Aim : Prove that when sigmoid activation function is used in hidden layer of neural network, the neural network suffer from vanishing gradient problem When to stop epoch of the model, there is a chance of overfitting

In [1]:

```
import sklearn.datasets
import matplotlib.pyplot as plt
```

In [2]:

```
x,y = sklearn.datasets.make_moons(n_samples=500,noise=0.1, random_state=25)
```

In [3]:

```
x
```

```
       [ 1.99504243, -0.12327768],
       [ 0.88068567, -0.44985994],
       [ 0.60023489, -0.53450188],
       [ 0.76241558, -0.21361533],
       [ 1.36890178, -0.49073241],
       [ 2.04405696,  0.15148708],
       [ 0.26950448,  0.97724599],
       [ 1.47324915, -0.41226699],
       [ 0.11471465,  0.29867123],
       [-0.64460467,  0.87344843],
       [ 2.08066339,  0.30217829],
       [ 1.92941428,  0.39332646],
       [ 1.9230529 ,  0.17393372],
       [ 0.13598423,  0.00419466],
       [ 1.57411289, -0.33740722],
       [ 1.85501796, -0.03113936],
       [ 1.43584879, -0.33321726],
       [ 0.89372777,  0.49922374],
       [ 0.73360886,  0.67669592],
```

In [4]:

```
y
```

Out[4]:

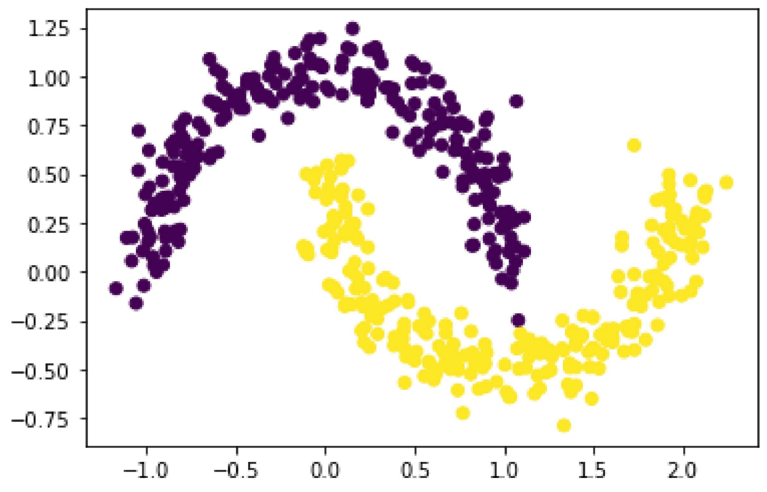
```
array([1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1,
       1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
       1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0,
       1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1,
       0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0,
       0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1,
       0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0,
       1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0,
       1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0,
       0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0,
       1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0,
       1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1,
       0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1,
       0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1,
       1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0], dtype=int64)
```

In [5]:

```
plt.scatter(x[:,0],x[:,1],c=y)
```

Out[5]:

```
<matplotlib.collections.PathCollection at 0x1e030253b80>
```



In [6]:

```
import numpy as np
from keras.layers import Dense
from keras.models import Sequential
```

In [7]:

```
model = Sequential()
```

In [8]:

```
model.add(Dense(units=128,activation='relu',input_dim=2))
model.add(Dense(units=128,activation='relu'))
model.add(Dense(units=128,activation='relu'))
model.add(Dense(units=1,activation='sigmoid'))
```

In [9]:

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

In [10]:

```
# model.fit(x,y,epochs=400,batch_size=16)
```

In [11]:

```
model.get_weights()
```

```
        [-0.11381148],
        [-0.01008773],
        [-0.12583409],
        [-0.17437677],
        [-0.08217353],
        [-0.08729662],
        [ 0.06237853],
        [ 0.079804  ],
        [-0.18366534],
        [-0.02647573],

        [ 0.20436168],
        [ 0.02822761],
        [ 0.15560889],
        [-0.10054444],
        [ 0.10066548],
        [ 0.06770143],
        [ 0.07917875],
        [-0.09317485],
        [ 0.0472858 ],
```

In [12]:

```
old_weight = model.get_weights()[0]
old_weight.shape
```

Out[12]:

```
(2, 128)
```

In [13]:

```
old_weight
```

Out[13]:

```
array([[-0.14066571, -0.17640314,  0.16030733,  0.18408863,  0.21211748,
         0.01056695, -0.15345   ,  0.08681957,  0.20820485, -0.21087292,
         0.08718164,  0.02184786, -0.09367666,  0.03757267, -0.21100184,
         0.10698442, -0.16799457, -0.13263857, -0.1632364 , -0.06933339,
         0.03399502, -0.14949581,  0.12152244,  0.05309992, -0.03542621,
        -0.04240061, -0.11420482,  0.06504075, -0.13834086,  0.07796045,
        -0.169397  ,  0.11520757,  0.21463884, -0.21298115, -0.09438437,
        -0.00329296, -0.1592713 , -0.04296787, -0.14841333, -0.01304777,
         0.13495462,  0.13869049, -0.11280737, -0.03444611,  0.00563774,
        -0.12116375, -0.0534417 ,  0.10251068,  0.06810047, -0.04562171,
        -0.03198707, -0.1556223 , -0.0708784 ,  0.14271764,  0.19169043,
         0.1589561 ,  0.1635388 , -0.04830839, -0.06645684, -0.08599681,
        -0.16025648,  0.01575196,  0.15368633, -0.06245555, -0.10693023,
        -0.17887776, -0.20737804, -0.17829594,  0.08083589,  0.10546593,
         0.01378615, -0.12792265, -0.16092727,  0.13128658,  0.0105105 ,
         0.12448867, -0.20701201,  0.20101272,  0.11069272, -0.14963329,
         0.03365839, -0.12439873, -0.12669721,  0.1839401 , -0.10543008,
         0.10833357,  0.13967796,  0.18909498,  0.06740166, -0.01597932,
         0.19179024,  0.19492562,  0.03295933,  0.06515534, -0.15823409,
        -0.20542306,  0.04250447, -0.20327747,  0.16177998,  0.15683727,
         0.00276734, -0.08283077, -0.06962775,  0.19444828, -0.01893571,
        -0.1621668 ,  0.16602488, -0.16771619,  0.19289048, -0.13047066,
         0.13142471, -0.21131629,  0.185434  ,  0.01871096,  0.04384415,
        -0.13017061, -0.02454877,  0.08537944, -0.02007271,  0.17362632,
         0.04315852, -0.02746548,  0.14189525,  0.19344471, -0.13813649,
         0.0608678 , -0.16757329,  0.18952595],
       [ 0.12721227, -0.01333404,  0.01160473, -0.03786452,  0.18147884,
         0.12272634,  0.15954207, -0.09525861,  0.1235161 , -0.02293374,
         0.15730138,  0.2031299 , -0.01890339,  0.07651739, -0.13075373,
        -0.17556281,  0.04028188,  0.09168781,  0.00745289,  0.05869924,
        -0.10985369,  0.17697148, -0.07558921, -0.03643388,  0.11912583,
        -0.07325684,  0.15934978, -0.19687445,  0.04626741,  0.20991509,
         0.0267764 ,  0.18165581,  0.12717827, -0.11831076, -0.07374327,
         0.03587   ,  0.1676646 , -0.00626601,  0.18548234,  0.12641616,
        -0.02377616,  0.10436879,  0.17860107,  0.10100873, -0.0433068 ,
         0.05818559, -0.17822945, -0.20007937, -0.06453387, -0.07841003,
         0.04293801,  0.09484009,  0.08683442, -0.0172247 , -0.16923796,
         0.08777688,  0.09258719,  0.076288  ,  0.14923419, -0.02030335,
        -0.135705  ,  0.14245136,  0.05459104,  0.07145013,  0.1468559 ,
         0.19300883, -0.15627353, -0.19598767, -0.19497606,  0.02812894,
        -0.02582759,  0.17581205,  0.11106525, -0.16585755,  0.1337594 ,
        -0.16976911, -0.2140935 , -0.07387783,  0.02760044,  0.15614854,
         0.15342094,  0.11899291, -0.16045071,  0.12268613, -0.15138733,
         0.21284764, -0.06270739, -0.08594389, -0.09539685, -0.08466579,
         0.10743038, -0.1032485 , -0.03167979,  0.0704592 ,  0.13652335,
        -0.09506279, -0.06384777,  0.11338495, -0.15867105, -0.07328244,
         0.17899735, -0.00344166,  0.1707802 , -0.09681325,  0.07868312,
         0.16620766,  0.01168515,  0.20274733, -0.15826134,  0.17960827,
         0.14306666, -0.05349067, -0.15344979,  0.10092093, -0.07830031,
         0.0547093 , -0.16564442, -0.1523255 , -0.12192059,  0.20041113,
        -0.04716817, -0.05873929,  0.05348791, -0.07492968,  0.02487141,
         0.0881262 ,  0.00880286, -0.06410521]], dtype=float32)
```

In [14]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=42)
```

In [15]:

```
model.fit(X_train,y_train,epochs=100)
```

```
Epoch 30/100
13/13 [==============================] - 0s 2ms/step - loss: 0.0033 - accu
racy: 1.0000
Epoch 31/100
13/13 [==============================] - 0s 2ms/step - loss: 0.0028 - accu
racy: 1.0000
Epoch 32/100
13/13 [==============================] - 0s 1ms/step - loss: 0.0025 - accu
racy: 1.0000
Epoch 33/100
13/13 [==============================] - 0s 1ms/step - loss: 0.0023 - accu
racy: 1.0000
Epoch 34/100
13/13 [==============================] - 0s 1ms/step - loss: 0.0020 - accu
racy: 1.0000
Epoch 35/100
13/13 [==============================] - 0s 1ms/step - loss: 0.0018 - accu
racy: 1.0000
Epoch 36/100
13/13 [==============================] - 0s 1ms/step - loss: 0.0017 - accu
```

In [16]:

```
new_weight = model.get_weights()[0]
```

In [17]:

```
new_weight
```

Out[17]:

```
array([[-0.16439508, -0.26342866,  0.2139388 ,  0.1693642 ,  0.2475452 ,
         0.01566632, -0.12896949,  0.10633375,  0.23772477, -0.28387713,
         0.11918657,  0.04305784, -0.13612962,  0.12070891, -0.2790749 ,
         0.12190154, -0.23822342, -0.17533383, -0.24046524, -0.00945631,
         0.06766909, -0.19942908,  0.07403941,  0.08044994, -0.01774754,
        -0.1256937 , -0.08698156,  0.09373008, -0.2490366 ,  0.06995557,
        -0.20825417,  0.18507683,  0.22288722, -0.2645494 , -0.17716205,
         0.00510343, -0.2109535 , -0.12479002, -0.14443372, -0.02225265,
         0.144975  ,  0.23313478, -0.05127065, -0.0155135 , -0.01173563,
        -0.16569744, -0.12573518,  0.12324277,  0.09528666, -0.13246311,
        -0.01731178, -0.21159074,  0.03941297,  0.18810247,  0.21458814,
         0.17994055,  0.1901029 , -0.04845009, -0.06115643, -0.16007149,
        -0.2220811 ,  0.01941903,  0.17042862, -0.09634206, -0.02289487,
        -0.0917784 , -0.2785842 , -0.2439856 ,  0.11533877,  0.18227403,
         0.08372505, -0.13307534, -0.20777065,  0.1502016 ,  0.01113592,
         0.14820726, -0.2568669 ,  0.22780967,  0.18116786, -0.01444541,
         0.02595313, -0.10240295, -0.19079238,  0.23621699, -0.10315399,
         0.1150419 ,  0.15855642,  0.18428838,  0.09193345, -0.1130795 ,
         0.1992723 ,  0.22749929,  0.09840029,  0.09724502, -0.18505886,
        -0.2656755 ,  0.06606409, -0.23646371,  0.16201983,  0.14630029,
         0.010037  , -0.16202953, -0.06923615,  0.25348726,  0.00590838,
        -0.18610533,  0.16728622, -0.06160085,  0.2066002 , -0.07396065,
         0.17516541, -0.2793147 ,  0.18125553,  0.10662021,  0.06860127,
        -0.17176785, -0.06460074,  0.11665172, -0.04703019,  0.19107226,
         0.07662158, -0.14046422,  0.17730774,  0.2322743 , -0.1701112 ,
         0.05935976, -0.24779443,  0.21072909],
       [ 0.14124638,  0.06497651,  0.09111632, -0.01335347,  0.17761256,
         0.1058484 ,  0.17762281, -0.12496661,  0.12869865,  0.02315781,
         0.16096625,  0.16704448,  0.05238903,  0.12223884, -0.04616054,
        -0.152746  ,  0.06599148,  0.1149858 ,  0.04999848,  0.1023223 ,
        -0.13079949,  0.23064741, -0.14915094,  0.02183983,  0.05126343,
         0.04935239,  0.20979425, -0.21604483,  0.06885181,  0.1958861 ,
         0.06712083,  0.19581395,  0.16842656, -0.06612989,  0.03950831,
         0.00834338,  0.21006463,  0.06280448,  0.19284512,  0.16495912,
        -0.01444886,  0.13768941,  0.18546665,  0.14832258, -0.02340749,
         0.093382  , -0.08969736, -0.1546097 , -0.07365023, -0.07455134,
         0.01263984,  0.1166325 ,  0.17389543,  0.05553646, -0.18394345,
         0.09854474,  0.10458752,  0.11898747,  0.18207581,  0.06096152,
        -0.0491097 ,  0.14210553,  0.05218474,  0.10471281,  0.16285762,
         0.23271447, -0.09531578, -0.13787287, -0.18829122,  0.06760187,
         0.07095378,  0.21179436,  0.12886278, -0.16774438,  0.08556917,
        -0.17257291, -0.1696529 , -0.0620657 ,  0.05149829,  0.17644633,
         0.13882849,  0.14745675, -0.11355498,  0.11976647, -0.14326257,
         0.20889068, -0.07100757, -0.09150648, -0.10089438, -0.05109631,
         0.12599961, -0.1082482 ,  0.07495589,  0.03957249,  0.14743683,
        -0.04796881, -0.08711113,  0.14475228, -0.12328976, -0.01534236,
         0.16573876,  0.05168033,  0.18792899,  0.04967576,  0.05605851,
         0.17273334,  0.07920987,  0.24174476, -0.17864834,  0.23656651,
         0.10815517, -0.01632296, -0.15461078,  0.24368604, -0.1183131 ,
         0.09546548, -0.44938156, -0.16130687, -0.3355594 ,  0.17031807,
        -0.02920767,  0.04310137,  0.0336033 , -0.09848002,  0.06903484,
         0.10443483,  0.05269535, -0.08268753]], dtype=float32)
```

In [18]:

```
model.optimizer.get_config()["learning_rate"]
```

Out[18]:

```
0.001
```

In [19]:

```
gradient = (old_weight - new_weight)/0.001
percent_change = abs(100*(old_weight-new_weight)/old_weight)
```

In [20]:

```
gradient
```

Out[20]:

```
array([[ 2.37293682e+01,  8.70255203e+01, -5.36314659e+01,
         1.47244329e+01, -3.54277191e+01, -5.09937286e+00,
        -2.44805050e+01, -1.95141716e+01, -2.95199146e+01,
         7.30042114e+01, -3.20049286e+01, -2.12099838e+01,
         4.24529533e+01, -8.31362381e+01,  6.80730591e+01,
        -1.49171200e+01,  7.02288437e+01,  4.26952515e+01,
         7.72288437e+01, -5.98770752e+01, -3.36740761e+01,
         4.99332657e+01,  4.74830246e+01, -2.73500214e+01,
        -1.76786747e+01,  8.32930756e+01, -2.72232647e+01,
        -2.86893311e+01,  1.10695732e+02,  8.00487328e+00,
         3.88571739e+01, -6.98692627e+01, -8.24837303e+00,
         5.15682526e+01,  8.27776718e+01, -8.39638996e+00,
         5.16822014e+01,  8.18221436e+01, -3.97960830e+00,
         9.20487881e+00, -1.00203896e+01, -9.44442825e+01,
        -6.15367203e+01, -1.89326057e+01,  1.73733692e+01,
         4.45336914e+01,  7.22934723e+01, -2.07320881e+01,
        -2.71861916e+01,  8.68414001e+01, -1.46752872e+01,
         5.59684296e+01, -1.10291374e+02, -4.53848228e+01,
        -2.28977051e+01, -2.09844551e+01, -2.65641060e+01,
         1.41698867e-01, -5.30041361e+00,  7.40746841e+01,
         6.18246155e+01, -3.66707325e+00, -1.67422886e+01,
         3.38865051e+01, -8.40353470e+01, -8.70993576e+01,
         7.12061615e+01,  6.56896515e+01, -3.45028763e+01,
        -7.68080902e+01, -6.99388962e+01,  5.15268707e+00,
         4.68433762e+01, -1.89150257e+01, -6.25416636e-01,
        -2.37185936e+01,  4.98548889e+01, -2.67969513e+01,
        -7.04751282e+01, -1.35187866e+02,  7.70525217e+00,
        -2.19957809e+01,  6.40951691e+01, -5.22768936e+01,
        -2.27609277e+00, -6.70833111e+00, -1.88784599e+01,
         4.80659294e+00, -2.45317879e+01,  9.71001816e+01,
        -7.48206663e+00, -3.25736694e+01, -6.54409561e+01,
        -3.20896721e+01,  2.68247719e+01,  6.02524261e+01,
        -2.35596142e+01,  3.31862411e+01, -2.39849076e-01,
         1.05369835e+01, -7.26966333e+00,  7.91987610e+01,
        -3.91595036e-01, -5.90389786e+01, -2.48440914e+01,
         2.39385204e+01, -1.26133847e+00, -1.06115334e+02,
        -1.37097235e+01, -5.65100136e+01, -4.37407036e+01,
         6.79984055e+01,  4.17846441e+00, -8.79092484e+01,
        -2.47571163e+01,  4.15972290e+01,  4.00519638e+01,
        -3.12722836e+01,  2.69574776e+01, -1.74459362e+01,
        -3.34630661e+01,  1.12998734e+02, -3.54124870e+01,
        -3.88295784e+01,  3.19747009e+01,  1.50804591e+00,
         8.02211456e+01, -2.12031441e+01],
       [-1.40341063e+01, -7.83105469e+01, -7.95115891e+01,
        -2.45110474e+01,  3.86628485e+00,  1.68779335e+01,
        -1.80807400e+01,  2.97080040e+01, -5.18254900e+00,
        -4.60915413e+01, -3.66486597e+00,  3.60854263e+01,
        -7.12924118e+01, -4.57214546e+01, -8.45931854e+01,
        -2.28168068e+01, -2.57095985e+01, -2.32979870e+01,
        -4.25455933e+01, -4.36230583e+01,  2.09457932e+01,
        -5.36759338e+01,  7.35617218e+01, -5.82737083e+01,
         6.78624039e+01, -1.22609222e+02, -5.04444656e+01,
         1.91703720e+01, -2.25844002e+01,  1.40289803e+01,
        -4.03444214e+01, -1.41581440e+01, -4.12482910e+01,
        -5.21808777e+01, -1.13251572e+02,  2.75266190e+01,
        -4.24000320e+01, -6.90704956e+01, -7.36278248e+00,
        -3.85429535e+01, -9.32729435e+00, -3.33206177e+01,
        -6.86557579e+00, -4.73138504e+01, -1.98993111e+01,
        -3.51964073e+01, -8.85320892e+01, -4.54696693e+01,
         9.11635876e+00, -3.85869265e+00,  3.02981682e+01,
        -2.17924042e+01, -8.70610123e+01, -7.27611542e+01,
         1.47054930e+01, -1.07678547e+01, -1.20003290e+01,
        -4.26994705e+01, -3.28416214e+01, -8.12648621e+01,
        -8.65952835e+01,  3.45826119e-01,  2.40630269e+00,
        -3.32626839e+01, -1.60017147e+01, -3.97056465e+01,
        -6.09577484e+01, -5.81147957e+01, -6.68483925e+00,
        -3.94729347e+01, -9.67813644e+01, -3.59823074e+01,
        -1.77975292e+01,  1.88682961e+00,  4.81902275e+01,
         2.80380225e+00, -4.44406090e+01, -1.18121243e+01,
        -2.38978481e+01, -2.02977943e+01,  1.45924530e+01,
        -2.84638386e+01, -4.68957329e+01,  2.91965890e+00,
        -8.12476826e+00,  3.95695853e+00,  8.30017757e+00,
         5.56258821e+00,  5.49752998e+00, -3.35694809e+01,
        -1.85692310e+01,  4.99970436e+00, -1.06635674e+02,
         3.08867073e+01, -1.09134760e+01, -4.70939789e+01,
         2.32633648e+01, -3.13673306e+01, -3.53812866e+01,
        -5.79400711e+01,  1.32585907e+01, -5.51219902e+01,
        -1.71487923e+01, -1.46488998e+02,  2.26246147e+01,
        -6.52568007e+00, -6.75247192e+01, -3.89974251e+01,
         2.03869934e+01, -5.69582405e+01,  3.49114876e+01,
        -3.71677094e+01,  1.16099417e+00, -1.42765091e+02,
         4.00127831e+01, -4.07561798e+01,  2.83737122e+02,
         8.98137665e+00,  2.13638809e+02,  3.00930576e+01,
        -1.79604912e+01, -1.01840652e+02,  1.98846149e+01,
         2.35503311e+01, -4.41634331e+01, -1.63086338e+01,
        -4.38924904e+01,  1.85823212e+01]], dtype=float32)
```

Early-Stooping

In [21]:

```
X,y = sklearn.datasets.make_circles(n_samples=100,noise=0.1,random_state=1)
```

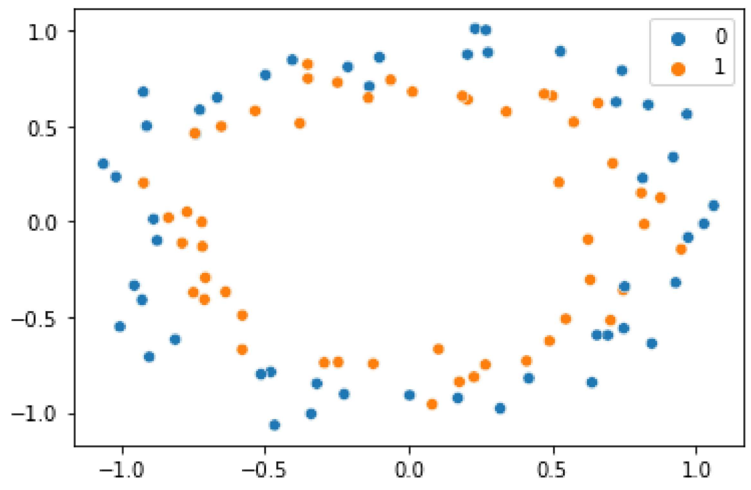
In [22]:

```
import seaborn as sns
sns.scatterplot(X[:,0],X[:,1],hue=y)
```

```
C:\Users\MSCIT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Future
Warning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[22]:

```
<AxesSubplot:>
```



In [23]:

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.20,random_state=2)
```

In [24]:

```
model = Sequential()
```

In [25]:

```
model.add(Dense(256,input_dim=2,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
```

In [26]:

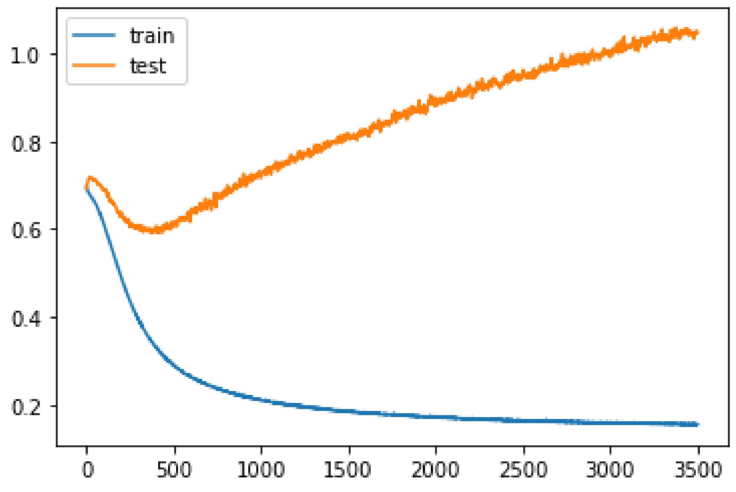
```
model.compile(loss='binary_crossentropy',optimizer='adam',metrics='accuracy')
```

In [27]:

```
history = model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=3500,verbose=0)
```

In [28]:

```
plt.plot(history.history['loss'],label='train')
plt.plot(history.history['val_loss'],label='test')
plt.legend()
plt.show()
```



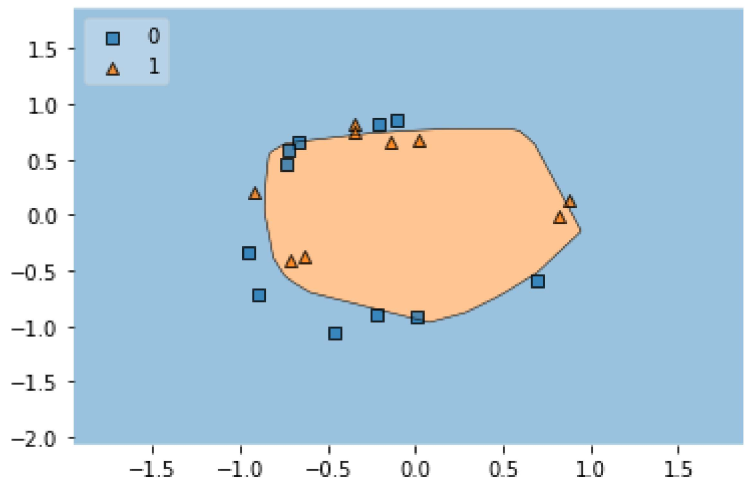
In [29]:

```
from mlxtend.plotting import plot_decision_regions
```

In [30]:

```
plot_decision_regions(X_test,y_test.ravel(),clf = model,legend=2)
plt.show()
```

```
3888/3888 [==============================] - 3s 747us/step
```



In [31]:

```
model = Sequential()
model.add(Dense(256,input_dim=2,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
```

In [32]:

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

In [33]:

```
from tensorflow.keras.callbacks import EarlyStopping
```

In [34]:

```
callback=EarlyStopping(
    monitor='val_loss',
    min_delta = 0.00001,
    patience=20,
    verbose=1,
    mode='auto',
    baseline=None,
    restore_best_weights=False)
```

In [41]:

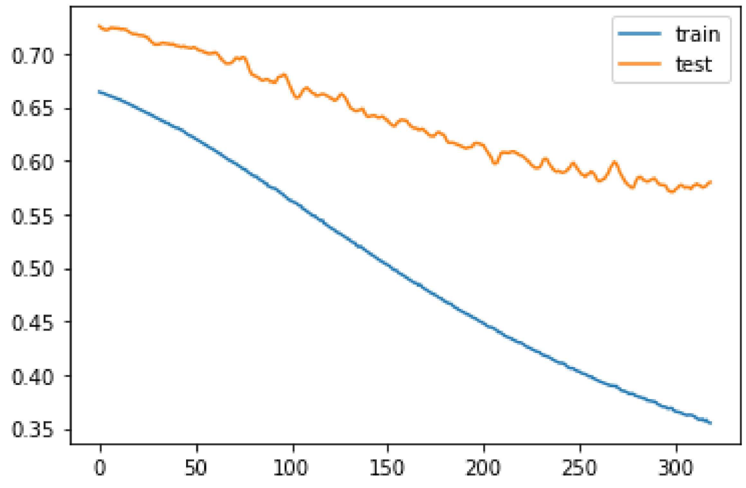
```
history = model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=3500,callbacks=c
```

```
3/3 [==============================] - 0s 11ms/step - loss: 0.3899 - accur
acy: 0.9000 - val_loss: 0.5958 - val_accuracy: 0.6500
Epoch 269/3500
3/3 [==============================] - 0s 11ms/step - loss: 0.3896 - accur
acy: 0.9000 - val_loss: 0.5989 - val_accuracy: 0.6500
Epoch 270/3500
3/3 [==============================] - 0s 12ms/step - loss: 0.3896 - accur
acy: 0.8750 - val_loss: 0.5977 - val_accuracy: 0.6500

Epoch 271/3500
3/3 [==============================] - 0s 10ms/step - loss: 0.3878 - accur
acy: 0.8750 - val_loss: 0.5939 - val_accuracy: 0.6500
Epoch 272/3500
3/3 [==============================] - 0s 10ms/step - loss: 0.3869 - accur
acy: 0.8625 - val_loss: 0.5890 - val_accuracy: 0.6500
Epoch 273/3500
3/3 [==============================] - 0s 10ms/step - loss: 0.3850 - accur
acy: 0.8875 - val_loss: 0.5858 - val_accuracy: 0.6500
Epoch 274/3500
3/3 [==============================] - 0s 10ms/step - loss: 0.3851 - accur
```

In [42]:

```
plt.plot(history.history['loss'],label='train')
plt.plot(history.history['val_loss'],label='test')
plt.legend()
plt.show()
```



In [ ]:

In [ ]: