In [23]:
```
import tensorflow as tf
import numpy as np
import pandas as pd
from sklearn import preprocessing
```

In [26]:
```
dfraw=np.loadtxt('/content/19664156-Audiobooks-data[UdemyIran.Com].csv', delim
df_input= dfraw[:,1:-1]
targetsall=dfraw[:,-1]
```

In [68]:
```
sum_of_targets= int(np.sum(targetsall))
zero_counter= 0
deletelist=[]
for i in range(targetsall.shape[0]):
  if targetsall[1]==0:
    zero_counter +=1
    if zero_counter> sum_of_targets:
      deletelist.append(i)

unscaled_inputs= np.delete(df_input, deletelist, axis=0)
targets_equal_periors= np.delete(targetsall, deletelist, axis=0)
```

In [69]:
```
scaled_inputs= preprocessing.scale (unscaled_inputs)
```

In [70]:
```
scaled_inputs2= np.arange(scaled_inputs.shape[0])
np.random.shuffle(scaled_inputs2)

shuffled_inputs= scaled_inputs[scaled_inputs2]
shuffled_targets=targets_equal_periors[scaled_inputs2]
```

In [71]:
```
sample_count= scaled_inputs.shape[0]
train_samples=int(sample_count*0.8)
validate_sample=int(sample_count*0.1)
test_sample=sample_count - train_samples-validate_sample
```

Alt+Q

In [72]:
```
train_inputs= shuffled_inputs[:train_samples]
train_targets=shuffled_targets[:train_samples]
validation_inputs= shuffled_inputs[train_samples:train_samples+validate_sample
validation_targets= shuffled_targets[train_samples:train_samples+validate_samp
test_inputs= shuffled_inputs[train_samples+validate_sample:]
test_targets=shuffled_targets[train_samples+validate_sample:]
print(np.sum(train_targets), train_samples)
print(np.sum(validation_targets), validate_sample)
print(np.sum(test_targets), test_sample)
```

```
428.0 1789
54.0 223
43.0 225
```

In [73]:
```
np.savez('audiobooks_train', inputs= train_inputs, targets=train_targets)
np.savez('audiobook_validation', inputs= validation_inputs, targets=validatic
np.savez('audiobook_test', inputs= test_inputs, targets= test_targets)
```

Alt+Q

In [74]:
```
npz= np.load('/content/audiobooks_train.npz')
train_inputs= npz['inputs'].astype(np.float)
train_targets=npz['targets'].astype(np.int)

npz=np.load('/content/audiobook_validation.npz')
validation_inputs=npz['inputs'].astype(np.float)
validation_targets=npz['targets'].astype(np.int)

npz= np.load('/content/audiobook_test.npz')
test_inputs=npz['inputs'].astype(np.float)
test_targets=npz['targets'].astype(np.int)
```

Alt+Q

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: DeprecationWa
rning: `np.float` is a deprecated alias for the builtin `float`. To silence t
his warning, use `float` by itself. Doing this will not modify any behavior a
nd is safe. If you specifically wanted the numpy scalar type, use `np.float64
` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: DeprecationWa
rning: `np.int` is a deprecated alias for the builtin `int`. To silence this
warning, use `int` by itself. Doing this will not modify any behavior and is
safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.int
32` to specify the precision. If you wish to review your current use, check t
he release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  This is separate from the ipykernel package so we can avoid doing imports u
ntil
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: DeprecationWa
rning: `np.float` is a deprecated alias for the builtin `float`. To silence t
his warning, use `float` by itself. Doing this will not modify any behavior a
nd is safe. If you specifically wanted the numpy scalar type, use `np.float64
` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: DeprecationWa
rning: `np.int` is a deprecated alias for the builtin `int`. To silence this
warning, use `int` by itself. Doing this will not modify any behavior and is
safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.int
32` to specify the precision. If you wish to review your current use, check t
he release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  import sys
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:10: DeprecationW
arning: `np.float` is a deprecated alias for the builtin `float`. To silence
this warning, use `float` by itself. Doing this will not modify any behavior
and is safe. If you specifically wanted the numpy scalar type, use `np.float6
4` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  # Remove the CWD from sys.path while we load stuff.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:11: DeprecationW
arning: `np.int` is a deprecated alias for the builtin `int`. To silence this
warning, use `int` by itself. Doing this will not modify any behavior and is
safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.int
32` to specify the precision. If you wish to review your current use, check t
he release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
```

```
se/1.20.0-notes.html#deprecations)
  # This is added back by InteractiveShellApp.init_path()
```

In [ ]:

```

```

Alt+Q

In [75]:
```
input_size=10
output_size=2
hidden_layer_size=50
model=tf.keras.Sequential([tf.keras.layers.Dense(hidden_layer_size, activation
                           tf.keras.layers.Dense(hidden_layer_size, activation
                           tf.keras.layers.Dense(output_size, activation='soft

])
model.compile(optimizer='adam',loss= 'sparse_categorical_crossentropy', metric

batch_size=50
max_epochs=50
early_stop=tf.keras.callbacks.EarlyStopping(patience=4)
model.fit(train_inputs,
          train_targets,
          batch_size=batch_size,
          epochs=max_epochs,
          callbacks=[early_stop],
          validation_data=(validation_inputs, validation_targets),
          verbose=1)
```

Alt+Q

```
Epoch 1/50
36/36 [==============================] - 1s 7ms/step - loss: 0.5425 - accurac
y: 0.7742 - val_loss: 0.4603 - val_accuracy: 0.8072
Epoch 2/50
36/36 [==============================] - 0s 3ms/step - loss: 0.4171 - accurac
y: 0.8278 - val_loss: 0.3796 - val_accuracy: 0.8430
Epoch 3/50
36/36 [==============================] - 0s 3ms/step - loss: 0.3613 - accurac
y: 0.8463 - val_loss: 0.3468 - val_accuracy: 0.8475
Epoch 4/50
36/36 [==============================] - 0s 3ms/step - loss: 0.3378 - accurac
y: 0.8496 - val_loss: 0.3212 - val_accuracy: 0.8520
Epoch 5/50
36/36 [==============================] - 0s 4ms/step - loss: 0.3216 - accurac
y: 0.8569 - val_loss: 0.3058 - val_accuracy: 0.8520
Epoch 6/50
36/36 [==============================] - 0s 3ms/step - loss: 0.3107 - accurac
y: 0.8563 - val_loss: 0.2967 - val_accuracy: 0.8520
Epoch 7/50
36/36 [==============================] - 0s 3ms/step - loss: 0.3022 - accurac
y: 0.8552 - val_loss: 0.2886 - val_accuracy: 0.8879
Epoch 8/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2982 - accurac
y: 0.8653 - val_loss: 0.2883 - val_accuracy: 0.8610
Epoch 9/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2954 - accurac
y: 0.8619 - val_loss: 0.2778 - val_accuracy: 0.8834
Epoch 10/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2932 - accurac
y: 0.8658 - val_loss: 0.2793 - val_accuracy: 0.8744
Epoch 11/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2890 - accurac
y: 0.8653 - val_loss: 0.2709 - val_accuracy: 0.8879
Epoch 12/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2832 - accurac
y: 0.8675 - val_loss: 0.2758 - val_accuracy: 0.8744
Epoch 13/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2825 - accurac
y: 0.8636 - val_loss: 0.2707 - val_accuracy: 0.8834
Epoch 14/50
36/36 [==============================] - 0s 4ms/step - loss: 0.2797 - accurac
y: 0.8709 - val_loss: 0.2661 - val_accuracy: 0.8834
Epoch 15/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2774 - accurac
y: 0.8709 - val_loss: 0.2696 - val_accuracy: 0.8744
Epoch 16/50
36/36 [==============================] - 0s 4ms/step - loss: 0.2754 - accurac
y: 0.8720 - val_loss: 0.2638 - val_accuracy: 0.8834
Epoch 17/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2774 - accurac
y: 0.8731 - val_loss: 0.2652 - val_accuracy: 0.8834
Epoch 18/50
36/36 [==============================] - 0s 4ms/step - loss: 0.2760 - accurac
y: 0.8748 - val_loss: 0.2589 - val_accuracy: 0.8879
Epoch 19/50
36/36 [==============================] - 0s 4ms/step - loss: 0.2741 - accurac
y: 0.8720 - val_loss: 0.2625 - val_accuracy: 0.8744
```

```
Epoch 20/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2695 - accurac
y: 0.8742 - val_loss: 0.2726 - val_accuracy: 0.8789
Epoch 21/50
36/36 [==============================] - 0s 3ms/step - loss: 0.2713 - accurac
y: 0.8703 - val_loss: 0.2641 - val_accuracy: 0.8789
Epoch 22/50
36/36 [==============================] - 0s 4ms/step - loss: 0.2699 - accurac
y: 0.8781 - val_loss: 0.2630 - val_accuracy: 0.8834
```

Out[75]:

```
<keras.callbacks.History at 0x7f95021dead0>
```

In [76]:

```
test_accuracy=model.evaluate(test_inputs, test_targets)
```

```
8/8 [==============================] - 0s 2ms/step - loss: 0.2821 - accuracy:
0.8933
```

In [77]:

```
print(test_accuracy)
```

```
[0.28206294775009155, 0.8933333158493042]
```

In [ ]:

```

```

Alt+Q