-03 -1.10216493e-01
  -1.57449276e-01  3.64106600e-02 -4.35440021e-02 -1.42084333e-01
   1.30750877e-01  2.68330435e-02  6.77532548e-02 -3.01716444e-01
  -1.22263636e-02  2.15382941e-01  1.26537049e-01  2.24429271e-02
   1.34745230e-01  3.23138170e-02  1.47580798e-01  5.67770933e-02
   2.91490606e-01 -3.25700173e-02]]
b_hard =  [-0.]
w_soft =  [[-1.21665426e-01  3.62093715e-04 -8.14547914e-02 -1.30461544e-01
```

```
 1.03679744e-01 -1.52182440e-01  1.04733167e-01  2.37085565e-02
 1.88904300e-01  4.14259231e-01 -5.74918926e-02  2.53091455e-02
-9.57967331e-02  2.78213946e-01  1.26402871e-02  4.24952941e-02
-1.08058239e-01 -3.17898534e-02  3.32288614e-03  1.71401564e-01
-6.93185355e-02 -2.85727264e-01 -1.11431511e-01 -2.81551300e-02
 7.14262962e-02  1.98803054e-01 -1.89996218e-01  9.63349060e-02
 1.20110591e-01 -5.53355480e-02 -4.29452659e-03 -1.10216493e-01
-1.57449276e-01  3.64106600e-02 -4.35440021e-02 -1.42084333e-01
 1.30750877e-01  2.68330435e-02  6.77532548e-02 -3.01716444e-01
-1.22263636e-02  2.15382941e-01  1.26537049e-01  2.24429271e-02
 1.34745230e-01  3.23138170e-02  1.47580798e-01  5.67770933e-02
 2.91490606e-01 -3.25700173e-02]]
b_soft =  [-0.]
|w_hard|_2 - |w_soft|_2 =  0.0
```

[26]:
```python
# 3.2
x = DATA["A"]["train"]["X"]
y = DATA["A"]["train"]["Y"]
w = svc_regr_soft.coef_
# np.dot(x, w)
print("x: ",np.shape(x))
print("w: ",np.shape(w))
print("y: ",np.shape(y))
y[y==0] = -1

A = y * np.dot(w, np.transpose(x))

print("A: ",np.shape(A))
print("#A <= 1: ",np.sum(A <= 1))
```

```
x:  (2000, 50)
w:  (1, 50)
y:  (2000,)
A:  (1, 2000)
#A <= 1:  2000
```

[27]:
```python
print('w_soft = ',svc_regr_soft.coef_)
print('b_soft = ',svc_regr_soft.intercept_)
print("Support Vector:", svc_regr_soft.support_vectors_)
print("Support Vector Index:", svc_regr_soft.support_)
print("SV size:", np.shape(svc_regr_soft.support_vectors_))
print("Alpha:", svc_regr_soft.dual_coef_)
```

```
w_soft =  [[-1.21665426e-01  3.62093715e-04 -8.14547914e-02 -1.30461544e-01
 1.03679744e-01 -1.52182440e-01  1.04733167e-01  2.37085565e-02
 1.88904300e-01  4.14259231e-01 -5.74918926e-02  2.53091455e-02
-9.57967331e-02  2.78213946e-01  1.26402871e-02  4.24952941e-02
-1.08058239e-01 -3.17898534e-02  3.32288614e-03  1.71401564e-01
```

```
  -6.93185355e-02 -2.85727264e-01 -1.11431511e-01 -2.81551300e-02
   7.14262962e-02  1.98803054e-01 -1.89996218e-01  9.63349060e-02
   1.20110591e-01 -5.53355480e-02 -4.29452659e-03 -1.10216493e-01
  -1.57449276e-01  3.64106600e-02 -4.35440021e-02 -1.42084333e-01
   1.30750877e-01  2.68330435e-02  6.77532548e-02 -3.01716444e-01
  -1.22263636e-02  2.15382941e-01  1.26537049e-01  2.24429271e-02
   1.34745230e-01  3.23138170e-02  1.47580798e-01  5.67770933e-02
   2.91490606e-01 -3.25700173e-02]]
b_soft =  [-0.]
Support Vector: [[-23.3061044   -9.71882945   9.34264516 -20.5378218
-22.61745076
    0.02889478  -1.94438239 -16.76330649  -9.43733536   3.33701093
   -2.58531872 -24.78561978  18.36212309  19.90592558  -3.8111526
   -3.2063545    9.2496695   17.79626523  16.48625912 -15.4272087
    5.71399884   7.4487629    7.27841757  -5.6443613   15.10283527
   11.38375049  -4.3943539   11.64219184 -16.6523856  -12.96526166
    2.42225696 -23.25482721  17.76306384  18.11354686  25.20209425
   -8.9744753  -20.56208618   0.56792822  -3.2520854   16.08640176
   17.78726158   1.78486979   8.73822818   2.91542174  -2.06436836
  -11.22137302 -14.53169281 -14.28012612  26.60954328  16.71357943]
 [-23.54943526  -9.71810526   9.17973558 -20.79874489 -22.41009127
   -0.2754701   -1.73491606 -16.71588938  -9.05952676   4.1655294
   -2.70030251 -24.73500149  18.17052963  20.46235347  -3.78587202
   -3.12136391   9.03355302  17.73268552  16.49290489 -15.08440557
    5.57536177   6.87730838   7.05555455  -5.70067156  15.24568786
   11.7813566   -4.77434633  11.83486165 -16.41216442 -13.07593275
    2.41366791 -23.4752602   17.44816529  18.18636818  25.11500624
   -9.25864397 -20.30058443   0.62159431  -3.11657889  15.48296887
   17.76280885   2.21563567   8.99130228   2.96030759  -1.7948779
  -11.15674538 -14.23653121 -14.16657193  27.19252449  16.64843939]]
Support Vector Index: [1999 1998]
SV size: (2, 50)
Alpha: [[-0.5  0.5]]
```

[28]:
```
# dataset B
# SVC-Soft
svc_regr_soft = SVC(
    C=1.0, # regularization
    kernel='linear'
)
svc_regr_soft.fit(DATA["B"]["train"]["X"], DATA["B"]["train"]["Y"])
```

[28]:
```
SVC(kernel='linear')
```

[29]:
```
# SVC-Hard
# svc_regr_hard = SVC(
#     C=float('inf'), # regularization
```

```
#     kernel='linear'
# )
# svc_regr_hard.fit(DATA["B"]["train"]["X"], DATA["B"]["train"]["Y"])
print("SVC-HARD: INFINITY LOOP, UNSEPARABLE")
```

```
SVC-HARD: INFINITY LOOP, UNSEPARABLE
```

[30]:
```
# SM-Soft
sm_logit = sm.Logit(DATA["B"]["train"]["Y"],DATA["B"]["train"]["X"])
model = sm_logit.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.030979
         Iterations 14
```

[31]:
```
print('w_hard = ',svc_regr_hard.coef_)
print('b_hard = ',svc_regr_hard.intercept_)
print('w_soft = ',svc_regr_soft.coef_)
print('b_soft = ',svc_regr_soft.intercept_)
print('|w_hard|_2 - |w_soft|_2 = ', (np.linalg.norm(svc_regr_hard.coef_) - np.
 ↪linalg.norm(svc_regr_soft.coef_)))
# 3.2
x = DATA["B"]["train"]["X"]
y = DATA["B"]["train"]["Y"]
w = svc_regr_soft.coef_
# np.dot(x, w)
print("x: ",np.shape(x))
print("w: ",np.shape(w))
print("y: ",np.shape(y))
y[y==0] = -1
A = y * np.dot(w, np.transpose(x))
print("A: ",np.shape(A))
print("#A <= 1: ",np.sum(A <= 1))
print('w_soft = ',svc_regr_soft.coef_)
print('b_soft = ',svc_regr_soft.intercept_)
print("Support Vector:", svc_regr_soft.support_vectors_)
print("SV size:", np.shape(svc_regr_soft.support_vectors_))
print("Alpha:", svc_regr_soft.dual_coef_)
```

```
w_hard =  [[-1.21665426e-01  3.62093715e-04 -8.14547914e-02 -1.30461544e-01
   1.03679744e-01 -1.52182440e-01  1.04733167e-01  2.37085565e-02
   1.88904300e-01  4.14259231e-01 -5.74918926e-02  2.53091455e-02
  -9.57967331e-02  2.78213946e-01  1.26402871e-02  4.24952941e-02
  -1.08058239e-01 -3.17898534e-02  3.32288614e-03  1.71401564e-01
  -6.93185355e-02 -2.85727264e-01 -1.11431511e-01 -2.81551300e-02
   7.14262962e-02  1.98803054e-01 -1.89996218e-01  9.63349060e-02
   1.20110591e-01 -5.53355480e-02 -4.29452659e-03 -1.10216493e-01
  -1.57449276e-01  3.64106600e-02 -4.35440021e-02 -1.42084333e-01
```

```
   1.30750877e-01  2.68330435e-02  6.77532548e-02 -3.01716444e-01
  -1.22263636e-02  2.15382941e-01  1.26537049e-01  2.24429271e-02
   1.34745230e-01  3.23138170e-02  1.47580798e-01  5.67770933e-02
   2.91490606e-01 -3.25700173e-02]]
b_hard =  [-0.]
w_soft =  [[-0.14883452 -1.33157854 -2.62109121 -0.14008824 -0.01591273
-2.36571773
  -0.11301085 -0.33174302  1.25422302  0.12226576  0.71048243 -1.99753926
   0.59084758  1.56039593  0.2022904  -0.16566907  0.96702447  1.19357528
   1.66424776 -1.98261002  0.46273767  3.02203225 -0.45076031 -1.15875928
  -0.57344131 -1.52304994  1.23503273 -0.66182995 -0.0203479   2.09304347
   2.14280852 -0.46211822  2.15805064  0.61019298  0.01783408 -0.33238668
  -1.60256375  0.25252616  0.08694866  0.36201239 -0.38220834  1.01984528
  -0.58570554  0.41533726 -0.73605655 -1.77306584  0.77725884 -0.6126139
  -1.89797582 -0.12823335]]
b_soft =  [-0.05146275]
|w_hard|_2 - |w_soft|_2 =  -7.659534185945252
x:  (2000, 50)
w:  (1, 50)
y:  (2000,)
A:  (1, 2000)
#A <= 1:  161
w_soft =  [[-0.14883452 -1.33157854 -2.62109121 -0.14008824 -0.01591273
-2.36571773
  -0.11301085 -0.33174302  1.25422302  0.12226576  0.71048243 -1.99753926
   0.59084758  1.56039593  0.2022904  -0.16566907  0.96702447  1.19357528
   1.66424776 -1.98261002  0.46273767  3.02203225 -0.45076031 -1.15875928
  -0.57344131 -1.52304994  1.23503273 -0.66182995 -0.0203479   2.09304347
   2.14280852 -0.46211822  2.15805064  0.61019298  0.01783408 -0.33238668
  -1.60256375  0.25252616  0.08694866  0.36201239 -0.38220834  1.01984528
  -0.58570554  0.41533726 -0.73605655 -1.77306584  0.77725884 -0.6126139
  -1.89797582 -0.12823335]]
b_soft =  [-0.05146275]
Support Vector: [[ 2.46332952 -0.77978006 -0.27155939 …  0.13239303
-1.96125867
   1.38804837]
 [-1.42552454 -2.16288447 -0.88451354 … -0.41396881 -0.74861742
  -0.11140769]
 [-0.13560691 -0.2559248   0.69591211 … -0.72321138 -0.07651619
  -0.77946941]
 …
 [-0.55038792  1.6337104   0.84330336 … -0.26088387 -0.93452198
   1.06213669]
 [-0.19594668 -0.74682712  0.79477568 … -0.04955724  0.05319692
   0.1507748 ]
 [-1.18788362  0.98410019  0.95051311 … -0.61773492  0.64569337
  -1.12892607]]
SV size: (192, 50)
```

```
Alpha: [[-0.18154043 -1.         -1.         -1.         -1.         -1.
  -1.         -0.46579615 -0.65113809 -1.         -1.         -1.
  -1.         -1.         -0.28149788 -1.         -1.         -1.
  -0.13851952 -1.         -1.         -1.         -0.6550175  -0.90999031
  -0.8893523  -0.41517747 -1.         -1.         -1.         -0.12184268
  -0.5046161  -1.         -0.72350645 -1.         -1.         -1.
  -1.         -1.         -0.84901527 -1.         -0.19431178 -1.
  -1.         -1.         -1.         -1.         -0.27641444 -1.
  -1.         -1.         -1.         -1.         -1.         -1.
  -1.         -1.         -1.         -1.         -1.         -1.
  -0.28400342 -1.         -1.         -0.7332253  -1.         -1.
  -1.         -0.79840101 -1.         -1.         -1.         -1.
  -1.         -1.         -1.         -1.         -0.76305116 -1.
  -1.         -1.         -1.         -1.         -1.         -1.
  -1.         -1.         -1.         -1.         -1.         -0.63660558
  -1.         -1.          0.65196242  0.42485213  0.35353782  1.
   0.94539376  1.          0.3388051   1.          1.          1.
   0.43223143  1.          1.          1.          0.90793149  1.
   1.          1.          0.04970768  1.          1.          1.
   1.          0.97289561  1.          1.          0.9196539   1.
   1.          0.93216669  1.          1.          0.51862456  0.7570981
   0.06533931  0.11250122  1.          0.65794784  1.          0.49810779
   1.          1.          1.          0.37562404  1.          0.03267648
   1.          1.          1.          1.          1.          0.65694822
   0.51235981  1.          1.          1.          1.          1.
   1.          1.          1.          1.          1.          1.
   0.29611049  0.08661775  1.          1.          0.20712176  0.08321326
   0.74696912  0.20421004  1.          1.          1.          1.
   1.          1.          1.          1.          0.8872863   0.49311911
   1.          1.          1.          1.          1.          0.33433675
   1.          1.          0.76428281  0.0078512   1.          0.42621879
   1.          1.          1.          0.81932003  1.          1.        ]]
```

[32]:
```python
result_right = sum(DATA["B"]["test"]["Y"] == svc_regr_soft.
 ↪predict(DATA["B"]["test"]["X"]))
empirical_accuracy = (result_right)/len(DATA["B"]["test"]["Y"])
print("Soft-SVM Test Performance (Empirical Accuracy): {}%".
 ↪format(empirical_accuracy*100))
```

```
Soft-SVM Test Performance (Empirical Accuracy): 97.15%
```

[33]:
```python
result_right = sum(DATA["B"]["test"]["Y"] == np.round(model.
 ↪predict(DATA["B"]["test"]["X"])))
empirical_accuracy = (result_right)/len(DATA["B"]["test"]["Y"])
print("Logit Test Performance (Empirical Accuracy): {}%".
 ↪format(empirical_accuracy*100))
```

```
Logit Test Performance (Empirical Accuracy): 96.95%
```