Part 3 ADTA 5550 Deep Learning with Big Data

In [12]:

```
pip install tensorflow
```

```
Requirement already satisfied: tensorflow in /opt/conda/lib/python3.7/site-packages
(1.15.5)
Requirement already satisfied: grpcio>=1.8.6 in /opt/conda/lib/python3.7/site-package
s (from tensorflow) (1.50.0)
Requirement already satisfied: astor>=0.6.0 in /opt/conda/lib/python3.7/site-packages
(from tensorflow) (0.8.1)
Requirement already satisfied: keras-preprocessing>=1.0.5 in /opt/conda/lib/python3.
7/site-packages (from tensorflow) (1.1.2)
Requirement already satisfied: gast==0.2.2 in /opt/conda/lib/python3.7/site-packages
(from tensorflow) (0.2.2)
Requirement already satisfied: tensorflow-estimator==1.15.1 in /opt/conda/lib/python
3.7/site-packages (from tensorflow) (1.15.1)
Requirement already satisfied: wrapt>=1.11.1 in /opt/conda/lib/python3.7/site-package
s (from tensorflow) (1.14.1)
Requirement already satisfied: h5py<=2.10.0 in /opt/conda/lib/python3.7/site-packages
(from tensorflow) (2.10.0)
Requirement already satisfied: google-pasta>=0.1.6 in /opt/conda/lib/python3.7/site-p
ackages (from tensorflow) (0.2.0)
Requirement already satisfied: protobuf>=3.6.1 in /opt/conda/lib/python3.7/site-packa
ges (from tensorflow) (3.20.3)
Requirement already satisfied: termcolor>=1.1.0 in /opt/conda/lib/python3.7/site-pack
ages (from tensorflow) (2.1.0)
Requirement already satisfied: wheel>=0.26 in /opt/conda/lib/python3.7/site-packages
(from tensorflow) (0.37.1)
Requirement already satisfied: six>=1.10.0 in /opt/conda/lib/python3.7/site-packages
(from tensorflow) (1.16.0)
Requirement already satisfied: opt-einsum>=2.3.2 in /opt/conda/lib/python3.7/site-pac
kages (from tensorflow) (3.3.0)
Requirement already satisfied: tensorboard<1.16.0,>=1.15.0 in /opt/conda/lib/python3.
7/site-packages (from tensorflow) (1.15.0)
Requirement already satisfied: keras-applications>=1.0.8 in /opt/conda/lib/python3.7/
site-packages (from tensorflow) (1.0.8)
Requirement already satisfied: absl-py>=0.7.0 in /opt/conda/lib/python3.7/site-packag
es (from tensorflow) (0.8.1)
Requirement already satisfied: numpy<1.19.0,>=1.16.0 in /opt/conda/lib/python3.7/site
-packages (from tensorflow) (1.18.5)
Requirement already satisfied: markdown>=2.6.8 in /opt/conda/lib/python3.7/site-packa
ges (from tensorboard<1.16.0,>=1.15.0->tensorflow) (3.4.1)
Requirement already satisfied: setuptools>=41.0.0 in /opt/conda/lib/python3.7/site-pa
ckages (from tensorboard<1.16.0,>=1.15.0->tensorflow) (59.8.0)
Requirement already satisfied: werkzeug>=0.11.15 in /opt/conda/lib/python3.7/site-pac
kages (from tensorboard<1.16.0,>=1.15.0->tensorflow) (2.2.2)
Requirement already satisfied: importlib-metadata>=4.4 in /opt/conda/lib/python3.7/si
te-packages (from markdown>=2.6.8->tensorboard<1.16.0,>=1.15.0->tensorflow) (4.11.4)
Requirement already satisfied: MarkupSafe>=2.1.1 in /opt/conda/lib/python3.7/site-pac
kages (from werkzeug>=0.11.15->tensorboard<1.16.0,>=1.15.0->tensorflow) (2.1.1)
Requirement already satisfied: zipp>=0.5 in /opt/conda/lib/python3.7/site-packages (f
rom importlib-metadata>=4.4->markdown>=2.6.8->tensorboard<1.16.0,>=1.15.0->tensorflo
w) (3.10.0)
Requirement already satisfied: typing-extensions>=3.6.4 in /opt/conda/lib/python3.7/s
ite-packages (from importlib-metadata>=4.4->markdown>=2.6.8->tensorboard<1.16.0,>=1.1
5.0->tensorflow) (4.4.0)
Note: you may need to restart the kernel to use updated packages.
```

In [13]:

```python
import pandas as pd
import numpy as np
from pandas.plotting import scatter_matrix
from matplotlib import pyplot
from sklearn.model_selection import train_test_split   #Train and Test data
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from keras.utils import np_utils
from sklearn.model_selection import cross_val_predict, KFold
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

In [14]:

```
# Load the dataset
import os
cwd = os.getcwd()
print(cwd)
path = cwd + '/Data/'
print(path)
#df = path
file = path + 'pima_diabetes.csv'
#df = pd.read_csv(file, header=None).values

pima_diabetes_data = pd.read_csv("pima_diabetes.csv", header=None).values

#dataset = pd.read_csv(file, header=None).values

X = pima_diabetes_data[:, :-1]
y = pima_diabetes_data[:, -1]

# Preprocess the data
scaler = StandardScaler()
X = scaler.fit_transform(X)
y = np.reshape(y, (-1, 1))

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=
```

```
/home/yogchaudhary9/JPTR_NTBK
/home/yogchaudhary9/JPTR_NTBK/Data/
```

In [15]:

```
pima_diabetes_data
```

Out[15]:

```
array([[  6.   , 148.   ,  72.   , ...,   0.627,  50.   ,   1.   ],
       [  1.   ,  85.   ,  66.   , ...,   0.351,  31.   ,   0.   ],
       [  8.   , 183.   ,  64.   , ...,   0.672,  32.   ,   1.   ],
       ...,
       [  5.   , 121.   ,  72.   , ...,   0.245,  30.   ,   0.   ],
       [  1.   , 126.   ,  60.   , ...,   0.349,  47.   ,   1.   ],
       [  1.   ,  93.   ,  70.   , ...,   0.315,  23.   ,   0.   ]])
```

In [16]:

```
from keras.models import Sequential
from keras.layers import Dense, Dropout
from keras.constraints import max_norm  # Import max_norm constraint

# Define the model architecture
model = Sequential()
model.add(Dense(32, input_dim=X_train.shape[1], activation='relu', kernel_constraint=m
model.add(Dropout(0.2))
model.add(Dense(32, activation='relu', kernel_constraint=max_norm(3)))  # Example max_
model.add(Dropout(0.2))
model.add(Dense(32, activation='relu', kernel_constraint=max_norm(3)))  # Example max_
model.add(Dropout(0.2))
model.add(Dense(1, activation='sigmoid'))
```

Complie Model

In [17]:

```
# Compile the model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

In [18]:

```
from keras.models import Sequential
from keras.layers import Dense, Dropout

# Define the model
model = Sequential()

# Add the input layer
model.add(Dense(32, activation='relu', input_dim=X_train.shape[1]))

# Add the first hidden layer
model.add(Dense(32, activation='relu'))

# Add dropout regularization
model.add(Dropout(0.2))

# Add the second hidden layer
model.add(Dense(32, activation='relu'))

# Add dropout regularization
model.add(Dropout(0.2))

# Add the output layer
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

Train Model

In [19]:

```
# Train the model
history = model.fit(X_train, y_train, epochs=100, batch_size=32, validation_data=(X_te
Train on 614 samples, validate on 154 samples
Epoch 1/100
614/614 [==============================] - 1s 2ms/step - loss: 0.6785 - accuracy: 0.6
010 - val_loss: 0.6516 - val_accuracy: 0.6948
Epoch 2/100
614/614 [==============================] - 0s 144us/step - loss: 0.6294 - accuracy:
0.6987 - val_loss: 0.6075 - val_accuracy: 0.7143
Epoch 3/100
614/614 [==============================] - 0s 139us/step - loss: 0.5792 - accuracy:
0.7296 - val_loss: 0.5625 - val_accuracy: 0.7403
Epoch 4/100
614/614 [==============================] - 0s 130us/step - loss: 0.5240 - accuracy:
0.7508 - val_loss: 0.5298 - val_accuracy: 0.7403
Epoch 5/100
614/614 [==============================] - 0s 142us/step - loss: 0.5093 - accuracy:
0.7394 - val_loss: 0.5160 - val_accuracy: 0.7338
Epoch 6/100
614/614 [==============================] - 0s 138us/step - loss: 0.4849 - accuracy:
0.7687 - val_loss: 0.5087 - val_accuracy: 0.7403
Epoch 7/100
614/614 [==============================] - 0s 133us/step - loss: 0.4790 - accuracy:
0.7655 - val_loss: 0.5072 - val_accuracy: 0.7468
Epoch 8/100
614/614 [==============================] - 0s 134us/step - loss: 0.4701 - accuracy:
0.7736 - val_loss: 0.5088 - val_accuracy: 0.7597
Epoch 9/100
614/614 [==============================] - 0s 133us/step - loss: 0.4717 - accuracy:
0.7785 - val_loss: 0.5110 - val_accuracy: 0.7727
Epoch 10/100
614/614 [==============================] - 0s 145us/step - loss: 0.4689 - accuracy:
0.7671 - val_loss: 0.5126 - val_accuracy: 0.7727
Epoch 11/100
614/614 [==============================] - 0s 150us/step - loss: 0.4519 - accuracy:
0.7769 - val_loss: 0.5150 - val_accuracy: 0.7727
Epoch 12/100
614/614 [==============================] - 0s 151us/step - loss: 0.4595 - accuracy:
0.7671 - val_loss: 0.5178 - val_accuracy: 0.7597
Epoch 13/100
614/614 [==============================] - 0s 137us/step - loss: 0.4526 - accuracy:
0.7720 - val_loss: 0.5201 - val_accuracy: 0.7597
Epoch 14/100
614/614 [==============================] - 0s 144us/step - loss: 0.4399 - accuracy:
0.7866 - val_loss: 0.5205 - val_accuracy: 0.7597
Epoch 15/100
614/614 [==============================] - 0s 147us/step - loss: 0.4462 - accuracy:
0.7883 - val_loss: 0.5197 - val_accuracy: 0.7597
Epoch 16/100
614/614 [==============================] - 0s 145us/step - loss: 0.4442 - accuracy:
0.7899 - val_loss: 0.5232 - val_accuracy: 0.7662
Epoch 17/100
614/614 [==============================] - 0s 147us/step - loss: 0.4518 - accuracy:
0.7899 - val_loss: 0.5285 - val_accuracy: 0.7727
Epoch 18/100
614/614 [==============================] - 0s 141us/step - loss: 0.4482 - accuracy:
0.7818 - val_loss: 0.5291 - val_accuracy: 0.7597
Epoch 19/100
614/614 [==============================] - 0s 132us/step - loss: 0.4452 - accuracy:
0.7752 - val_loss: 0.5300 - val_accuracy: 0.7662
Epoch 20/100
614/614 [==============================] - 0s 128us/step - loss: 0.4401 - accuracy:
```

```
0.7785 - val_loss: 0.5302 - val_accuracy: 0.7532
Epoch 21/100
614/614 [==============================] - 0s 140us/step - loss: 0.4421 - accuracy:
0.7801 - val_loss: 0.5286 - val_accuracy: 0.7597
Epoch 22/100
614/614 [==============================] - 0s 142us/step - loss: 0.4267 - accuracy:
0.7818 - val_loss: 0.5337 - val_accuracy: 0.7597
Epoch 23/100
614/614 [==============================] - 0s 141us/step - loss: 0.4406 - accuracy:
0.7752 - val_loss: 0.5293 - val_accuracy: 0.7792
Epoch 24/100
614/614 [==============================] - 0s 137us/step - loss: 0.4324 - accuracy:
0.7801 - val_loss: 0.5313 - val_accuracy: 0.7597
Epoch 25/100
614/614 [==============================] - 0s 140us/step - loss: 0.4379 - accuracy:
0.7866 - val_loss: 0.5307 - val_accuracy: 0.7597
Epoch 26/100
614/614 [==============================] - 0s 130us/step - loss: 0.4301 - accuracy:
0.7915 - val_loss: 0.5307 - val_accuracy: 0.7662
Epoch 27/100
614/614 [==============================] - 0s 138us/step - loss: 0.4274 - accuracy:
0.7932 - val_loss: 0.5337 - val_accuracy: 0.7662
Epoch 28/100
614/614 [==============================] - 0s 126us/step - loss: 0.4278 - accuracy:
0.7932 - val_loss: 0.5380 - val_accuracy: 0.7727
Epoch 29/100
614/614 [==============================] - 0s 131us/step - loss: 0.4087 - accuracy:
0.7932 - val_loss: 0.5457 - val_accuracy: 0.7662
Epoch 30/100
614/614 [==============================] - 0s 129us/step - loss: 0.4237 - accuracy:
0.7883 - val_loss: 0.5451 - val_accuracy: 0.7662
Epoch 31/100
614/614 [==============================] - 0s 129us/step - loss: 0.4234 - accuracy:
0.7866 - val_loss: 0.5460 - val_accuracy: 0.7597
Epoch 32/100
614/614 [==============================] - 0s 143us/step - loss: 0.4205 - accuracy:
0.7980 - val_loss: 0.5438 - val_accuracy: 0.7662
Epoch 33/100
614/614 [==============================] - 0s 136us/step - loss: 0.4068 - accuracy:
0.7948 - val_loss: 0.5506 - val_accuracy: 0.7662
Epoch 34/100
614/614 [==============================] - 0s 132us/step - loss: 0.4033 - accuracy:
0.8176 - val_loss: 0.5514 - val_accuracy: 0.7662
Epoch 35/100
614/614 [==============================] - 0s 131us/step - loss: 0.4184 - accuracy:
0.8062 - val_loss: 0.5495 - val_accuracy: 0.7532
Epoch 36/100
614/614 [==============================] - 0s 132us/step - loss: 0.4057 - accuracy:
0.8046 - val_loss: 0.5572 - val_accuracy: 0.7662
Epoch 37/100
614/614 [==============================] - 0s 133us/step - loss: 0.4058 - accuracy:
0.8078 - val_loss: 0.5593 - val_accuracy: 0.7597
Epoch 38/100
614/614 [==============================] - 0s 143us/step - loss: 0.4084 - accuracy:
0.7948 - val_loss: 0.5582 - val_accuracy: 0.7662
Epoch 39/100
614/614 [==============================] - 0s 123us/step - loss: 0.3986 - accuracy:
0.8046 - val_loss: 0.5574 - val_accuracy: 0.7662
Epoch 40/100
614/614 [==============================] - 0s 132us/step - loss: 0.4016 - accuracy:
```

```
0.8013 - val_loss: 0.5581 - val_accuracy: 0.7597
Epoch 41/100
614/614 [==============================] - 0s 133us/step - loss: 0.4080 - accuracy:
0.8013 - val_loss: 0.5640 - val_accuracy: 0.7597
Epoch 42/100
614/614 [==============================] - 0s 126us/step - loss: 0.4085 - accuracy:
0.8078 - val_loss: 0.5634 - val_accuracy: 0.7662
Epoch 43/100
614/614 [==============================] - 0s 130us/step - loss: 0.3965 - accuracy:
0.8143 - val_loss: 0.5598 - val_accuracy: 0.7597
Epoch 44/100
614/614 [==============================] - 0s 141us/step - loss: 0.4012 - accuracy:
0.8094 - val_loss: 0.5610 - val_accuracy: 0.7532
Epoch 45/100
614/614 [==============================] - 0s 148us/step - loss: 0.3941 - accuracy:
0.8127 - val_loss: 0.5611 - val_accuracy: 0.7468
Epoch 46/100
614/614 [==============================] - 0s 145us/step - loss: 0.3899 - accuracy:
0.8257 - val_loss: 0.5641 - val_accuracy: 0.7468
Epoch 47/100
614/614 [==============================] - 0s 152us/step - loss: 0.3986 - accuracy:
0.8078 - val_loss: 0.5643 - val_accuracy: 0.7532
Epoch 48/100
614/614 [==============================] - 0s 145us/step - loss: 0.4130 - accuracy:
0.8127 - val_loss: 0.5649 - val_accuracy: 0.7468
Epoch 49/100
614/614 [==============================] - 0s 148us/step - loss: 0.3795 - accuracy:
0.8160 - val_loss: 0.5715 - val_accuracy: 0.7532
Epoch 50/100
614/614 [==============================] - 0s 135us/step - loss: 0.3963 - accuracy:
0.8176 - val_loss: 0.5687 - val_accuracy: 0.7662
Epoch 51/100
614/614 [==============================] - 0s 131us/step - loss: 0.3866 - accuracy:
0.8062 - val_loss: 0.5803 - val_accuracy: 0.7532
Epoch 52/100
614/614 [==============================] - 0s 138us/step - loss: 0.3834 - accuracy:
0.7980 - val_loss: 0.5769 - val_accuracy: 0.7597
Epoch 53/100
614/614 [==============================] - 0s 139us/step - loss: 0.3826 - accuracy:
0.8094 - val_loss: 0.5828 - val_accuracy: 0.7792
Epoch 54/100
614/614 [==============================] - 0s 138us/step - loss: 0.3914 - accuracy:
0.8127 - val_loss: 0.5735 - val_accuracy: 0.7597
Epoch 55/100
614/614 [==============================] - 0s 145us/step - loss: 0.3838 - accuracy:
0.8208 - val_loss: 0.5700 - val_accuracy: 0.7597
Epoch 56/100
614/614 [==============================] - 0s 141us/step - loss: 0.3862 - accuracy:
0.8192 - val_loss: 0.5754 - val_accuracy: 0.7468
Epoch 57/100
614/614 [==============================] - 0s 139us/step - loss: 0.3832 - accuracy:
0.8274 - val_loss: 0.5824 - val_accuracy: 0.7597
Epoch 58/100
614/614 [==============================] - 0s 145us/step - loss: 0.3818 - accuracy:
0.8127 - val_loss: 0.5842 - val_accuracy: 0.7468
Epoch 59/100
614/614 [==============================] - 0s 142us/step - loss: 0.3854 - accuracy:
0.8127 - val_loss: 0.5771 - val_accuracy: 0.7532
Epoch 60/100
614/614 [==============================] - 0s 152us/step - loss: 0.3796 - accuracy:
```

```
0.8029 - val_loss: 0.5839 - val_accuracy: 0.7403
Epoch 61/100
614/614 [==============================] - 0s 162us/step - loss: 0.3682 - accuracy:
0.8176 - val_loss: 0.5804 - val_accuracy: 0.7532
Epoch 62/100
614/614 [==============================] - 0s 153us/step - loss: 0.3700 - accuracy:
0.8274 - val_loss: 0.5909 - val_accuracy: 0.7662
Epoch 63/100
614/614 [==============================] - 0s 150us/step - loss: 0.3728 - accuracy:
0.8192 - val_loss: 0.5817 - val_accuracy: 0.7727
Epoch 64/100
614/614 [==============================] - 0s 136us/step - loss: 0.3684 - accuracy:
0.8322 - val_loss: 0.5852 - val_accuracy: 0.7532
Epoch 65/100
614/614 [==============================] - 0s 142us/step - loss: 0.3647 - accuracy:
0.8274 - val_loss: 0.5876 - val_accuracy: 0.7532
Epoch 66/100
614/614 [==============================] - 0s 149us/step - loss: 0.3670 - accuracy:
0.8339 - val_loss: 0.5890 - val_accuracy: 0.7532
Epoch 67/100
614/614 [==============================] - 0s 132us/step - loss: 0.3592 - accuracy:
0.8371 - val_loss: 0.5847 - val_accuracy: 0.7532
Epoch 68/100
614/614 [==============================] - 0s 141us/step - loss: 0.3615 - accuracy:
0.8322 - val_loss: 0.5963 - val_accuracy: 0.7597
Epoch 69/100
614/614 [==============================] - 0s 143us/step - loss: 0.3582 - accuracy:
0.8404 - val_loss: 0.5991 - val_accuracy: 0.7597
Epoch 70/100
614/614 [==============================] - 0s 136us/step - loss: 0.3604 - accuracy:
0.8290 - val_loss: 0.6079 - val_accuracy: 0.7662
Epoch 71/100
614/614 [==============================] - 0s 143us/step - loss: 0.3556 - accuracy:
0.8453 - val_loss: 0.6052 - val_accuracy: 0.7597
Epoch 72/100
614/614 [==============================] - 0s 134us/step - loss: 0.3601 - accuracy:
0.8371 - val_loss: 0.5982 - val_accuracy: 0.7532
Epoch 73/100
614/614 [==============================] - 0s 129us/step - loss: 0.3580 - accuracy:
0.8371 - val_loss: 0.6071 - val_accuracy: 0.7468
Epoch 74/100
614/614 [==============================] - 0s 129us/step - loss: 0.3593 - accuracy:
0.8355 - val_loss: 0.6031 - val_accuracy: 0.7597
Epoch 75/100
614/614 [==============================] - 0s 131us/step - loss: 0.3599 - accuracy:
0.8355 - val_loss: 0.6023 - val_accuracy: 0.7662
Epoch 76/100
614/614 [==============================] - 0s 138us/step - loss: 0.3460 - accuracy:
0.8469 - val_loss: 0.6064 - val_accuracy: 0.7597
Epoch 77/100
614/614 [==============================] - 0s 141us/step - loss: 0.3639 - accuracy:
0.8339 - val_loss: 0.6018 - val_accuracy: 0.7532
Epoch 78/100
614/614 [==============================] - 0s 143us/step - loss: 0.3550 - accuracy:
0.8355 - val_loss: 0.6281 - val_accuracy: 0.7403
Epoch 79/100
614/614 [==============================] - 0s 136us/step - loss: 0.3536 - accuracy:
0.8322 - val_loss: 0.6045 - val_accuracy: 0.7532
Epoch 80/100
614/614 [==============================] - 0s 131us/step - loss: 0.3580 - accuracy:
```

```
0.8257 - val_loss: 0.6025 - val_accuracy: 0.7468
Epoch 81/100
614/614 [==============================] - 0s 139us/step - loss: 0.3423 - accuracy:
0.8502 - val_loss: 0.6234 - val_accuracy: 0.7532
Epoch 82/100
614/614 [==============================] - 0s 130us/step - loss: 0.3454 - accuracy:
0.8453 - val_loss: 0.6258 - val_accuracy: 0.7597
Epoch 83/100
614/614 [==============================] - 0s 130us/step - loss: 0.3541 - accuracy:
0.8339 - val_loss: 0.6292 - val_accuracy: 0.7597
Epoch 84/100
614/614 [==============================] - 0s 129us/step - loss: 0.3486 - accuracy:
0.8355 - val_loss: 0.6299 - val_accuracy: 0.7468
Epoch 85/100
614/614 [==============================] - 0s 131us/step - loss: 0.3390 - accuracy:
0.8453 - val_loss: 0.6432 - val_accuracy: 0.7468
Epoch 86/100
614/614 [==============================] - 0s 136us/step - loss: 0.3370 - accuracy:
0.8404 - val_loss: 0.6373 - val_accuracy: 0.7403
Epoch 87/100
614/614 [==============================] - 0s 144us/step - loss: 0.3411 - accuracy:
0.8502 - val_loss: 0.6327 - val_accuracy: 0.7403
Epoch 88/100
614/614 [==============================] - 0s 144us/step - loss: 0.3389 - accuracy:
0.8339 - val_loss: 0.6425 - val_accuracy: 0.7597
Epoch 89/100
614/614 [==============================] - 0s 144us/step - loss: 0.3332 - accuracy:
0.8502 - val_loss: 0.6345 - val_accuracy: 0.7532
Epoch 90/100
614/614 [==============================] - 0s 138us/step - loss: 0.3436 - accuracy:
0.8453 - val_loss: 0.6355 - val_accuracy: 0.7532
Epoch 91/100
614/614 [==============================] - 0s 133us/step - loss: 0.3305 - accuracy:
0.8534 - val_loss: 0.6510 - val_accuracy: 0.7468
Epoch 92/100
614/614 [==============================] - 0s 139us/step - loss: 0.3204 - accuracy:
0.8453 - val_loss: 0.6567 - val_accuracy: 0.7338
Epoch 93/100
614/614 [==============================] - 0s 129us/step - loss: 0.3219 - accuracy:
0.8583 - val_loss: 0.6591 - val_accuracy: 0.7403
Epoch 94/100
614/614 [==============================] - 0s 130us/step - loss: 0.3325 - accuracy:
0.8436 - val_loss: 0.6617 - val_accuracy: 0.7468
Epoch 95/100
614/614 [==============================] - 0s 130us/step - loss: 0.3316 - accuracy:
0.8567 - val_loss: 0.6613 - val_accuracy: 0.7468
Epoch 96/100
614/614 [==============================] - 0s 117us/step - loss: 0.3394 - accuracy:
0.8599 - val_loss: 0.6600 - val_accuracy: 0.7403
Epoch 97/100
614/614 [==============================] - 0s 127us/step - loss: 0.3325 - accuracy:
0.8664 - val_loss: 0.6681 - val_accuracy: 0.7338
Epoch 98/100
614/614 [==============================] - 0s 138us/step - loss: 0.3208 - accuracy:
0.8518 - val_loss: 0.6619 - val_accuracy: 0.7403
Epoch 99/100
614/614 [==============================] - 0s 138us/step - loss: 0.3233 - accuracy:
0.8632 - val_loss: 0.6830 - val_accuracy: 0.7468
Epoch 100/100
```

```
614/614 [==============================] - 0s 135us/step - loss: 0.3230 - accuracy:
0.8567 - val_loss: 0.6732 - val_accuracy: 0.7403
```

Evaluate Model

In [20]:

```
# Evaluate the model
_, accuracy = model.evaluate(X_test, y_test)
print('Accuracy: %.2f' % (accuracy*100))
```

```
154/154 [==============================] - 0s 68us/step
Accuracy: 74.03
```

In [21]:

```
# cross-validation model
print(" Cross-Validation Accuracy Training: ", _, accuracy )
#print(" Mean Cross-Validation Accuracy Evaluation: ", _, accuracy.mean() )
```

```
 Cross-Validation Accuracy Training:  0.6731921744036984 0.7402597665786743
```

In [ ]:

```

```