Program no. 5

Implementing regularization to avoid overfitting in binary classification

```
from sklearn.datasets import make_moons
from keras.models import Sequential
from keras.layers import Dense
```

```
X,y= make_moons(n_samples=100,noise=0.2,random_state=1)
```

```
X.shape
```

```
    (100, 2)
```

```
y.shape
```

```
    (100,)
```

```
X_train,Xtest=X[:30,:],X[30:,:]
```

```
Y_train,Ytest=y[:30],y[30:]
```

```
model=Sequential()
model.add(Dense(500,input_dim=2,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
```

```
model.compile(loss='binary_crossentropy',optimizer='adam',metrics='accuracy')
```

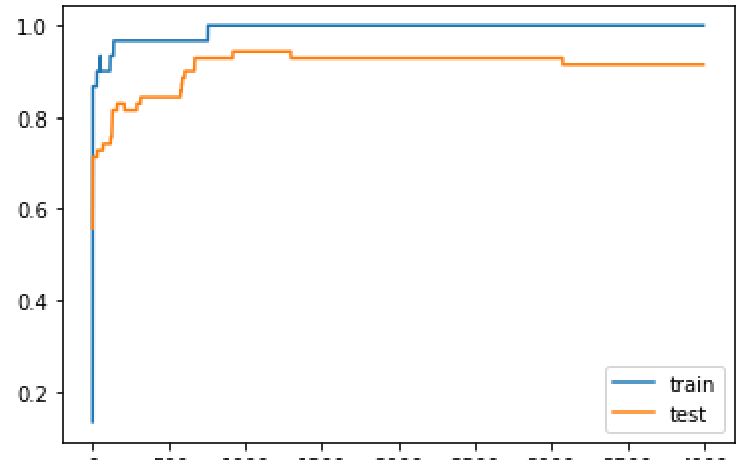
```
history=model.fit(X_train,Y_train,validation_data=(Xtest,Ytest),epochs=4000)
```

```
    1/1 [==============================] - 0s 47ms/step - loss: 2.8060e-04 - accuracy:
    Epoch 3890/4000
    1/1 [==============================] - 0s 57ms/step - loss: 2.8036e-04 - accuracy:
    Epoch 3891/4000
    1/1 [==============================] - 0s 59ms/step - loss: 2.8013e-04 - accuracy:
    Epoch 3892/4000
    1/1 [==============================] - 0s 59ms/step - loss: 2.7990e-04 - accuracy:
    Epoch 3893/4000
    1/1 [==============================] - 0s 47ms/step - loss: 2.7969e-04 - accuracy:
    Epoch 3894/4000
    1/1 [==============================] - 0s 42ms/step - loss: 2.7947e-04 - accuracy:
    Epoch 3895/4000
```

```
1/1 [==============================] - 0s 60ms/step - loss: 2.7924e-04 - accuracy:
Epoch 3896/4000
1/1 [==============================] - 0s 49ms/step - loss: 2.7902e-04 - accuracy:
Epoch 3897/4000
1/1 [==============================] - 0s 38ms/step - loss: 2.7877e-04 - accuracy:
Epoch 3898/4000
1/1 [==============================] - 0s 35ms/step - loss: 2.7855e-04 - accuracy:
Epoch 3899/4000
1/1 [==============================] - 0s 52ms/step - loss: 2.7834e-04 - accuracy:
Epoch 3900/4000
1/1 [==============================] - 0s 55ms/step - loss: 2.7811e-04 - accuracy:
Epoch 3901/4000
1/1 [==============================] - 0s 62ms/step - loss: 2.7787e-04 - accuracy:
Epoch 3902/4000
1/1 [==============================] - 0s 35ms/step - loss: 2.7765e-04 - accuracy:
Epoch 3903/4000
1/1 [==============================] - 0s 54ms/step - loss: 2.7742e-04 - accuracy:
Epoch 3904/4000
1/1 [==============================] - 0s 53ms/step - loss: 2.7722e-04 - accuracy:
Epoch 3905/4000
1/1 [==============================] - 0s 67ms/step - loss: 2.7700e-04 - accuracy:
Epoch 3906/4000
1/1 [==============================] - 0s 40ms/step - loss: 2.7677e-04 - accuracy:
Epoch 3907/4000
1/1 [==============================] - 0s 52ms/step - loss: 2.7653e-04 - accuracy:
Epoch 3908/4000
1/1 [==============================] - 0s 54ms/step - loss: 2.7631e-04 - accuracy:
Epoch 3909/4000
1/1 [==============================] - 0s 37ms/step - loss: 2.7609e-04 - accuracy:
Epoch 3910/4000
1/1 [==============================] - 0s 58ms/step - loss: 2.7588e-04 - accuracy:
Epoch 3911/4000
1/1 [==============================] - 0s 53ms/step - loss: 2.7565e-04 - accuracy:
Epoch 3912/4000
1/1 [==============================] - 0s 55ms/step - loss: 2.7542e-04 - accuracy:
Epoch 3913/4000
1/1 [==============================] - 0s 41ms/step - loss: 2.7520e-04 - accuracy:
Epoch 3914/4000
1/1 [==============================] - 0s 41ms/step - loss: 2.7498e-04 - accuracy:
Epoch 3915/4000
1/1 [==============================] - 0s 53ms/step - loss: 2.7476e-04 - accuracy:
Epoch 3916/4000
1/1 [==============================] - 0s 56ms/step - loss: 2.7453e-04 - accuracy:
Epoch 3917/4000
1/1 [==============================] - 0s 56ms/step - loss: 2.7432e-04 - accuracy:
```

```
from matplotlib import pyplot
pyplot.plot(history.history['accuracy'],label='train')
pyplot.plot(history.history['val_accuracy'],label='test')
pyplot.legend()
```

```
<matplotlib.legend.Legend at 0x7fa72ad87ad0>
```



L2 Regularization

```
from keras.regularizers import l2
model_l2=Sequential()
model_l2.add(Dense(500,input_dim=2,activation='relu',kernel_regularizer=l2(0.001)))
model_l2.add(Dense(1,activation='sigmoid'))
```

```
model_l2.compile(loss='binary_crossentropy',optimizer='adam',metrics='accuracy')
```

```
history2=model_l2.fit(X_train,Y_train,validation_data=(Xtest,Ytest),epochs=4000)
```

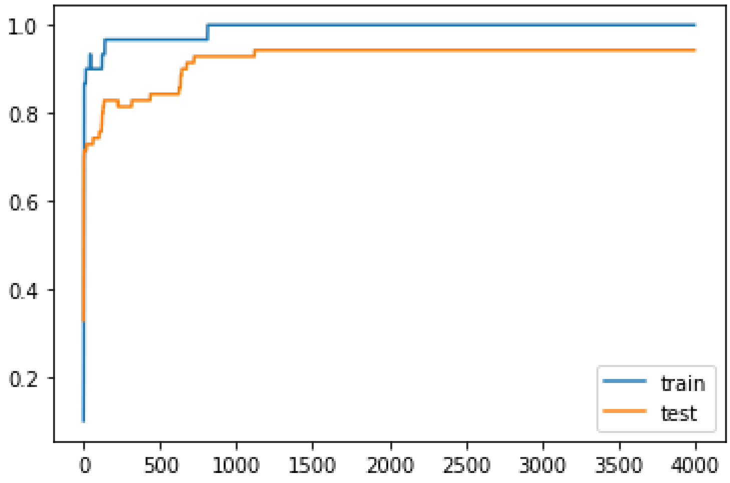
```
1/1 [==============================] - 0s 50ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3962/4000
1/1 [==============================] - 0s 52ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3963/4000
1/1 [==============================] - 0s 47ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3964/4000
1/1 [==============================] - 0s 53ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3965/4000
1/1 [==============================] - 0s 38ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3966/4000
1/1 [==============================] - 0s 52ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3967/4000
1/1 [==============================] - 0s 37ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3968/4000
1/1 [==============================] - 0s 52ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3969/4000
1/1 [==============================] - 0s 37ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3970/4000
1/1 [==============================] - 0s 37ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3971/4000
1/1 [==============================] - 0s 48ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3972/4000
1/1 [==============================] - 0s 55ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3973/4000
1/1 [==============================] - 0s 52ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3974/4000
1/1 [==============================] - 0s 55ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3975/4000
1/1 [==============================] - 0s 53ms/step - loss: 0.0158 - accuracy: 1.0

Epoch 3976/4000
1/1 [==============================] - 0s 58ms/step - loss: 0.0158 - accuracy: 1.0
```

```
1/1 [==============================] - 0s 58ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3977/4000
1/1 [==============================] - 0s 46ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3978/4000
1/1 [==============================] - 0s 37ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3979/4000
1/1 [==============================] - 0s 52ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3980/4000
1/1 [==============================] - 0s 53ms/step - loss: 0.0158 - accuracy: 1.0
Epoch 3981/4000
1/1 [==============================] - 0s 53ms/step - loss: 0.0157 - accuracy: 1.0
Epoch 3982/4000
1/1 [==============================] - 0s 57ms/step - loss: 0.0157 - accuracy: 1.0
Epoch 3983/4000
1/1 [==============================] - 0s 55ms/step - loss: 0.0157 - accuracy: 1.0
Epoch 3984/4000
1/1 [==============================] - 0s 54ms/step - loss: 0.0157 - accuracy: 1.0
Epoch 3985/4000
1/1 [==============================] - 0s 44ms/step - loss: 0.0157 - accuracy: 1.0
Epoch 3986/4000
1/1 [==============================] - 0s 57ms/step - loss: 0.0157 - accuracy: 1.0
Epoch 3987/4000
1/1 [==============================] - 0s 39ms/step - loss: 0.0157 - accuracy: 1.0
Epoch 3988/4000
1/1 [==============================] - 0s 53ms/step - loss: 0.0157 - accuracy: 1.0
Epoch 3989/4000
1/1 [==============================] - 0s 55ms/step - loss: 0.0157 - accuracy: 1.0
```

```
pyplot.plot(history2.history['accuracy'],label='train')
pyplot.plot(history2.history['val_accuracy'],label='test')
pyplot.legend()
```

```
<matplotlib.legend.Legend at 0x7fa72acd2f50>
```



L1 regularization (Lasso regularization)

```
from keras.regularizers import l1
model_l1=Sequential()
model_l1.add(Dense(500,input_dim=2,activation='relu',kernel_regularizer=l1(0.0001)))
model_l1.add(Dense(1,activation='sigmoid'))
```

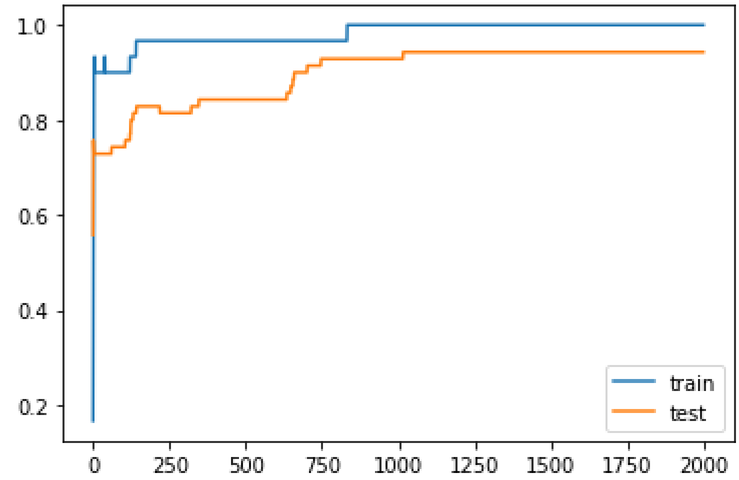
```
model_l1.compile(loss='binary_crossentropy',optimizer='adam',metrics='accuracy')
```

```
history3=model_l1.fit(X_train,Y_train,validation_data=(Xtest,Ytest),epochs=2000)
```

```
1/1 [==============================] - 0s 41ms/step - loss: 0.0151 - accuracy: 1.0
Epoch 1815/2000
1/1 [==============================] - 0s 46ms/step - loss: 0.0151 - accuracy: 1.0
Epoch 1816/2000
1/1 [==============================] - 0s 59ms/step - loss: 0.0151 - accuracy: 1.0
Epoch 1817/2000
1/1 [==============================] - 0s 48ms/step - loss: 0.0151 - accuracy: 1.0
Epoch 1818/2000
1/1 [==============================] - 0s 37ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1819/2000
1/1 [==============================] - 0s 43ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1820/2000
1/1 [==============================] - 0s 54ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1821/2000
1/1 [==============================] - 0s 43ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1822/2000
1/1 [==============================] - 0s 43ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1823/2000
1/1 [==============================] - 0s 50ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1824/2000
1/1 [==============================] - 0s 41ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1825/2000
1/1 [==============================] - 0s 55ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1826/2000
1/1 [==============================] - 0s 55ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1827/2000
1/1 [==============================] - 0s 52ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1828/2000
1/1 [==============================] - 0s 51ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1829/2000
1/1 [==============================] - 0s 53ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1830/2000
1/1 [==============================] - 0s 43ms/step - loss: 0.0150 - accuracy: 1.0
Epoch 1831/2000
1/1 [==============================] - 0s 54ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1832/2000
1/1 [==============================] - 0s 56ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1833/2000
1/1 [==============================] - 0s 56ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1834/2000
1/1 [==============================] - 0s 42ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1835/2000
1/1 [==============================] - 0s 43ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1836/2000
1/1 [==============================] - 0s 41ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1837/2000
1/1 [==============================] - 0s 40ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1838/2000
1/1 [==============================] - 0s 55ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1839/2000
1/1 [==============================] - 0s 53ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1840/2000
1/1 [==============================] - 0s 56ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1841/2000
1/1 [==============================] - 0s 39ms/step - loss: 0.0149 - accuracy: 1.0
Epoch 1842/2000
Epoch 1842/2000
1/1 [==============================] - 0s 41ms/step - loss: 0.0149 - accuracy: 1.00
```

```
pyplot.plot(history3.history['accuracy'],label='train')
pyplot.plot(history3.history['val_accuracy'],label='test')
pyplot.legend()
```

```
<matplotlib.legend.Legend at 0x7fa72829e690>
```



L1 and L2 regularization

```
from keras.regularizers import l1_l2
model_l12=Sequential()
model_l12.add(Dense(500,input_dim=2,activation='relu',kernel_regularizer=l1_l2(l1=0.001,l2
model_l12.add(Dense(1,activation='sigmoid'))
```

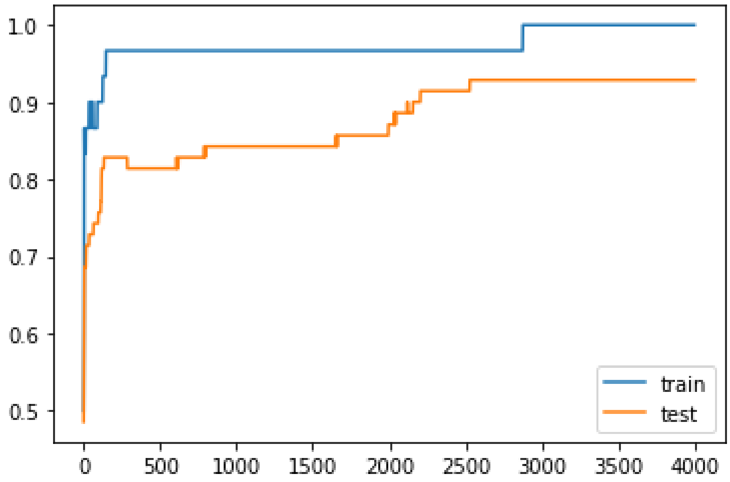
```
model_l12.compile(loss='binary_crossentropy',optimizer='adam',metrics='accuracy')
```

```
history4=model_l12.fit(X_train,Y_train,validation_data=(Xtest,Ytest),epochs=4000)
```

```
1/1 [==============================] - 0s 36ms/step - loss: 0.0500 - accuracy: 1.00
Epoch 3939/4000
1/1 [==============================] - 0s 54ms/step - loss: 0.0499 - accuracy: 1.00
Epoch 3940/4000
1/1 [==============================] - 0s 48ms/step - loss: 0.0499 - accuracy: 1.00
Epoch 3941/4000
1/1 [==============================] - 0s 39ms/step - loss: 0.0499 - accuracy: 1.00
Epoch 3942/4000
1/1 [==============================] - 0s 38ms/step - loss: 0.0499 - accuracy: 1.00
Epoch 3943/4000
1/1 [==============================] - 0s 35ms/step - loss: 0.0499 - accuracy: 1.00
Epoch 3944/4000
1/1 [==============================] - 0s 53ms/step - loss: 0.0499 - accuracy: 1.00
Epoch 3945/4000
1/1 [==============================] - 0s 37ms/step - loss: 0.0499 - accuracy: 1.00
Epoch 3946/4000
1/1 [==============================] - 0s 42ms/step - loss: 0.0499 - accuracy: 1.00
Epoch 3947/4000
Epoch 3947/4000
1/1 [==============================] - 0s 42ms/step - loss: 0.0498 - accuracy: 1.0
Epoch 3948/4000
1/1 [==============================] - 0s 35ms/step - loss: 0.0498 - accuracy: 1.0
Epoch 3949/4000
1/1 [==============================] - 0s 40ms/step - loss: 0.0498 - accuracy: 1.0
Epoch 3950/4000
1/1 [==============================] - 0s 62ms/step - loss: 0.0498 - accuracy: 1.0
Epoch 3951/4000
1/1 [==============================] - 0s 54ms/step - loss: 0.0498 - accuracy: 1.0
Epoch 3952/4000
1/1 [==============================] - 0s 54ms/step - loss: 0.0498 - accuracy: 1.0
Epoch 3953/4000
1/1 [==============================] - 0s 44ms/step - loss: 0.0498 - accuracy: 1.0
Epoch 3954/4000
1/1 [==============================] - 0s 54ms/step - loss: 0.0497 - accuracy: 1.0
Epoch 3955/4000
1/1 [==============================] - 0s 53ms/step - loss: 0.0497 - accuracy: 1.0
Epoch 3956/4000
1/1 [==============================] - 0s 53ms/step - loss: 0.0497 - accuracy: 1.0
Epoch 3957/4000
1/1 [==============================] - 0s 64ms/step - loss: 0.0497 - accuracy: 1.0
Epoch 3958/4000
1/1 [==============================] - 0s 44ms/step - loss: 0.0497 - accuracy: 1.0
Epoch 3959/4000
1/1 [==============================] - 0s 61ms/step - loss: 0.0497 - accuracy: 1.0
Epoch 3960/4000
1/1 [==============================] - 0s 45ms/step - loss: 0.0497 - accuracy: 1.0
Epoch 3961/4000
1/1 [==============================] - 0s 52ms/step - loss: 0.0497 - accuracy: 1.0
Epoch 3962/4000
1/1 [==============================] - 0s 60ms/step - loss: 0.0496 - accuracy: 1.0
Epoch 3963/4000
1/1 [==============================] - 0s 43ms/step - loss: 0.0496 - accuracy: 1.0
Epoch 3964/4000
1/1 [==============================] - 0s 43ms/step - loss: 0.0496 - accuracy: 1.0
Epoch 3965/4000
1/1 [==============================] - 0s 44ms/step - loss: 0.0496 - accuracy: 1.0
Epoch 3966/4000
1/1 [==============================] - 0s 39ms/step - loss: 0.0496 - accuracy: 1.0
Epoch 3967/4000
```

```
pyplot.plot(history4.history['accuracy'],label='train')
pyplot.plot(history4.history['val_accuracy'],label='test')
pyplot.legend()
```

```
<matplotlib.legend.Legend at 0x7fa727e0d9d0>
```


1.0
0.9
0.8
0.7
0.6
0.5
0
500
1000
1500
2000
2500
3000
3500
4000
train
test

●