LS_Handy-CardenasL

November 18, 2020

[4]:

```python
import numpy as np
# from spicy.io import loadmat
import matplotlib.pyplot as plt

#A = np.genfromtxt('Data_Raw.csv', delimiter=',')
#print(A.dtype)

Data = np.genfromtxt('Data.csv', delimiter=',')
x_train = Data[0:1001,0:14] # features
y_train = Data[0:1001,14] # corresponding labels

# evaluation data
x_eval= Data[1001:12330,0:14] # features
y_eval = Data[1001:12330,14] # corresponding labels

# X = Data[0:3,0:14]
# y = Data[:,14]

# Classifier 1
#w = (X^T X)^(-1)X^T y
X = x_train
y = y_train
w = np.linalg.inv(X.transpose()@X)@X.transpose()@y
#A = np.linalg.inv(X@X.T)

print(np.round(w,2))
```

```
[ 0.01  0.    0.02 -0.   -0.    0.    0.04 -0.09  0.01  0.01 -0.01  0.
 -0.    0.  ]
```

[5]:

```python
import numpy as np
# from spicy.io import loadmat
import matplotlib.pyplot as plt

#A = np.genfromtxt('Data_Raw.csv', delimiter=',')
#print(A.dtype)

Data = np.genfromtxt('Data.csv', delimiter=',')
```

```
x_train = Data[0:1001,0:14] # features
y_train = Data[0:1001,14] # corresponding labels

# evaluation data
x_eval= Data[1001:12330,0:14] # features
y_eval = Data[1001:12330,14] # corresponding labels

# X = Data[0:3,0:14]
# y = Data[:,14]

# Classifier 1
#w = (X^T X)^(-1)X^T y
X = x_eval
y = y_eval
w = np.linalg.inv(X.transpose()@X)@X.transpose()@y
#A = np.linalg.inv(X@X.T)

print(np.round(w,2))
```

```
[nan nan nan nan nan nan nan nan nan nan nan nan nan nan]
```

[ ]:

```
# 2d

x_eval = X
y_eval = y

# all features
print('considering all features')
x_train = X
w_train = w
y_hat = np.sign(x_train@w_train)

error_vec = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_eval))]
print('Errors: '+ str(sum(error_vec)))
print('Percent error: '+str(100.0*sum(error_vec)/len(error_vec))+'%')

# 3 main features
print('considering 3 main features')
x_train = X[:, [0, 2, 3]]
w_train = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y
y_hat = np.sign(x_train@w_train)

error_vec = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_eval))]
print('Errors: '+ str(sum(error_vec)))
print('Percent error: '+str(100.0*sum(error_vec)/len(error_vec))+'%')

# w = (X^T X)^(-1)X^T y
# w_opt = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y_train
# print(w_opt)
# y_hat = np.sign(x_eval@w_opt)
```

[ ]:

```
# 2f
in_data = loadmat('face_emotion_data.mat')

X = in_data['X']
y = in_data['y']

x_1 = X[range(0,16), :]
x_2 = X[range(16,32), :]
x_3 = X[range(16,48), :]
x_4 = X[range(48,64), :]
x_5 = X[range(64,80), :]
x_6 = X[range(80,96), :]
x_7 = X[range(96,112), :]
x_8 = X[range(112,128), :]

y_1 = y[range(0,16), :]
y_2 = y[range(16,32), :]
y_3 = y[range(16,48), :]
y_4 = y[range(48,64), :]
y_5 = y[range(64,80), :]
y_6 = y[range(80,96), :]
y_7 = y[range(96,112), :]
y_8 = y[range(112,128), :]

# 1 variation, hold x_8 and y_8
print('considering 1st variation')
x_train = np.vstack((x_1, x_2, x_3, x_4, x_5, x_6, x_7))
y_train = np.vstack((y_1, y_2, y_3, y_4, y_5, y_6, y_7))
w_train = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y_train
y_hat = np.sign(x_8@w_train)
# print(w_train)
error_vec_1 = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_8))]
print('Errors: '+ str(sum(error_vec_1)))
print('Percent error: '+str(100.0*sum(error_vec_1)/16)+'%')

# 2 variation, hold x_1 and y_1
print('considering 2nd variation')
x_train = np.vstack((x_2, x_3, x_4, x_5, x_6, x_7, x_8))
y_train = np.vstack((y_2, y_3, y_4, y_5, y_6, y_7, y_8))
w_train = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y_train
y_hat = np.sign(x_1@w_train)
# print(w_train)
error_vec_2 = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_1))]
```

```python
print('Errors: '+ str(sum(error_vec_2)))
print('Percent error: '+str(100.0*sum(error_vec_2)/16)+'%')

# 3 variation, , hold x_2 and y_2
print('considering 3rd variation')
x_train = np.vstack((x_1, x_3, x_4, x_5, x_6, x_7, x_8))
y_train = np.vstack((y_1, y_3, y_4, y_5, y_6, y_7, y_8))
w_train = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y_train
y_hat = np.sign(x_2@w_train)
# print(w_train)
error_vec_3 = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_2))]
print('Errors: '+ str(sum(error_vec_3)))
print('Percent error: '+str(100.0*sum(error_vec_3)/16)+'%')

# 4 variation, hold x_3 and y_3
print('considering 4th variation')
x_train = np.vstack((x_1, x_2, x_4, x_5, x_6, x_7, x_8))
y_train = np.vstack((y_1, y_2, y_4, y_5, y_6, y_7, y_8))
w_train = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y_train
y_hat = np.sign(x_3@w_train)
# print(w_train)
error_vec_4 = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_3))]
print('Errors: '+ str(sum(error_vec_4)))
print('Percent error: '+str(100.0*sum(error_vec_4)/16)+'%')

# 5 variation, hold x_4 and y_4
print('considering 5th variation')
x_train = np.vstack((x_1, x_2, x_3, x_5, x_6, x_7, x_8))
y_train = np.vstack((y_1, y_2, y_3, y_5, y_6, y_7, y_8))
w_train = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y_train
y_hat = np.sign(x_4@w_train)
# print(w_train)
error_vec_5 = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_4))]
print('Errors: '+ str(sum(error_vec_5)))
print('Percent error: '+str(100.0*sum(error_vec_5)/16)+'%')

# 6 variation, hold x_5 and y_5
print('considering 6th variation')
x_train = np.vstack((x_1, x_2, x_3, x_4, x_6, x_7, x_8))
y_train = np.vstack((y_1, y_2, y_3, y_4, y_6, y_7, y_8))
w_train = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y_train
y_hat = np.sign(x_5@w_train)
# print(w_train)
error_vec_6 = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_5))]
print('Errors: '+ str(sum(error_vec_6)))
print('Percent error: '+str(100.0*sum(error_vec_6)/16)+'%')
```

```
# 7 variation, hold x_6 and y_6
print('considering 7th variation')
x_train = np.vstack((x_1, x_2, x_3, x_4, x_5, x_7, x_8))
y_train = np.vstack((y_1, y_2, y_3, y_4, y_5, y_7, y_8))
w_train = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y_train
y_hat = np.sign(x_6@w_train)
# print(w_train)
error_vec_7 = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_6))]
print('Errors: '+ str(sum(error_vec_7)))
print('Percent error: '+str(100.0*sum(error_vec_7)/16)+'%')

# 8 variation, hold x_7 and y_7
print('considering 8th variation')
x_train = np.vstack((x_1, x_2, x_3, x_4, x_5, x_6, x_8))
y_train = np.vstack((y_1, y_2, y_3, y_4, y_5, y_6, y_8))
w_train = np.linalg.inv(x_train.transpose()@x_train)@x_train.transpose()@y_train
y_hat = np.sign(x_7@w_train)
# print(w_train)
error_vec_8 = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat, y_7))]
print('Errors: '+ str(sum(error_vec_8)))
print('Percent error: '+str(100.0*sum(error_vec_8)/16)+'%')

print('Final performance estimate:')
#␣
 ↪average_error_rate=(error_vec_1+error_vec_2+error_vec_3+error_vec_4+error_vec_5+error_vec_6
 ↪8
hola = ((1/
 ↪16)*(sum(error_vec_1)+sum(error_vec_2)+sum(error_vec_3)+sum(error_vec_4)+sum(error_vec_5)+s
 ↪8)*100
print(str(hola)+'%')
```

```
[ ]:
```

```
[ ]:
```