Bahria University,

Karachi Campus

LAB EXPERIMENT NO.

13

LIST OF TASKS

TASK NO	OBJECTIVE
1	Train the Artificial Neural Network using the dataset provided in the lab.
2	Train the Artificial Neural Network using the iris dataset and discuss the results.

Submitted On:

____10/01/2023_____

(Date: DD/MM/YY)

Task# 01: - Train the Artificial Neural Network using the dataset provided in the lab.

Solution: -

```
from sklearn.model_selection import train_test_split # Import train_test_split
function
from sklearn import metrics #Import scikit-learn metrics module for accuracy
calculation
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from sklearn.preprocessing import scale
from keras.utils import np_utils
from sklearn.metrics import accuracy_score
df =
pd.read_csv('https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians
-diabetes.csv', header=None)
X = df.drop(8, axis = 1)
y = df[8]
X = scale(X)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
random_state=2)
y_train = np_utils.to_categorical(y_train)
model = Sequential()
model.add(Dense(12, input_dim=8))
model.add(Dense(8, activation='relu'))
model.add(Dense(2, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(X_train, y_train, epochs=150, batch_size=10)
y_pred = model.predict(X_test)
y_pred = np.argmax(y_pred, axis=1)
accuracy_score(y_test, y_pred)
```

Output: -

```
Epoch 1/150
52/52 [==============================] - 1s 3ms/step - loss: 0.6736 - accuracy: 0.7062
Epoch 2/150
52/52 [==============================] - 0s 1ms/step - loss: 0.6103 - accuracy: 0.7315
Epoch 3/150
52/52 [==============================] - 0s 1ms/step - loss: 0.5644 - accuracy: 0.7451
Epoch 4/150
52/52 [==============================] - 0s 1ms/step - loss: 0.5314 - accuracy: 0.7588
Epoch 5/150
52/52 [==============================] - 0s 1ms/step - loss: 0.5091 - accuracy: 0.7646
Epoch 6/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4962 - accuracy: 0.7607
Epoch 7/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4881 - accuracy: 0.7685
Epoch 8/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4820 - accuracy: 0.7685
Epoch 9/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4779 - accuracy: 0.7588
Epoch 10/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4741 - accuracy: 0.7743
Epoch 11/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4711 - accuracy: 0.7763
Epoch 12/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4695 - accuracy: 0.7763
Epoch 13/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4679 - accuracy: 0.7821
Epoch 14/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4660 - accuracy: 0.7821
Epoch 15/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4645 - accuracy: 0.7860
Epoch 16/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4638 - accuracy: 0.7821
Epoch 17/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4629 - accuracy: 0.7802
Epoch 18/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4606 - accuracy: 0.7899
Epoch 19/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4602 - accuracy: 0.7860
Epoch 20/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4583 - accuracy: 0.7879
Epoch 21/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4580 - accuracy: 0.7860
Epoch 22/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4568 - accuracy: 0.7840
Epoch 23/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4563 - accuracy: 0.7840
Epoch 24/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4549 - accuracy: 0.7782
Epoch 25/150
52/52 [==============================] - 0s 2ms/step - loss: 0.4547 - accuracy: 0.7821
Epoch 26/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4539 - accuracy: 0.7821
Epoch 27/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4540 - accuracy: 0.7840
Epoch 28/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4525 - accuracy: 0.7821
Epoch 29/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4518 - accuracy: 0.7840
```

```
Epoch 123/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4270 - accuracy: 0.7938
Epoch 124/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4267 - accuracy: 0.7977
Epoch 125/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4273 - accuracy: 0.7977
Epoch 126/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4260 - accuracy: 0.7996
Epoch 127/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4271 - accuracy: 0.7996
Epoch 128/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4266 - accuracy: 0.7899
Epoch 129/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4280 - accuracy: 0.7957
Epoch 130/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4257 - accuracy: 0.7957
Epoch 131/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4270 - accuracy: 0.7938
Epoch 132/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4262 - accuracy: 0.7977
Epoch 133/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4258 - accuracy: 0.7957
Epoch 134/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4278 - accuracy: 0.8016
Epoch 135/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4257 - accuracy: 0.7957
Epoch 136/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4263 - accuracy: 0.8016
Epoch 137/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4257 - accuracy: 0.8016
Epoch 138/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4262 - accuracy: 0.7977
Epoch 139/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4255 - accuracy: 0.7977
Epoch 140/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4263 - accuracy: 0.7996
Epoch 141/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4269 - accuracy: 0.7957
Epoch 142/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4255 - accuracy: 0.7938
Epoch 143/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4250 - accuracy: 0.7996
Epoch 144/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4250 - accuracy: 0.7977
Epoch 145/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4250 - accuracy: 0.7938
Epoch 146/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4252 - accuracy: 0.7938
Epoch 147/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4258 - accuracy: 0.7957
Epoch 148/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4246 - accuracy: 0.7977
Epoch 149/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4250 - accuracy: 0.7996
Epoch 150/150
52/52 [==============================] - 0s 1ms/step - loss: 0.4249 - accuracy: 0.8035
8/8 [==============================] - 0s 2ms/step
0.7519685039370079
```

Task 02: - Train the Artificial Neural Network using the iris dataset and discuss the results.

Solution: -

```
from sklearn.model_selection import train_test_split
from sklearn import metrics
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from sklearn import datasets
from keras.utils import np_utils
```

```
from sklearn.preprocessing import scale
from sklearn.metrics import accuracy_score
Iris = datasets.load_iris()
Iris_df = pd.DataFrame(Iris.data)
X = Iris_df
y = Iris.target
X = scale(X)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
random_state=2)
y_train = np_utils.to_categorical(y_train)
model = Sequential()
model.add(Dense(12, input_dim=4))
model.add(Dense(8, activation='relu'))
model.add(Dense(3, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(X_train, y_train, epochs=250, batch_size=10)
y_pred = model.predict(X_test)
y_pred = np.argmax(y_pred, axis=1)
accuracy_score(y_test, y_pred)
```

Output: -

```
Epoch 1/250
10/10 [==============================] - 1s 2ms/step - loss: 0.6684 - accuracy: 0.3000
Epoch 2/250
10/10 [==============================] - 0s 2ms/step - loss: 0.6392 - accuracy: 0.3400
Epoch 3/250
10/10 [==============================] - 0s 2ms/step - loss: 0.6116 - accuracy: 0.3300
Epoch 4/250
10/10 [==============================] - 0s 2ms/step - loss: 0.5869 - accuracy: 0.3600
Epoch 5/250
10/10 [==============================] - 0s 2ms/step - loss: 0.5634 - accuracy: 0.5100
Epoch 6/250
10/10 [==============================] - 0s 2ms/step - loss: 0.5420 - accuracy: 0.6300
Epoch 7/250
10/10 [==============================] - 0s 2ms/step - loss: 0.5236 - accuracy: 0.6600
Epoch 8/250
10/10 [==============================] - 0s 2ms/step - loss: 0.5056 - accuracy: 0.6600
Epoch 9/250
10/10 [==============================] - 0s 1ms/step - loss: 0.4900 - accuracy: 0.6600
Epoch 10/250
10/10 [==============================] - 0s 2ms/step - loss: 0.4747 - accuracy: 0.6600
Epoch 11/250
10/10 [==============================] - 0s 1ms/step - loss: 0.4613 - accuracy: 0.6600
Epoch 12/250
10/10 [==============================] - 0s 2ms/step - loss: 0.4472 - accuracy: 0.6600
Epoch 13/250
10/10 [==============================] - 0s 2ms/step - loss: 0.4344 - accuracy: 0.6600
Epoch 14/250
10/10 [==============================] - 0s 1ms/step - loss: 0.4224 - accuracy: 0.6600
Epoch 15/250
10/10 [==============================] - 0s 2ms/step - loss: 0.4106 - accuracy: 0.6700
Epoch 16/250
10/10 [==============================] - 0s 2ms/step - loss: 0.3995 - accuracy: 0.6800
Epoch 17/250
10/10 [==============================] - 0s 1ms/step - loss: 0.3892 - accuracy: 0.6800
Epoch 18/250
10/10 [==============================] - 0s 1ms/step - loss: 0.3795 - accuracy: 0.6800
Epoch 19/250
10/10 [==============================] - 0s 1ms/step - loss: 0.3704 - accuracy: 0.6800
Epoch 20/250
10/10 [==============================] - 0s 1ms/step - loss: 0.3618 - accuracy: 0.6800
Epoch 21/250
10/10 [==============================] - 0s 2ms/step - loss: 0.3545 - accuracy: 0.6800
Epoch 22/250
10/10 [==============================] - 0s 2ms/step - loss: 0.3465 - accuracy: 0.6800
Epoch 23/250
10/10 [==============================] - 0s 2ms/step - loss: 0.3400 - accuracy: 0.6900
Epoch 24/250
10/10 [==============================] - 0s 2ms/step - loss: 0.3336 - accuracy: 0.6900
Epoch 25/250
10/10 [==============================] - 0s 1ms/step - loss: 0.3269 - accuracy: 0.7000
Epoch 26/250
10/10 [==============================] - 0s 1ms/step - loss: 0.3210 - accuracy: 0.7000
Epoch 27/250
10/10 [==============================] - 0s 1ms/step - loss: 0.3153 - accuracy: 0.7000
Epoch 28/250
10/10 [==============================] - 0s 1ms/step - loss: 0.3094 - accuracy: 0.7300
Epoch 29/250
10/10 [==============================] - 0s 1ms/step - loss: 0.3037 - accuracy: 0.7400
2/2 [==============================] - 0s 6ms/step
0.98
```

Discussion: -

Here, Our model's training accuracy is 97% while test accuracy is 98%. We can still increase our accuracy by increasing the layers or nodes in a model or preprocessing the data.