```python
In [1]: import numpy as np
        import pandas as pd
        import sklearn
        import matplotlib.pyplot as plt
        import seaborn as sns
        import scipy
        import keras
        import tensorflow as tf
        from keras.utils import to_categorical
```

```python
In [2]: data = pd.read_csv("breast-cancer.data",header=None)
        data.columns = ['Class','age','menopause','tumor-size','inv-nodes','node-caps','d
```

```python
In [3]: data
```

Out[3]:

	Class	age	menopause	tumor-size	inv-nodes	node-caps	deg-malig	breast	breast-quad	irradiat
0	no-recurrence-events	30-39	premeno	30-34	0-2	no	3	left	left_low	no
1	no-recurrence-events	40-49	premeno	20-24	0-2	no	2	right	right_up	no
2	no-recurrence-events	40-49	premeno	20-24	0-2	no	2	left	left_low	no
3	no-recurrence-events	60-69	ge40	15-19	0-2	no	2	right	left_up	no
4	no-recurrence-events	40-49	premeno	0-4	0-2	no	2	right	right_low	no
...	...	...	...	...	...	...	...	...	...	...
281	recurrence-events	30-39	premeno	30-34	0-2	no	2	left	left_up	no
282	recurrence-events	30-39	premeno	20-24	0-2	no	3	left	left_up	yes
283	recurrence-events	60-69	ge40	20-24	0-2	no	1	right	left_up	no
284	recurrence-events	40-49	ge40	30-34	3-5	no	3	left	left_low	no
285	recurrence-events	50-59	ge40	30-34	3-5	no	3	left	left_low	no

286 rows × 10 columns

```python
In [4]: from sklearn.preprocessing import LabelEncoder
        le = LabelEncoder()
        for i in data:
            data[i] = le.fit_transform(data[i])
```

```
In [5]: x= data.drop("irradiat",axis=1)
        y = data["irradiat"]
```

```
In [6]: x
```

Out[6]:

	Class	age	menopause	tumor-size	inv-nodes	node-caps	deg-malig	breast	breast-quad
0	0	1	2	5	0	1	2	0	2
1	0	2	2	3	0	1	1	1	5
2	0	2	2	3	0	1	1	0	2
3	0	4	0	2	0	1	1	1	3
4	0	2	2	0	0	1	1	1	4
...	...	...	...	...	...	...	...	...	...
281	1	1	2	5	0	1	1	0	3
282	1	1	2	3	0	1	2	0	3
283	1	4	0	3	0	1	0	1	3
284	1	2	0	5	4	1	2	0	2
285	1	3	0	5	4	1	2	0	2

286 rows × 9 columns

```
In [7]: y
```

```
Out[7]: 0      0
        1      0
        2      0
        3      0
        4      0
              ..
        281    0
        282    1
        283    0
        284    0
        285    0
        Name: irradiat, Length: 286, dtype: int32
```

```
In [8]: x.shape
```

Out[8]: (286, 9)

```
In [9]: y.shape
```

Out[9]: (286,)

```
In [10]: from sklearn.model_selection import train_test_split
         xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size=0.25,random_state=0)
```

```
In [11]: xtrain1 = np.array(xtrain)
         xtest1 = np.array(xtest)
         ytrain1 = np.array(ytrain)
         ytest1 = np.array(ytest)
```

```
In [12]: xtrain1 = xtrain1.reshape(xtrain1.shape[0],xtrain1.shape[1],1)
         xtest1 = xtest1.reshape(xtest1.shape[0],xtest1.shape[1],1)
```

```
In [13]: xtrain1.shape
```

Out[13]: (214, 9, 1)

```
In [14]: xtest1.shape
```

Out[14]: (72, 9, 1)

```
In [15]: ytrain1.shape
```

Out[15]: (214,)

```
In [16]: ytest1.shape
```

Out[16]: (72,)

```
In [17]: from keras.layers import LSTM,Dense,Activation,Flatten
         from keras.models import Sequential
```

```
In [18]: model = Sequential()
         model.add(LSTM(100,input_shape=(xtrain1.shape[1],1)))
         model.add(Dense(10))
         model.add(Dense(8))
         model.add(Dense(1))
         model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 100)	40800
dense (Dense)	(None, 10)	1010
dense_1 (Dense)	(None, 8)	88
dense_2 (Dense)	(None, 1)	9

Total params: 41,907
Trainable params: 41,907
Non-trainable params: 0

```
In [19]:  model.compile(metrics= ["accuracy"],optimizer="adam",loss="categorical_crossentro
```

```
In [20]:  history = model.fit(xtrain1, ytrain1,batch_size=10,epochs=250)
```

```
Epoch 1/250
22/22 [==============================] - 0s 9ms/step - loss: 2.8410e-08 - acc
uracy: 0.7617
Epoch 2/250
22/22 [==============================] - 0s 7ms/step - loss: 2.8410e-08 - acc
uracy: 0.7617
Epoch 3/250
22/22 [==============================] - 0s 6ms/step - loss: 2.8410e-08 - acc
uracy: 0.7617
Epoch 4/250
22/22 [==============================] - 0s 7ms/step - loss: 2.8410e-08 - acc
uracy: 0.7617
Epoch 5/250
22/22 [==============================] - 0s 7ms/step - loss: 2.8410e-08 - acc
uracy: 0.7617
Epoch 6/250
22/22 [==============================] - 0s 6ms/step - loss: 2.8410e-08 - acc
uracy: 0.7617
Epoch 7/250
22/22 [                              ] - 0s 6ms/step - loss: 2.8410e-08 - acc
```

```
In [21]:  history = pd.DataFrame(history.history)
          history
```
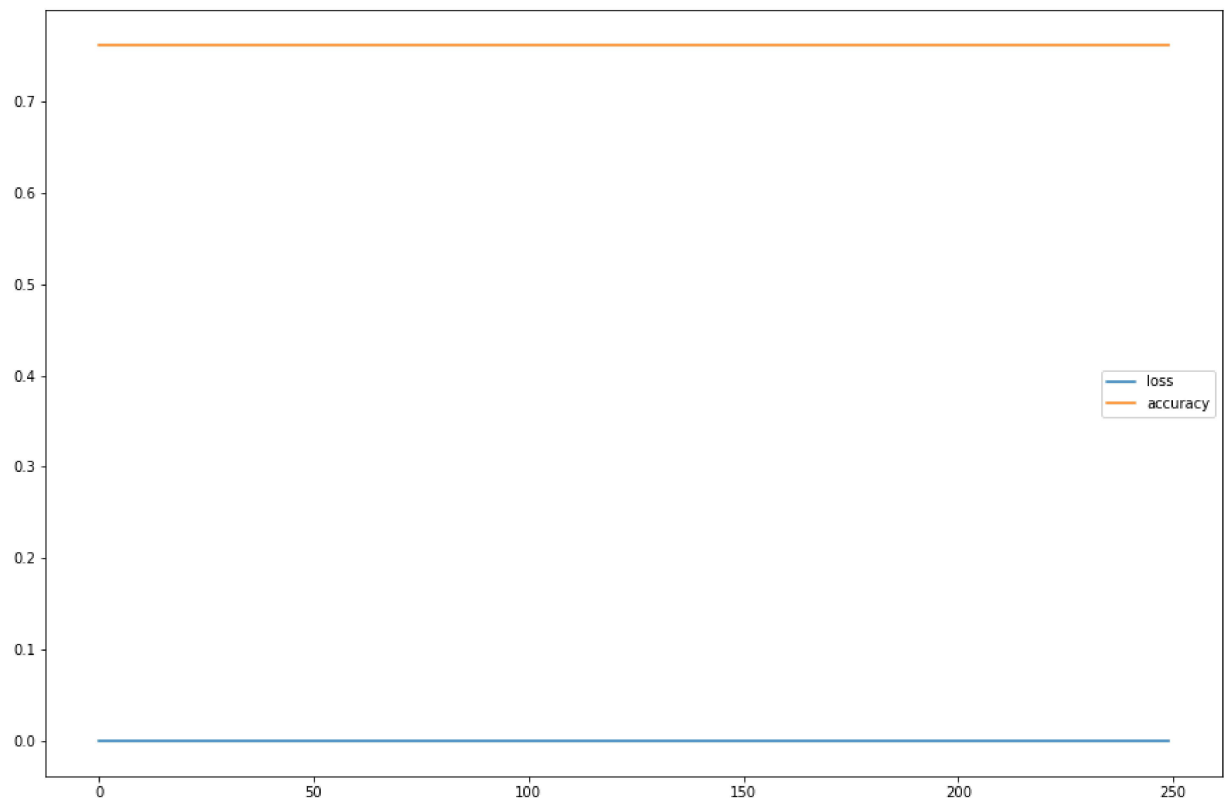
Out[21]:

	loss	accuracy
0	2.840969e-08	0.761682
1	2.840969e-08	0.761682
2	2.840969e-08	0.761682
3	2.840969e-08	0.761682
4	2.840969e-08	0.761682
...	...	...
245	2.840969e-08	0.761682
246	2.840969e-08	0.761682
247	2.840969e-08	0.761682
248	2.840969e-08	0.761682
249	2.840969e-08	0.761682

250 rows × 2 columns

```
In [22]:  history.plot(figsize=(15,10))
```

Out[22]: <AxesSubplot:>


```
In [23]:  ypredict = np.argmax(model.predict(xtest1), axis=-1)
```

```
In [24]:  from sklearn.metrics import accuracy_score
          accuracy_score(ytest1,ypredict)
```

Out[24]: 0.7638888888888888