Name : Annapoornima S

Roll No. : 225229101

Lab : 5 : Text corpus creation and binary classification using DNN¶

Dataset :

In [6]:

```
import nltk
import pandas as pd
```

In [7]:

```
from nltk.corpus import stopwords
from sklearn.model_selection import train_test_split
from nltk.stem import WordNetLemmatizer
```

In [8]:

```
import warnings
warnings.filterwarnings("ignore", category=YourWarningCategory, action="once")
```

In [9]:

```
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
nltk.download('wordnet')
nltk.download('omw-1.4')
```

In [10]:

```
df=pd.read_csv("Motive - Sheet1.csv",encoding='cp1252')
```

In [11]:

```
df.shape
```

In [12]:

```
df.head()
```

In [13]:

```
df.groupby('Label').count()
```

Pre-processing:

In [8]:

```
X=df.Statement
y=df.Label
```

In [9]:

```
lemmatizer=WordNetLemmatizer()
```

In [10]:

```
def clean_review(review):

    tokens = review.lower().split()
    filtered_tokens = [lemmatizer.lemmatize(w) for w in tokens if w not in stop_words]
    return " ".join(filtered_tokens)
```

In [11]:

```
temp=X.tolist()
fax=[]
for i in temp:
    fax.append(clean_review(i))
n_X=pd.Series(fax)
```

In [12]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
import pandas as pd

tfidf = TfidfVectorizer()
vectors = tfidf.fit_transform(n_X)
features_names = tfidf.get_feature_names_out()
text_vect = pd.DataFrame(vectors.todense(), columns=features_names)
text_vect
```

Out[12]:

	abilities	achieve	afraid	anything	believe	bother	capable	challenge	change	come
0	0.000000	0.550965	0.000000	0.550965	0.626798	0.000000	0.00000	0.00000	0.000000	0.000000
1	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.00000	0.000000	0.000000
2	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.00000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.00000	0.000000	0.519707
4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.00000	0.000000	0.000000
5	0.000000	0.000000	0.421482	0.000000	0.000000	0.000000	0.00000	0.00000	0.000000	0.000000
6	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.57735	0.57735	0.000000	0.000000
7	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.00000	0.000000	0.000000
8	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.00000	0.000000	0.000000

In [13]:

```
import tensorflow as tf
temp = tf.Variable(text_vect)
```

In [14]:

```
X_train,X_test,y_train,y_test=train_test_split(text_vect,y,train_size=0.75,test_size=0.2
```

In [15]:

```
print(X_train.shape)
```

```
(15, 62)
```

In [16]:

```
print(y_train.shape)
```

```
(15,)
```

In [17]:

```
print(X_test.shape)
```

```
(5, 62)
```

In [18]:

```
print(y_test.shape)
```

```
(5,)
```

Model Build

In [19]:

```
import tensorflow as tf
from tensorflow.keras import Sequential
from keras.layers import Dense,Activation
```

Model 1 : Hidden Layer : 8

In [20]:

```
model1 = Sequential()
model1.add(Dense(8, activation='relu',input_dim=X_train.shape[1]))
model1.add(Dense(2, activation='sigmoid'))
model1.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 8)                 504

 dense_1 (Dense)             (None, 2)                 18

=================================================================
Total params: 522 (2.04 KB)
Trainable params: 522 (2.04 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [27]:

```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
y_train = label_encoder.fit_transform(y_train)
y_test = label_encoder.fit_transform(y_test)
```

In [28]:

```
model1.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accurac
history=model1.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=
```

```
Epoch 54/100
1/1 - 0s - loss: 0.2691 - accuracy: 1.0000 - val_loss: 0.9616 - val_ac
curacy: 0.3333 - 55ms/epoch - 55ms/step
Epoch 55/100
1/1 - 0s - loss: 0.2671 - accuracy: 1.0000 - val_loss: 0.9638 - val_ac
curacy: 0.3333 - 58ms/epoch - 58ms/step
Epoch 56/100
1/1 - 0s - loss: 0.2651 - accuracy: 1.0000 - val_loss: 0.9660 - val_ac
curacy: 0.3333 - 56ms/epoch - 56ms/step
Epoch 57/100
1/1 - 0s - loss: 0.2630 - accuracy: 1.0000 - val_loss: 0.9682 - val_ac
curacy: 0.3333 - 60ms/epoch - 60ms/step
Epoch 58/100
1/1 - 0s - loss: 0.2611 - accuracy: 1.0000 - val_loss: 0.9705 - val_ac
curacy: 0.3333 - 55ms/epoch - 55ms/step
Epoch 59/100
1/1 - 0s - loss: 0.2591 - accuracy: 1.0000 - val_loss: 0.9728 - val_ac
curacy: 0.3333 - 78ms/epoch - 78ms/step
Epoch 60/100
1/1 - 0s - loss: 0.2571 - accuracy: 1.0000 - val_loss: 0.9752 - val_ac
```

In [29]:

```
model1.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 50ms/step - loss: 1.2599 - accu
racy: 0.2000
```

Out[29]:

```
[1.2599141597747803, 0.20000000298023224]
```

Model 2 : Hidden Layer : 16

In [30]:

```
model2 = Sequential()
model2.add(Dense(16, activation='relu',input_dim=X_train.shape[1]))
model2.add(Dense(8, activation='relu'))
model2.add(Dense(2, activation='sigmoid'))
model2.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_2 (Dense)             (None, 16)                1008

 dense_3 (Dense)             (None, 8)                 136

 dense_4 (Dense)             (None, 2)                 18

=================================================================
Total params: 1162 (4.54 KB)
Trainable params: 1162 (4.54 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [31]:

```
model2.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accurac
history=model2.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=
```

```
Epoch 1/100
1/1 - 1s - loss: 0.7032 - accuracy: 0.5000 - val_loss: 0.6688 - val_ac
curacy: 0.6667 - 1s/epoch - 1s/step
Epoch 2/100
1/1 - 0s - loss: 0.6996 - accuracy: 0.5000 - val_loss: 0.6696 - val_ac
curacy: 0.6667 - 76ms/epoch - 76ms/step
Epoch 3/100
1/1 - 0s - loss: 0.6961 - accuracy: 0.5000 - val_loss: 0.6702 - val_ac
curacy: 0.6667 - 54ms/epoch - 54ms/step
Epoch 4/100
1/1 - 0s - loss: 0.6926 - accuracy: 0.5000 - val_loss: 0.6709 - val_ac
curacy: 0.6667 - 56ms/epoch - 56ms/step
Epoch 5/100
1/1 - 0s - loss: 0.6892 - accuracy: 0.5833 - val_loss: 0.6715 - val_ac
curacy: 0.6667 - 60ms/epoch - 60ms/step
Epoch 6/100
1/1 - 0s - loss: 0.6857 - accuracy: 0.5833 - val_loss: 0.6721 - val_ac
curacy: 0.6667 - 54ms/epoch - 54ms/step
Epoch 7/100
```

In [32]:

```
model2.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 55ms/step - loss: 0.9272 - accu
racy: 0.2000
```

Out[32]:

```
[0.9271775484085083, 0.20000000298023224]
```

Model 3 : Hidden Layer : 32

In [33]:

```
model3 = Sequential()
model3.add(Dense(32, activation='relu',input_dim=X_train.shape[1]))
model3.add(Dense(16, activation='relu'))
model3.add(Dense(8, activation='relu'))
model3.add(Dense(2, activation='sigmoid'))
model3.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_5 (Dense)             (None, 32)                2016

 dense_6 (Dense)             (None, 16)                528

 dense_7 (Dense)             (None, 8)                 136

 dense_8 (Dense)             (None, 2)                 18

=================================================================
Total params: 2698 (10.54 KB)
Trainable params: 2698 (10.54 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [34]:

```
model3.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accurac
history=model3.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=
```

```
Epoch 1/100
1/1 - 2s - loss: 0.6831 - accuracy: 0.5833 - val_loss: 0.7003 - val_ac
curacy: 0.3333 - 2s/epoch - 2s/step
Epoch 2/100
1/1 - 0s - loss: 0.6792 - accuracy: 0.6667 - val_loss: 0.6990 - val_ac
curacy: 0.3333 - 64ms/epoch - 64ms/step
Epoch 3/100
1/1 - 0s - loss: 0.6756 - accuracy: 0.6667 - val_loss: 0.6979 - val_ac
curacy: 0.3333 - 58ms/epoch - 58ms/step
Epoch 4/100
1/1 - 0s - loss: 0.6718 - accuracy: 0.6667 - val_loss: 0.6969 - val_ac
curacy: 0.3333 - 64ms/epoch - 64ms/step
Epoch 5/100
1/1 - 0s - loss: 0.6677 - accuracy: 0.6667 - val_loss: 0.6959 - val_ac
curacy: 0.3333 - 65ms/epoch - 65ms/step
Epoch 6/100
1/1 - 0s - loss: 0.6637 - accuracy: 0.6667 - val_loss: 0.6954 - val_ac
curacy: 0.3333 - 67ms/epoch - 67ms/step
Epoch 7/100
```

In [35]:

```
model3.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 45ms/step - loss: 1.6447 - accu
racy: 0.2000
```

Out[35]:

```
[1.6446993350982666, 0.20000000298023224]
```

Model 4 : Hidden Layer : 64

In [36]:

```
model4 = Sequential()
model4.add(Dense(64, activation='relu',input_dim=X_train.shape[1]))
model4.add(Dense(32, activation='relu'))
model4.add(Dense(16, activation='relu'))
model4.add(Dense(8, activation='relu'))
model4.add(Dense(2, activation='sigmoid'))
model4.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_9 (Dense)             (None, 64)                4032

 dense_10 (Dense)            (None, 32)                2080

 dense_11 (Dense)            (None, 16)                528

 dense_12 (Dense)            (None, 8)                 136

 dense_13 (Dense)            (None, 2)                 18

=================================================================
Total params: 6794 (26.54 KB)
Trainable params: 6794 (26.54 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [37]:

```
model4.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accurac
history=model4.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=
```

```
Epoch 1/100
1/1 - 2s - loss: 0.6941 - accuracy: 0.3333 - val_loss: 0.7014 - val_ac
curacy: 0.3333 - 2s/epoch - 2s/step
Epoch 2/100
1/1 - 0s - loss: 0.6841 - accuracy: 0.4167 - val_loss: 0.7032 - val_ac
curacy: 0.3333 - 78ms/epoch - 78ms/step
Epoch 3/100
1/1 - 0s - loss: 0.6763 - accuracy: 0.6667 - val_loss: 0.7050 - val_ac
curacy: 0.3333 - 60ms/epoch - 60ms/step
Epoch 4/100
1/1 - 0s - loss: 0.6693 - accuracy: 0.8333 - val_loss: 0.7073 - val_ac
curacy: 0.3333 - 61ms/epoch - 61ms/step
Epoch 5/100
1/1 - 0s - loss: 0.6638 - accuracy: 0.9167 - val_loss: 0.7095 - val_ac
curacy: 0.3333 - 54ms/epoch - 54ms/step
Epoch 6/100
1/1 - 0s - loss: 0.6587 - accuracy: 0.9167 - val_loss: 0.7112 - val_ac
curacy: 0.3333 - 55ms/epoch - 55ms/step
Epoch 7/100
```

In [38]:

```
model4.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 58ms/step - loss: 1.7602 - accu
racy: 0.0000e+00
```

Out[38]:

```
[1.7601642608642578, 0.0]
```

Model 5 : Hidden Layer : 128

In [39]:

```
model5 = Sequential()
model5.add(Dense(128, activation='relu',input_dim=X_train.shape[1]))
model5.add(Dense(64, activation='relu'))
model5.add(Dense(32, activation='relu'))
model5.add(Dense(16, activation='relu'))
model5.add(Dense(8, activation='relu'))
model5.add(Dense(2, activation='sigmoid'))
model5.summary()
```

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_14 (Dense)            (None, 128)               8064

 dense_15 (Dense)            (None, 64)                8256

 dense_16 (Dense)            (None, 32)                2080

 dense_17 (Dense)            (None, 16)                528

 dense_18 (Dense)            (None, 8)                 136

 dense_19 (Dense)            (None, 2)                 18

=================================================================
Total params: 19082 (74.54 KB)
Trainable params: 19082 (74.54 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [40]:

```
model5.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accurac
history=model5.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=
```

```
Epoch 1/100
1/1 - 2s - loss: 0.6907 - accuracy: 0.6667 - val_loss: 0.6951 - val_ac
curacy: 0.6667 - 2s/epoch - 2s/step
Epoch 2/100
1/1 - 0s - loss: 0.6832 - accuracy: 1.0000 - val_loss: 0.6965 - val_ac
curacy: 0.3333 - 84ms/epoch - 84ms/step
Epoch 3/100
1/1 - 0s - loss: 0.6769 - accuracy: 1.0000 - val_loss: 0.6981 - val_ac
curacy: 0.3333 - 72ms/epoch - 72ms/step
Epoch 4/100
1/1 - 0s - loss: 0.6708 - accuracy: 1.0000 - val_loss: 0.7008 - val_ac
curacy: 0.3333 - 86ms/epoch - 86ms/step
Epoch 5/100
1/1 - 0s - loss: 0.6648 - accuracy: 1.0000 - val_loss: 0.7040 - val_ac
curacy: 0.3333 - 82ms/epoch - 82ms/step
Epoch 6/100
1/1 - 0s - loss: 0.6587 - accuracy: 1.0000 - val_loss: 0.7074 - val_ac
curacy: 0.3333 - 79ms/epoch - 79ms/step
Epoch 7/100
```

In [41]:

```
model5.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 46ms/step - loss: 2.7402 - accu
racy: 0.0000e+00
```

Out[41]:

```
[2.7402262687683105, 0.0]
```

Model 6 : Hidden Layer : 256

In [42]:

```
model6 = Sequential()
model6.add(Dense(256, activation='relu',input_dim=X_train.shape[1]))
model6.add(Dense(128, activation='relu'))
model6.add(Dense(64, activation='relu'))
model6.add(Dense(32, activation='relu'))
model6.add(Dense(16, activation='relu'))
model6.add(Dense(6, activation='relu'))
model6.add(Dense(2, activation='sigmoid'))
model6.summary()
```

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_20 (Dense)            (None, 256)               16128

 dense_21 (Dense)            (None, 128)               32896

 dense_22 (Dense)            (None, 64)                8256

 dense_23 (Dense)            (None, 32)                2080

 dense_24 (Dense)            (None, 16)                528

 dense_25 (Dense)            (None, 6)                 102

 dense_26 (Dense)            (None, 2)                 14

=================================================================
Total params: 60004 (234.39 KB)
Trainable params: 60004 (234.39 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [43]:

```
model6.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accurac
history=model6.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=
```

```
Epoch 5/100
1/1 - 0s - loss: 0.6802 - accuracy: 1.0000 - val_loss: 0.6893 - val_ac
curacy: 0.3333 - 58ms/epoch - 58ms/step
Epoch 6/100
1/1 - 0s - loss: 0.6770 - accuracy: 1.0000 - val_loss: 0.6902 - val_ac
curacy: 0.3333 - 56ms/epoch - 56ms/step
Epoch 7/100
1/1 - 0s - loss: 0.6730 - accuracy: 1.0000 - val_loss: 0.6907 - val_ac
curacy: 0.6667 - 61ms/epoch - 61ms/step
Epoch 8/100
1/1 - 0s - loss: 0.6698 - accuracy: 1.0000 - val_loss: 0.6923 - val_ac
curacy: 0.6667 - 53ms/epoch - 53ms/step
Epoch 9/100
1/1 - 0s - loss: 0.6646 - accuracy: 1.0000 - val_loss: 0.6947 - val_ac
curacy: 0.6667 - 59ms/epoch - 59ms/step
Epoch 10/100
1/1 - 0s - loss: 0.6603 - accuracy: 1.0000 - val_loss: 0.6956 - val_ac
curacy: 0.6667 - 54ms/epoch - 54ms/step
Epoch 11/100
1/1 - 0s - loss: 0.6555 - accuracy: 1.0000 - val_loss: 0.6961 - val_ac
```

In [44]:

```
model6.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 45ms/step - loss: 5.1345 - accu
racy: 0.0000e+00
```

Out[44]:

```
[5.134530067443848, 0.0]
```

Model 7 : Hidden Layer : 512

In [45]:

```
model7 = Sequential()
model7.add(Dense(512, activation='relu',input_dim=X_train.shape[1]))
model7.add(Dense(256, activation='relu'))
model7.add(Dense(128, activation='relu'))
model7.add(Dense(64, activation='relu'))
model7.add(Dense(32, activation='relu'))
model7.add(Dense(16, activation='relu'))
model7.add(Dense(8, activation='relu'))
model7.add(Dense(2, activation='sigmoid'))
model7.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_27 (Dense)            (None, 512)               32256

 dense_28 (Dense)            (None, 256)               131328

 dense_29 (Dense)            (None, 128)               32896

 dense_30 (Dense)            (None, 64)                8256

 dense_31 (Dense)            (None, 32)                2080

 dense_32 (Dense)            (None, 16)                528

 dense_33 (Dense)            (None, 8)                 136

 dense_34 (Dense)            (None, 2)                 18

=================================================================
Total params: 207498 (810.54 KB)
Trainable params: 207498 (810.54 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [46]:

```
model7.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accurac
history=model7.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=
```

```
Epoch 1/100
1/1 - 3s - loss: 0.6870 - accuracy: 0.6667 - val_loss: 0.7017 - val_ac
curacy: 0.3333 - 3s/epoch - 3s/step
Epoch 2/100
1/1 - 0s - loss: 0.6729 - accuracy: 0.6667 - val_loss: 0.7075 - val_ac
curacy: 0.3333 - 70ms/epoch - 70ms/step
Epoch 3/100
1/1 - 0s - loss: 0.6533 - accuracy: 0.6667 - val_loss: 0.7161 - val_ac
curacy: 0.3333 - 74ms/epoch - 74ms/step
Epoch 4/100
1/1 - 0s - loss: 0.6324 - accuracy: 0.6667 - val_loss: 0.7281 - val_ac
curacy: 0.3333 - 76ms/epoch - 76ms/step
Epoch 5/100
1/1 - 0s - loss: 0.6070 - accuracy: 0.6667 - val_loss: 0.7417 - val_ac
curacy: 0.3333 - 80ms/epoch - 80ms/step
Epoch 6/100
1/1 - 0s - loss: 0.5782 - accuracy: 0.6667 - val_loss: 0.7613 - val_ac
curacy: 0.3333 - 66ms/epoch - 66ms/step
Epoch 7/100
```

In [47]:

```
model7.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 54ms/step - loss: 4.6134 - accu
racy: 0.0000e+00
```

Out[47]:

```
[4.6134138107299805, 0.0]
```

Model 8 : Hidden Layer : 1028

In [48]:

```
model8 = Sequential()
model8.add(Dense(1028, activation='relu',input_dim=X_train.shape[1]))
model8.add(Dense(512, activation='relu'))
model8.add(Dense(356, activation='relu'))
model8.add(Dense(128, activation='relu'))
model8.add(Dense(64, activation='relu'))
model8.add(Dense(32, activation='relu'))
model8.add(Dense(16, activation='relu'))
model8.add(Dense(8, activation='relu'))
model8.add(Dense(2, activation='sigmoid'))
model8.summary()
```

```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_35 (Dense)            (None, 1028)              64764

 dense_36 (Dense)            (None, 512)               526848

 dense_37 (Dense)            (None, 356)               182628

 dense_38 (Dense)            (None, 128)               45696

 dense_39 (Dense)            (None, 64)                8256

 dense_40 (Dense)            (None, 32)                2080

 dense_41 (Dense)            (None, 16)                528

 dense_42 (Dense)            (None, 8)                 136

 dense_43 (Dense)            (None, 2)                 18

=================================================================
Total params: 830954 (3.17 MB)
Trainable params: 830954 (3.17 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [49]:

```
model8.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accurac
history=model8.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=
```

```
Epoch 1/100
1/1 - 3s - loss: 0.6959 - accuracy: 0.3333 - val_loss: 0.6933 - val_ac
curacy: 0.6667 - 3s/epoch - 3s/step
Epoch 2/100
1/1 - 0s - loss: 0.6935 - accuracy: 0.3333 - val_loss: 0.6926 - val_ac
curacy: 0.6667 - 87ms/epoch - 87ms/step
Epoch 3/100
1/1 - 0s - loss: 0.6894 - accuracy: 0.5000 - val_loss: 0.6934 - val_ac
curacy: 0.3333 - 86ms/epoch - 86ms/step
Epoch 4/100
1/1 - 0s - loss: 0.6864 - accuracy: 0.7500 - val_loss: 0.6946 - val_ac
curacy: 0.3333 - 88ms/epoch - 88ms/step
Epoch 5/100
1/1 - 0s - loss: 0.6798 - accuracy: 0.8333 - val_loss: 0.6952 - val_ac
curacy: 0.3333 - 101ms/epoch - 101ms/step
Epoch 6/100
1/1 - 0s - loss: 0.6720 - accuracy: 1.0000 - val_loss: 0.6953 - val_ac
curacy: 0.3333 - 111ms/epoch - 111ms/step
Epoch 7/100
```

In [50]:

```
model8.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 57ms/step - loss: 16.2926 - acc
uracy: 0.2000
```

Out[50]:

```
[16.292612075805664, 0.20000000298023224]
```

Model 1 : Outer Layer : 2

In [51]:

```
model = Sequential()
model.add(Dense(32, activation='relu',input_dim=X_train.shape[1]))
model.add(Dense(2, activation='sigmoid'))
model.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_44 (Dense)            (None, 32)                2016

 dense_45 (Dense)            (None, 2)                 66

=================================================================
Total params: 2082 (8.13 KB)
Trainable params: 2082 (8.13 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [52]:

```
model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accuracy
history=model.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=1
```

```
Epoch 1/100
1/1 - 1s - loss: 0.6854 - accuracy: 0.5833 - val_loss: 0.7520 - val_ac
curacy: 0.3333 - 1s/epoch - 1s/step
Epoch 2/100
1/1 - 0s - loss: 0.6794 - accuracy: 0.5833 - val_loss: 0.7539 - val_ac
curacy: 0.3333 - 68ms/epoch - 68ms/step
Epoch 3/100
1/1 - 0s - loss: 0.6735 - accuracy: 0.5833 - val_loss: 0.7558 - val_ac
curacy: 0.3333 - 63ms/epoch - 63ms/step
Epoch 4/100
1/1 - 0s - loss: 0.6678 - accuracy: 0.5833 - val_loss: 0.7578 - val_ac
curacy: 0.3333 - 62ms/epoch - 62ms/step
Epoch 5/100
1/1 - 0s - loss: 0.6621 - accuracy: 0.6667 - val_loss: 0.7598 - val_ac
curacy: 0.3333 - 63ms/epoch - 63ms/step
Epoch 6/100
1/1 - 0s - loss: 0.6564 - accuracy: 0.6667 - val_loss: 0.7619 - val_ac
curacy: 0.3333 - 63ms/epoch - 63ms/step
Epoch 7/100
```

In [53]:

```
model.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 47ms/step - loss: 0.9879 - accu
racy: 0.2000
```

Out[53]:

```
[0.987899124622345, 0.20000000298023224]
```

Model 2 : Outer Layer : 3

In [54]:

```
model0 = Sequential()
model0.add(Dense(32, activation='relu',input_dim=X_train.shape[1]))
model0.add(Dense(3, activation='sigmoid'))
model0.summary()
```

```
Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_46 (Dense)            (None, 32)                2016

 dense_47 (Dense)            (None, 3)                 99

=================================================================
Total params: 2115 (8.26 KB)
Trainable params: 2115 (8.26 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [55]:

```
model0.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accurac
history=model0.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size=
```

```
1/1 - 0s - loss: 1.0254 - accuracy: 0.5000 - val_loss: 1.0224 - val_ac
curacy: 0.6667 - 109ms/epoch - 109ms/step
Epoch 7/100
1/1 - 0s - loss: 1.0177 - accuracy: 0.7500 - val_loss: 1.0215 - val_ac
curacy: 0.6667 - 69ms/epoch - 69ms/step
Epoch 8/100
1/1 - 0s - loss: 1.0100 - accuracy: 0.7500 - val_loss: 1.0206 - val_ac
curacy: 0.6667 - 59ms/epoch - 59ms/step
Epoch 9/100
1/1 - 0s - loss: 1.0023 - accuracy: 0.7500 - val_loss: 1.0197 - val_ac
curacy: 0.6667 - 56ms/epoch - 56ms/step
Epoch 10/100
1/1 - 0s - loss: 0.9947 - accuracy: 0.7500 - val_loss: 1.0187 - val_ac
curacy: 0.6667 - 56ms/epoch - 56ms/step
Epoch 11/100
1/1 - 0s - loss: 0.9872 - accuracy: 0.7500 - val_loss: 1.0179 - val_ac
curacy: 0.6667 - 53ms/epoch - 53ms/step
Epoch 12/100
1/1 - 0s - loss: 0.9796 - accuracy: 0.7500 - val_loss: 1.0171 - val_ac
```

In [56]:

```
model0.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 49ms/step - loss: 1.2712 - accu
racy: 0.4000
```

Out[56]:

```
[1.27116060256958, 0.4000000059604645]
```

Model 3 : Outer Layer : 4

In [57]:

```
model01 = Sequential()
model01.add(Dense(32, activation='relu',input_dim=X_train.shape[1]))
model01.add(Dense(4, activation='sigmoid'))
model01.summary()
```

```
Model: "sequential_10"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_48 (Dense)            (None, 32)                2016

 dense_49 (Dense)            (None, 4)                 132

=================================================================
Total params: 2148 (8.39 KB)
Trainable params: 2148 (8.39 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [58]:

```
model01.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accura
history=model01.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size
```

```
Epoch 1/100
1/1 - 1s - loss: 1.4518 - accuracy: 0.0833 - val_loss: 1.4335 - val_ac
curacy: 0.0000e+00 - 1s/epoch - 1s/step
Epoch 2/100
1/1 - 0s - loss: 1.4416 - accuracy: 0.0833 - val_loss: 1.4308 - val_ac
curacy: 0.0000e+00 - 73ms/epoch - 73ms/step
Epoch 3/100
1/1 - 0s - loss: 1.4315 - accuracy: 0.1667 - val_loss: 1.4282 - val_ac
curacy: 0.0000e+00 - 61ms/epoch - 61ms/step
Epoch 4/100
1/1 - 0s - loss: 1.4215 - accuracy: 0.2500 - val_loss: 1.4255 - val_ac
curacy: 0.0000e+00 - 59ms/epoch - 59ms/step
Epoch 5/100
1/1 - 0s - loss: 1.4115 - accuracy: 0.2500 - val_loss: 1.4228 - val_ac
curacy: 0.0000e+00 - 59ms/epoch - 59ms/step
Epoch 6/100
1/1 - 0s - loss: 1.4016 - accuracy: 0.2500 - val_loss: 1.4201 - val_ac
curacy: 0.0000e+00 - 59ms/epoch - 59ms/step
Epoch 7/100
```

In [59]:

```
model01.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 46ms/step - loss: 1.4003 - accu
racy: 0.2000
```

Out[59]:

```
[1.4003479480743408, 0.20000000298023224]
```

Model 4 : Outer Layer : 5

In [60]:

```
model02 = Sequential()
model02.add(Dense(32, activation='relu',input_dim=X_train.shape[1]))
model02.add(Dense(5, activation='sigmoid'))
model02.summary()
```

```
Model: "sequential_11"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_50 (Dense)            (None, 32)                2016

 dense_51 (Dense)            (None, 5)                 165

=================================================================
Total params: 2181 (8.52 KB)
Trainable params: 2181 (8.52 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [61]:

```
model02.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accura
history=model02.fit(X_train,y_train,epochs=100,verbose=2,validation_split=0.2,batch_size
```

```
1/1 - 0s - loss: 1.5989 - accuracy: 0.0833 - val_loss: 1.4883 - val_ac
curacy: 0.6667 - 65ms/epoch - 65ms/step
Epoch 4/100
1/1 - 0s - loss: 1.5866 - accuracy: 0.2500 - val_loss: 1.4838 - val_ac
curacy: 0.6667 - 55ms/epoch - 55ms/step
Epoch 5/100
1/1 - 0s - loss: 1.5743 - accuracy: 0.3333 - val_loss: 1.4794 - val_ac
curacy: 0.6667 - 62ms/epoch - 62ms/step
Epoch 6/100
1/1 - 0s - loss: 1.5621 - accuracy: 0.5000 - val_loss: 1.4750 - val_ac
curacy: 0.6667 - 62ms/epoch - 62ms/step
Epoch 7/100
1/1 - 0s - loss: 1.5500 - accuracy: 0.5833 - val_loss: 1.4706 - val_ac
curacy: 0.6667 - 53ms/epoch - 53ms/step
Epoch 8/100
1/1 - 0s - loss: 1.5380 - accuracy: 0.5833 - val_loss: 1.4663 - val_ac
curacy: 0.6667 - 67ms/epoch - 67ms/step
Epoch 9/100
1/1 - 0s - loss: 1.5261 - accuracy: 0.5833 - val_loss: 1.4620 - val_ac
curacy: 1.0000 - 54ms/epoch - 54ms/step
```

In [62]:

```
model02.evaluate(X_test,y_test)
```

```
1/1 [==============================] - 0s 47ms/step - loss: 1.4644 - accu
racy: 0.4000
```

Out[62]:

```
[1.4643977880477905, 0.4000000059604645]
```

In [ ]: