In [1]:

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
# 讀取資料
from sklearn.datasets import load_breast_cancer
data = load_breast_cancer(return_X_y=False)
X, y = data.data, data.target
```

In [3]:

```
# 由以下資料可知 y=0代表惡性腫瘤，y=1代表良性腫瘤
data.target_names
```

Out[3]:

```
array(['malignant', 'benign'], dtype='<U9')
```

In [4]:

```
import pandas as pd
# 儲存模型
metrics = pd.DataFrame({'mean_accuracy':[],'mean_precision':[],'mean_recall':[]}).T
```

In [5]:

```
def sigmoid(x):
    return (1/(1+np.exp(-x)))
```

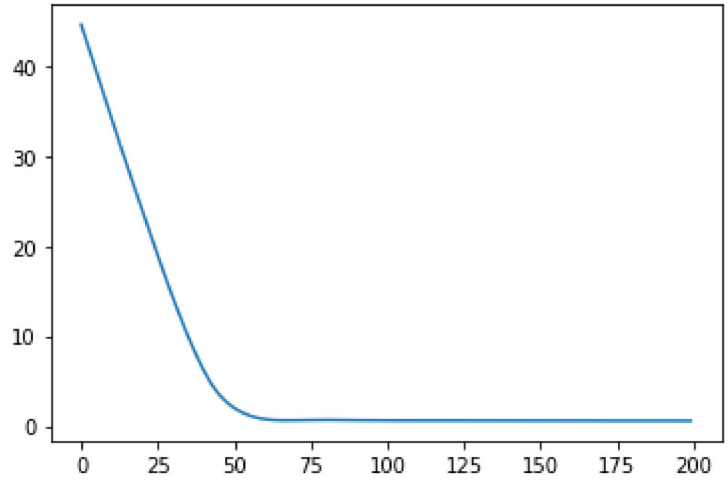
示範:利用tf.keras建立model

In [6]:

```
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras import regularizers
from sklearn.metrics import confusion_matrix


n, p = X.shape
model = tf.keras.Sequential(name='quadraticreg')
model.add(layers.Dense(1, activation='linear',
                       kernel_regularizer=regularizers.l2(0.01),
                       input_shape=(p,)))
model.summary()
model.compile(optimizer='adam', loss=tf.keras.losses.BinaryCrossentropy(from_logits=True))
history = model.fit(X, y, batch_size=n, epochs=200, verbose=0)
plt.plot(history.history['loss'])
plt.show()
```

```
Model: "quadraticreg"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 1)                 31
=================================================================
Total params: 31
Trainable params: 31
Non-trainable params: 0
_________________________________________________________________
```



Model1：一般線性模型

In [7]:

```
# 利用 RepeatedStratifiedKFold 驗證
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.metrics import confusion_matrix
```

In [8]:

```
#fit model
p = X.shape[1]
model = tf.keras.Sequential()
model.add(layers.Dense(1, activation='linear',
                       kernel_regularizer=regularizers.l2(0.01),
                       input_shape=(p,)))
model.compile(optimizer='adam',
                loss=tf.keras.losses.BinaryCrossentropy(from_logits=True))
```

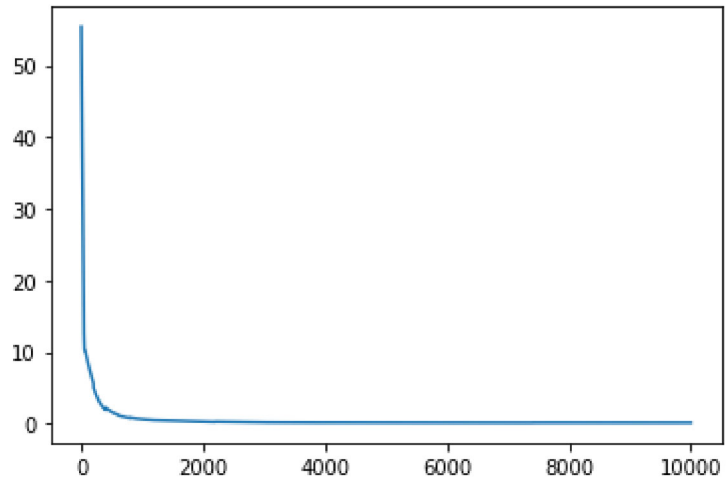
In [9]:

```
# repeat 10-Fold 5 次
accuracy = []
precision = []
recall = []
loss = []
rskf = RepeatedStratifiedKFold(n_splits=10, n_repeats = 5)
for train_index, test_index in rskf.split(X, y):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Fit model
    n = X_train.shape[0]
    history = model.fit(X_train, y_train, batch_size=n, epochs=200, verbose=0)
    #把loss存起來
    loss.extend(history.history['loss'])
    pred = (sigmoid(model.predict(X_test))>0.5).astype("int32")
    # test
    cm = confusion_matrix(pred,y_test)
    # 臨床實驗中 1 代表陽性，但此題因為0的位置代表惡性腫瘤，所以取 0 的位置當作陽性
    TP, FN = cm[0,0], cm[0,1]
    FP, TN = cm[1,0], cm[1,1]
    del cm
    accuracy.append((TP+TN)/X_test.shape[0])
    precision.append(TP/(TP+FP))
    recall.append(TP/(TP+FN))

print('mean accuracy:{:.2f}'.format(np.mean(accuracy)))
print('mean precision:{:.2f}'.format(np.mean(precision)))
print('mean recall:{:.2f}'.format(np.mean(recall)))
plt.plot(loss)
plt.show()
# 存下來
metrics['Model1']=[np.mean(accuracy),np.mean(precision),np.mean(recall)]
metrics
```

```
mean accuracy:0.92
mean precision:0.89
mean recall:0.91
```



Out[9]:

	Model1

	Model1
mean_accuracy	0.916698
mean_precision	0.887835
mean_recall	0.905221

Model2 : Normalize

In [10]:

```
from sklearn.preprocessing import normalize
norm_X = normalize(X)
```

In [11]:

```
#fit model
p = norm_X.shape[1]
model = tf.keras.Sequential()
model.add(layers.Dense(1, activation='linear',
        kernel_regularizer=regularizers.l2(0.01),
        input_shape=(p,)))
model.compile(optimizer='adam',
                loss=tf.keras.losses.BinaryCrossentropy(from_logits=True))
```

In [12]:

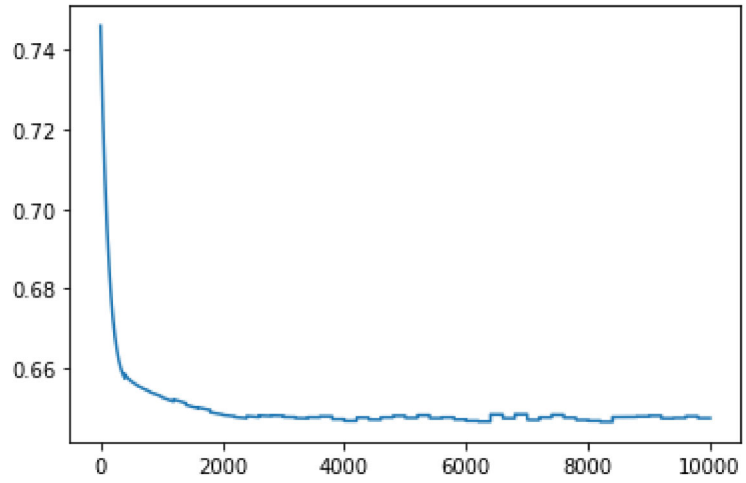
```
# repeat 10-Fold 5 次
accuracy = []
precision = []
recall = []
loss = []
rskf = RepeatedStratifiedKFold(n_splits=10, n_repeats = 5)
for train_index, test_index in rskf.split(norm_X, y):
    X_train, X_test = norm_X[train_index], norm_X[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Fit model
    n = X_train.shape[0]
    history = model.fit(X_train, y_train, batch_size=n, epochs=200, verbose=0)
    #把loss存起來
    loss.extend(history.history['loss'])
    pred = (sigmoid(model.predict(X_test))>0.5).astype("int32")
    # test
    cm = confusion_matrix(pred,y_test)

    # 臨床實驗中 1 代表陽性，但此題因為0的位置代表惡性腫瘤，所以取 0 的位置當作陽性
    TP, FN = cm[0,0], cm[0,1]
    FP, TN = cm[1,0], cm[1,1]
    del cm
    accuracy.append((TP+TN)/X_test.shape[0])
    precision.append(TP/(TP+FP))
    recall.append(TP/(TP+FN))

print('mean accuracy:{:.2f}'.format(np.mean(accuracy)))
print('mean precision:{:.2f}'.format(np.mean(precision)))
print('mean recall:{:.2f}'.format(np.mean(recall)))
plt.plot(loss)
plt.show()
# 存下來
metrics['Model2']=[np.mean(accuracy),np.mean(precision),np.mean(recall)]
metrics
```

```
mean accuracy:0.63
mean precision:0.00
mean recall:nan
```



Out[12]:

	Model1	Model2
mean_accuracy	0.916698	0.627412
mean_precision	0.887835	0.000000
mean_recall	0.905221	NaN

Model3 : Standardize

In [13]:

```
from sklearn.preprocessing import StandardScaler
std = StandardScaler()
std_X = std.fit_transform(X)
```

In [14]:

```
#fit model
p = std_X.shape[1]
model = tf.keras.Sequential()
model.add(layers.Dense(1, activation='linear',
        kernel_regularizer=regularizers.l2(0.01),
        input_shape=(p,)))
model.compile(optimizer='adam',
               loss=tf.keras.losses.BinaryCrossentropy(from_logits=True))
```

In [15]:

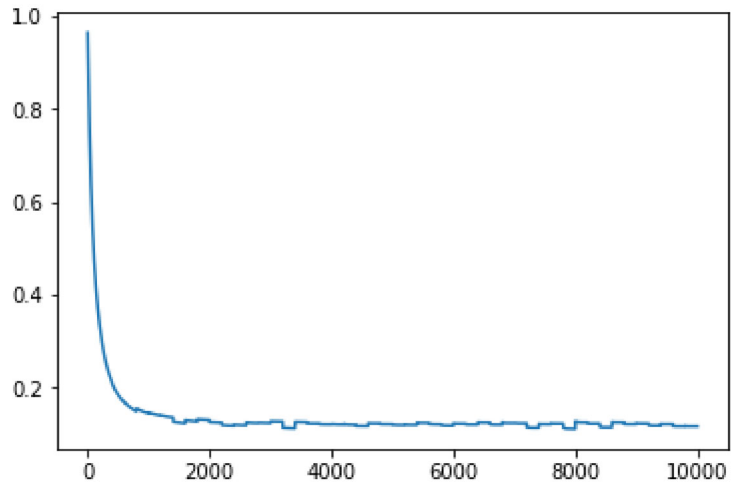
```
# repeat 10-Fold 5 次
accuracy = []
precision = []
recall = []
loss = []
rskf = RepeatedStratifiedKFold(n_splits=10, n_repeats = 5)
for train_index, test_index in rskf.split(std_X, y):
    X_train, X_test = std_X[train_index], std_X[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Fit model
    n = X_train.shape[0]
    history = model.fit(X_train, y_train, batch_size=n, epochs=200, verbose=0)
    #把loss存起來
    loss.extend(history.history['loss'])
    pred = (sigmoid(model.predict(X_test))>0.5).astype("int32")
    # test
    cm = confusion_matrix(pred,y_test)

    # 臨床實驗中 1 代表陽性，但此題因為0的位置代表惡性腫瘤，所以取 0 的位置當作陽性
    TP, FN = cm[0,0], cm[0,1]
    FP, TN = cm[1,0], cm[1,1]
    del cm
    accuracy.append((TP+TN)/X_test.shape[0])
    precision.append(TP/(TP+FP))
    recall.append(TP/(TP+FN))

print('mean accuracy:{:.2f}'.format(np.mean(accuracy)))
print('mean precision:{:.2f}'.format(np.mean(precision)))
print('mean recall:{:.2f}'.format(np.mean(recall)))
plt.plot(loss)
plt.show()
# 存下來
metrics['Model3']=[np.mean(accuracy),np.mean(precision),np.mean(recall)]
metrics
```

```
mean accuracy:0.98
mean precision:0.95
mean recall:0.99
```



Out[15]:

	Model1	Model2	Model3
mean_accuracy	0.916698	0.627412	0.976441
mean_precision	0.887835	0.000000	0.949048
mean_recall	0.905221	NaN	0.988054

Model4：取 $log(x)$ 來嘗試

In [16]:

```
log_X = np.log(X+1e-6) #因為log(0)會出錯，所以加入極小值使資料恆正
log_X.shape
```

Out[16]:

```
(569, 30)
```

In [17]:

```
#fit model
p = log_X.shape[1]
model = tf.keras.Sequential()
model.add(layers.Dense(1, activation='linear',
        kernel_regularizer=regularizers.l2(0.01),
        input_shape=(p,)))
model.compile(optimizer='adam',
              loss=tf.keras.losses.BinaryCrossentropy(from_logits=True))
```

In [18]:

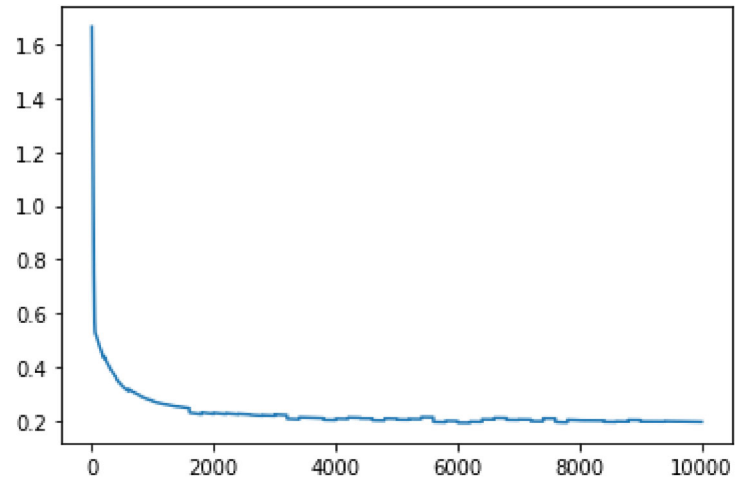
```
# repeat 10-Fold 5 次
accuracy = []
precision = []
recall = []
loss = []
rskf = RepeatedStratifiedKFold(n_splits=10, n_repeats = 5)
for train_index, test_index in rskf.split(log_X, y):
    X_train, X_test = log_X[train_index], log_X[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Fit model
    n = X_train.shape[0]
    history = model.fit(X_train, y_train, batch_size=n, epochs=200, verbose=0)
    #把loss存起來
    loss.extend(history.history['loss'])
    pred = (sigmoid(model.predict(X_test))>0.5).astype("int32")
    # test
    cm = confusion_matrix(pred,y_test)
    # 劃出confusion matrix(因為篇幅關係沒有畫出來，避免同學看得混亂)
    # sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', annot_kws={'size':50})
    # plt.title('confusion matrix')
    # plt.ylabel('Actual')
    # plt.xlabel('Predicted')
    # plt.show()

    # 臨床實驗中 1 代表陽性，但此題因為0的位置代表惡性腫瘤，所以取 0 的位置當作陽性
    TP, FN = cm[0,0], cm[0,1]
    FP, TN = cm[1,0], cm[1,1]
    del cm
    accuracy.append((TP+TN)/X_test.shape[0])
    precision.append(TP/(TP+FP))
    recall.append(TP/(TP+FN))

print('mean accuracy:{:.2f}'.format(np.mean(accuracy)))
print('mean precision:{:.2f}'.format(np.mean(precision)))
print('mean recall:{:.2f}'.format(np.mean(recall)))
plt.plot(loss)
plt.show()
# 存下來
metrics['Model4']=[np.mean(accuracy),np.mean(precision),np.mean(recall)]
metrics
```

```
mean accuracy:0.94
mean precision:0.90
mean recall:0.95
```



Out[18]:

	Model1	Model2	Model3	Model4
mean_accuracy	0.916698	0.627412	0.976441	0.943026
mean_precision	0.887835	0.000000	0.949048	0.896667
mean_recall	0.905221	NaN	0.988054	0.949176

結果

1.單從accuracy可看出model3(standardize)結果很好。

2.Model2的recall雖然很高，但precision不好，整體而言不是很好的模型。

3.Model3的各項指數皆高於Model1與Model4，所以我選擇使用Model3(先Standardize data,然後利用LogisticRegression去fit模型)

In [ ]: