Neural Networks & Deep Learning Assignment-9

Harshavardhan Reddy Pattepur
700754833

Repository Link :

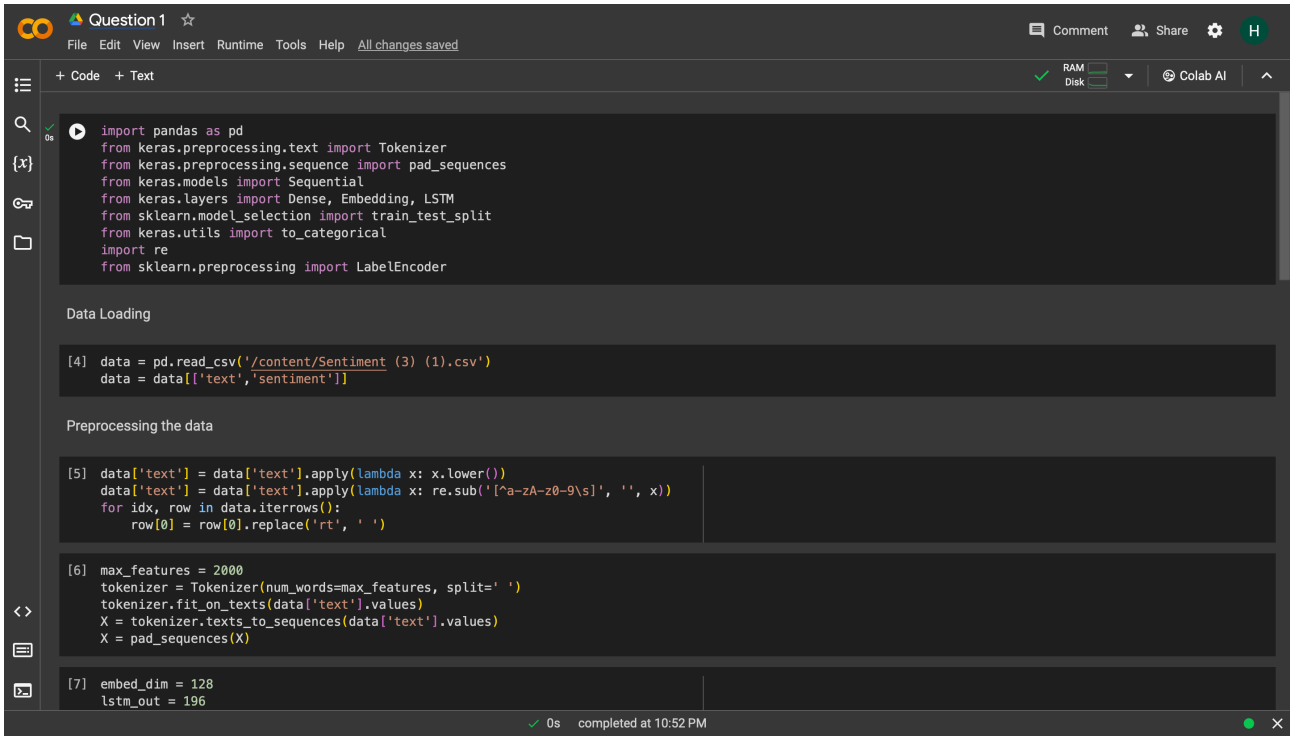
https://github.com/harshavardhanreddy27/NNDL-Assignment---9

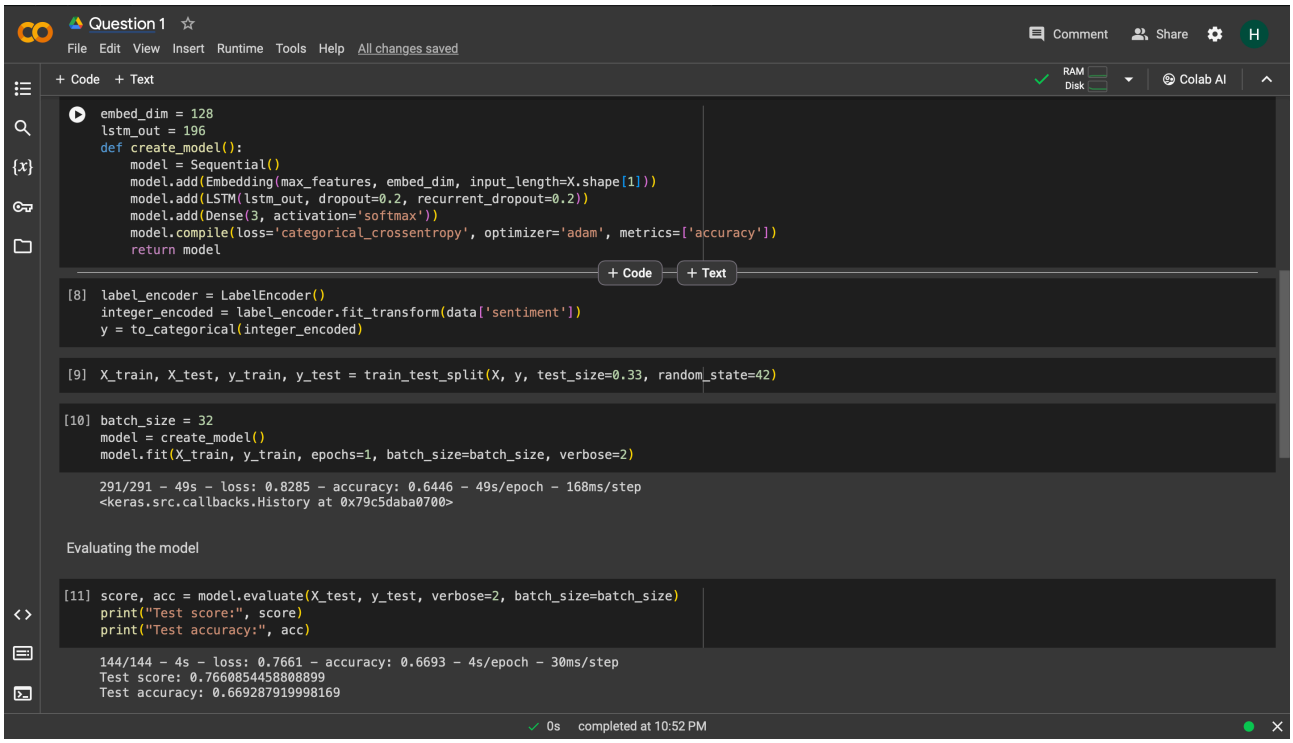
Video Link:

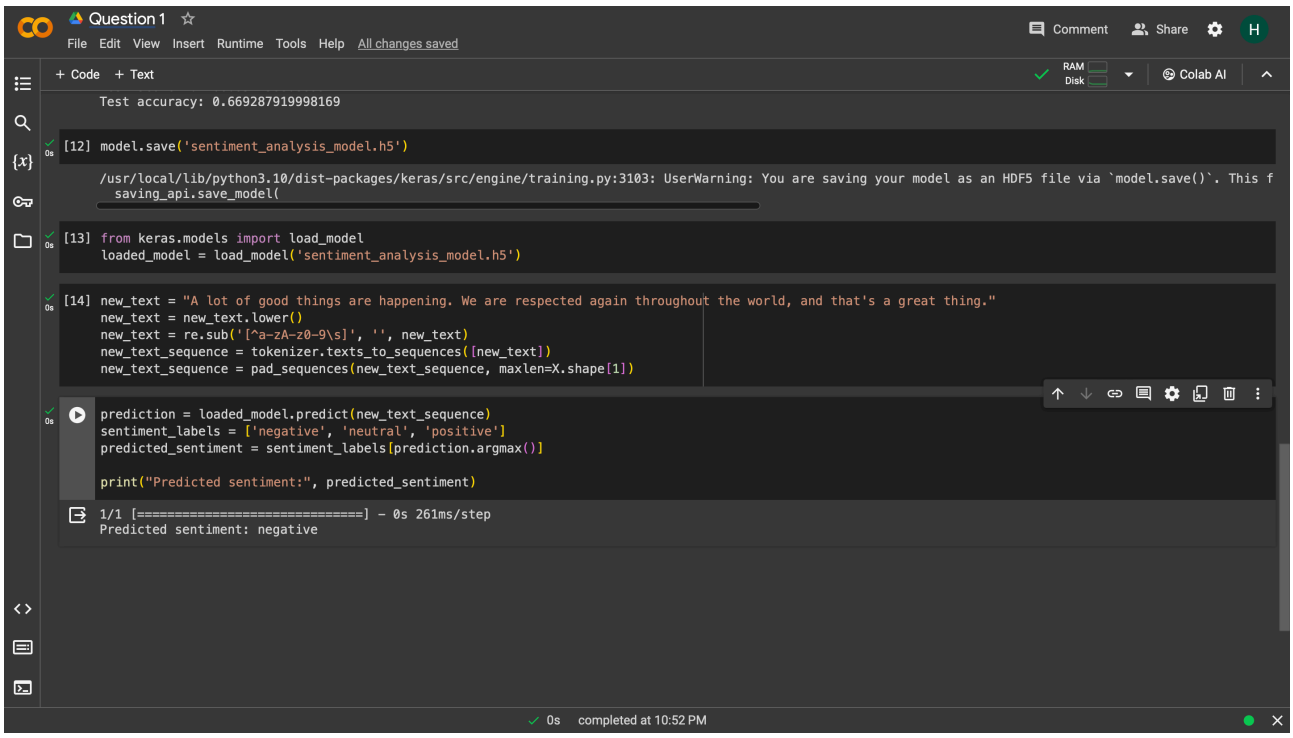
https://drive.google.com/file/d/1FMAO2jQIRUJejgt7kiUK5YeDmp3cEYr0/view?usp=share_link

Code Screenshots:

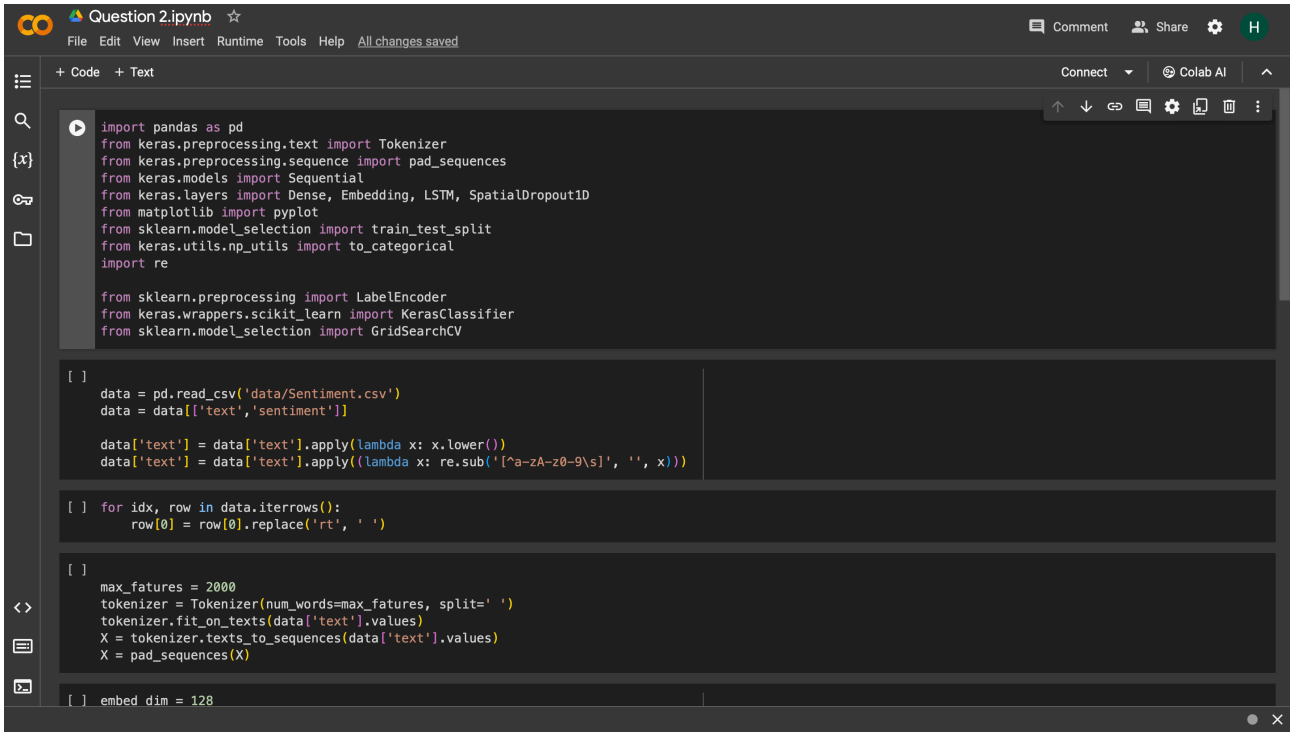
Question 1:-




Question 1
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
H
+ Code + Text
RAM
Disk
Colab AI
embed_dim = 128
lstm_out = 196
def create_model():
model = Sequential()
model.add(Embedding(max_features, embed_dim, input_length=X.shape[1]))
model.add(LSTM(lstm_out, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(3, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
return model
+ Code + Text
[8] label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(data['sentiment'])
y = to_categorical(integer_encoded)
[9] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
[10] batch_size = 32
model = create_model()
model.fit(X_train, y_train, epochs=1, batch_size=batch_size, verbose=2)
291/291 - 49s - loss: 0.8285 - accuracy: 0.6446 - 49s/epoch - 168ms/step
<keras.src.callbacks.History at 0x79c5daba0700>
Evaluating the model
[11] score, acc = model.evaluate(X_test, y_test, verbose=2, batch_size=batch_size)
print("Test score:", score)
print("Test accuracy:", acc)
144/144 - 4s - loss: 0.7661 - accuracy: 0.6693 - 4s/epoch - 30ms/step
Test score: 0.7660854458808899
Test accuracy: 0.669287919998169
0s completed at 10:52 PM


Question 1
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
H
+ Code + Text
RAM
Disk
Colab AI
Test accuracy: 0.669287919998169
[12] model.save('sentiment_analysis_model.h5')
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This f
saving_api.save_model(
[13] from keras.models import load_model
loaded_model = load_model('sentiment_analysis_model.h5')
[14] new_text = "A lot of good things are happening. We are respected again throughout the world, and that's a great thing."
new_text = new_text.lower()
new_text = re.sub('[^a-zA-z0-9\s]', '', new_text)
new_text_sequence = tokenizer.texts_to_sequences([new_text])
new_text_sequence = pad_sequences(new_text_sequence, maxlen=X.shape[1])
prediction = loaded_model.predict(new_text_sequence)
sentiment_labels = ['negative', 'neutral', 'positive']
predicted_sentiment = sentiment_labels[prediction.argmax()]
print("Predicted sentiment:", predicted_sentiment)
1/1 [==============================] - 0s 261ms/step
Predicted sentiment: negative
0s completed at 10:52 PM

Question 2:-



Question 2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share H

+ Code + Text

Connect Colab AI

```
import pandas as pd
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D
from matplotlib import pyplot
from sklearn.model_selection import train_test_split
from keras.utils.np_utils import to_categorical
import re

from sklearn.preprocessing import LabelEncoder
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import GridSearchCV
```

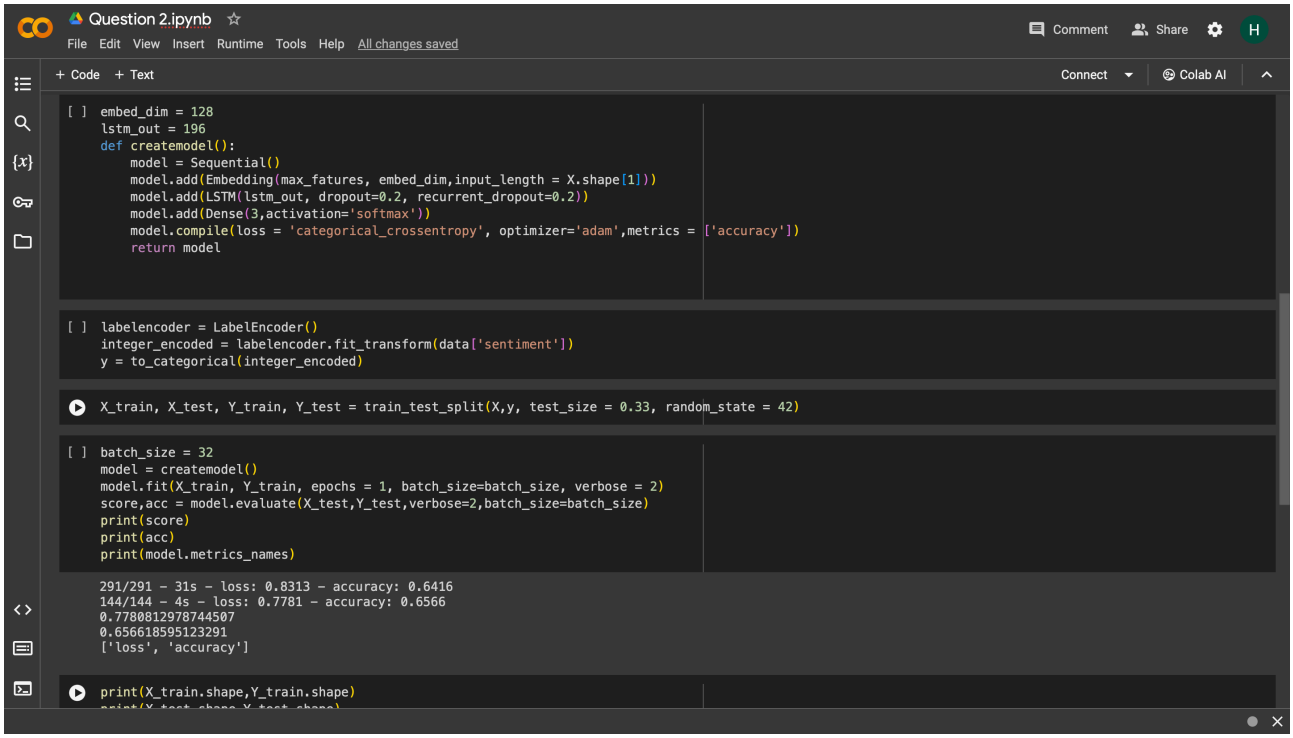
```
[ ]
    data = pd.read_csv('data/Sentiment.csv')
    data = data[['text','sentiment']]

    data['text'] = data['text'].apply(lambda x: x.lower())
    data['text'] = data['text'].apply((lambda x: re.sub('[^a-zA-z0-9\s]', '', x)))
```

```
[ ] for idx, row in data.iterrows():
        row[0] = row[0].replace('rt', ' ')
```

```
[ ]
    max_fatures = 2000
    tokenizer = Tokenizer(num_words=max_fatures, split=' ')
    tokenizer.fit_on_texts(data['text'].values)
    X = tokenizer.texts_to_sequences(data['text'].values)
    X = pad_sequences(X)
```

```
[ ] embed_dim = 128
```



Question 2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share H

+ Code + Text

Connect Colab AI

```
[ ] embed_dim = 128
    lstm_out = 196
    def createmodel():
        model = Sequential()
        model.add(Embedding(max_fatures, embed_dim,input_length = X.shape[1]))
        model.add(LSTM(lstm_out, dropout=0.2, recurrent_dropout=0.2))
        model.add(Dense(3,activation='softmax'))
        model.compile(loss = 'categorical_crossentropy', optimizer='adam',metrics = ['accuracy'])
        return model
```

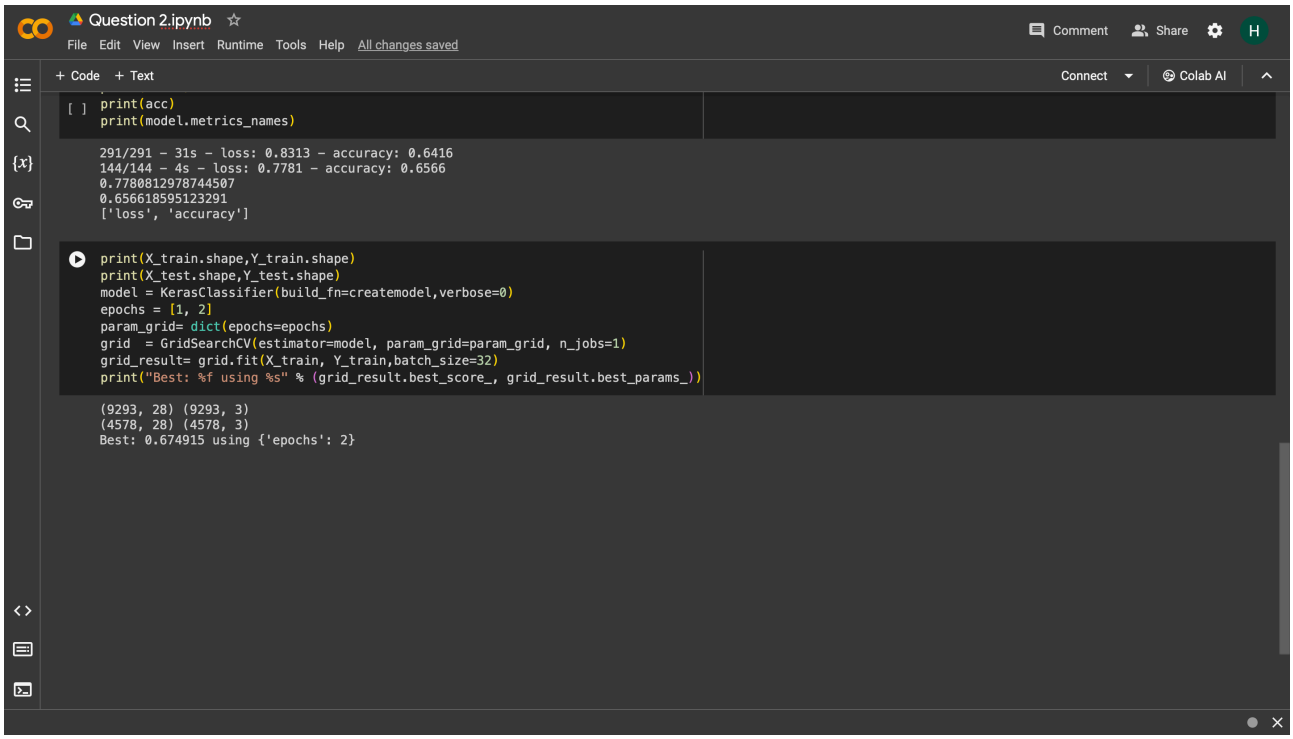
```
[ ] labelencoder = LabelEncoder()
    integer_encoded = labelencoder.fit_transform(data['sentiment'])
    y = to_categorical(integer_encoded)
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X,y, test_size = 0.33, random_state = 42)
```

```
[ ] batch_size = 32
    model = createmodel()
    model.fit(X_train, Y_train, epochs = 1, batch_size=batch_size, verbose = 2)
    score,acc = model.evaluate(X_test,Y_test,verbose=2,batch_size=batch_size)
    print(score)
    print(acc)
    print(model.metrics_names)
```

```
291/291 - 31s - loss: 0.8313 - accuracy: 0.6416
144/144 - 4s - loss: 0.7781 - accuracy: 0.6566
0.7780812978744507
0.656618595123291
['loss', 'accuracy']
```

```
print(X_train.shape,Y_train.shape)
```


Question 2.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
H
+ Code
+ Text
Connect
Colab AI
[ ]
print(acc)
print(model.metrics_names)
291/291 - 31s - loss: 0.8313 - accuracy: 0.6416
144/144 - 4s - loss: 0.7781 - accuracy: 0.6566
0.7780812978744507
0.656618595123291
['loss', 'accuracy']
print(X_train.shape,Y_train.shape)
print(X_test.shape,Y_test.shape)
model = KerasClassifier(build_fn=createmodel,verbose=0)
epochs = [1, 2]
param_grid= dict(epochs=epochs)
grid = GridSearchCV(estimator=model, param_grid=param_grid, n_jobs=1)
grid_result= grid.fit(X_train, Y_train,batch_size=32)
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
(9293, 28) (9293, 3)
(4578, 28) (4578, 3)
Best: 0.674915 using {'epochs': 2}