

ANN

preprocessing images:

First I started trying reading data
by function "create_training_data_gray"
and started normalization steps
by covert the images to gray scale and
resize it (64,64)
then I shuffled that data to make it mixed
x is features and Y is the final result
and did more of preprocessing by converting
X and Y to float32
and in X I divided each by 255

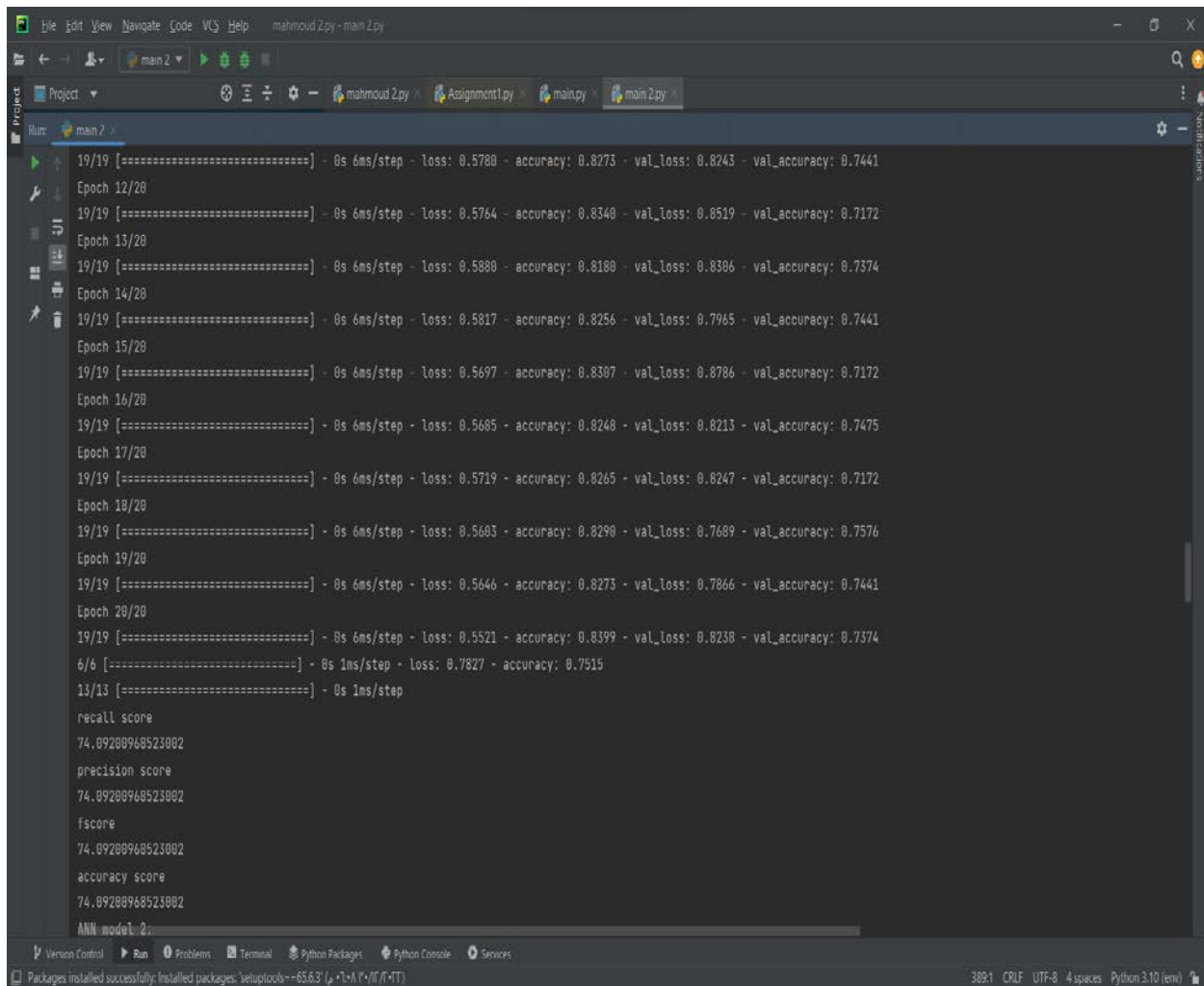
Splitting data:

reshape X splitting data to 80:20
and covert both y train and test to binary class matrices

Calling models:

and did two models with different numbers of layers and nodes
and then applied K fold in both models

1- first model results which is more accurate and more complex
mainly goes from 71:75%
recall score=74.09200968523002
precision score = 74.09200968523002
fscore = 74.09200968523002
accuracy score = 74.09200968523002



```
19/19 [=====] - 0s 6ms/step - loss: 0.5780 - accuracy: 0.8273 - val_loss: 0.8243 - val_accuracy: 0.7441
Epoch 12/20
19/19 [=====] - 0s 6ms/step - loss: 0.5764 - accuracy: 0.8348 - val_loss: 0.8519 - val_accuracy: 0.7172
Epoch 13/20
19/19 [=====] - 0s 6ms/step - loss: 0.5880 - accuracy: 0.8180 - val_loss: 0.8306 - val_accuracy: 0.7374
Epoch 14/20
19/19 [=====] - 0s 6ms/step - loss: 0.5817 - accuracy: 0.8256 - val_loss: 0.7965 - val_accuracy: 0.7441
Epoch 15/20
19/19 [=====] - 0s 6ms/step - loss: 0.5697 - accuracy: 0.8307 - val_loss: 0.8786 - val_accuracy: 0.7172
Epoch 16/20
19/19 [=====] - 0s 6ms/step - loss: 0.5685 - accuracy: 0.8248 - val_loss: 0.8213 - val_accuracy: 0.7475
Epoch 17/20
19/19 [=====] - 0s 6ms/step - loss: 0.5719 - accuracy: 0.8265 - val_loss: 0.8247 - val_accuracy: 0.7172
Epoch 18/20
19/19 [=====] - 0s 6ms/step - loss: 0.5683 - accuracy: 0.8290 - val_loss: 0.7689 - val_accuracy: 0.7576
Epoch 19/20
19/19 [=====] - 0s 6ms/step - loss: 0.5646 - accuracy: 0.8273 - val_loss: 0.7866 - val_accuracy: 0.7441
Epoch 20/20
19/19 [=====] - 0s 6ms/step - loss: 0.5521 - accuracy: 0.8399 - val_loss: 0.8238 - val_accuracy: 0.7374
6/6 [=====] - 0s 1ms/step - loss: 0.7827 - accuracy: 0.7515
13/13 [=====] - 0s 1ms/step
recall score
74.09200968523002
precision score
74.09200968523002
fscore
74.09200968523002
accuracy score
74.09200968523002
ANN model: 2:
```

2- second model results which is less accurate but still in a good range which is mainly goes from 61:68 but it is still

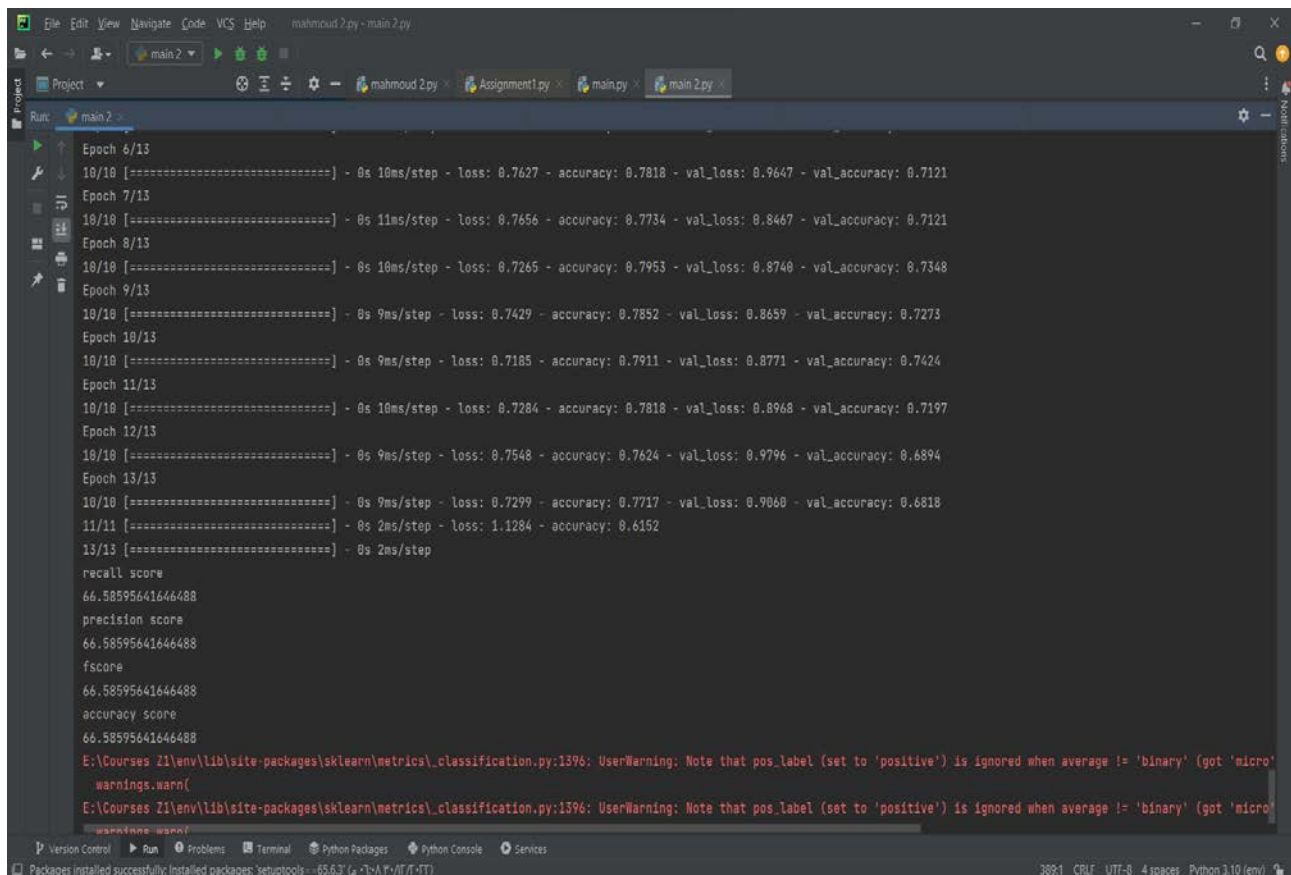
more simpler than the first one

recall score = 66.58595641646488

precision score = 66.58595641646488

fscore = 66.58595641646488

accuracy score = 66.58595641646488



```
Epoch 6/13
10/10 [=====] - 0s 10ms/step - loss: 0.7627 - accuracy: 0.7818 - val_loss: 0.9647 - val_accuracy: 0.7121
Epoch 7/13
10/10 [=====] - 0s 11ms/step - loss: 0.7656 - accuracy: 0.7734 - val_loss: 0.8467 - val_accuracy: 0.7121
Epoch 8/13
10/10 [=====] - 0s 10ms/step - loss: 0.7265 - accuracy: 0.7953 - val_loss: 0.8748 - val_accuracy: 0.7348
Epoch 9/13
10/10 [=====] - 0s 9ms/step - loss: 0.7429 - accuracy: 0.7852 - val_loss: 0.8659 - val_accuracy: 0.7273
Epoch 10/13
10/10 [=====] - 0s 9ms/step - loss: 0.7185 - accuracy: 0.7911 - val_loss: 0.8771 - val_accuracy: 0.7424
Epoch 11/13
10/10 [=====] - 0s 10ms/step - loss: 0.7284 - accuracy: 0.7818 - val_loss: 0.8968 - val_accuracy: 0.7197
Epoch 12/13
10/10 [=====] - 0s 9ms/step - loss: 0.7548 - accuracy: 0.7624 - val_loss: 0.9796 - val_accuracy: 0.6894
Epoch 13/13
10/10 [=====] - 0s 9ms/step - loss: 0.7299 - accuracy: 0.7717 - val_loss: 0.9068 - val_accuracy: 0.6818
11/11 [=====] - 0s 2ms/step - loss: 1.1284 - accuracy: 0.6152
13/13 [=====] - 0s 2ms/step
recall score
66.58595641646488
precision score
66.58595641646488
fscore
66.58595641646488
accuracy score
66.58595641646488
E:\Courses 21\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
E:\Courses 21\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
Version Control | Run | Problems | Terminal | Python Packages | Python Console | Services
Packages installed successfully: Installed packages: setuptools -- 65.6.3 (Python 3.10 (env))
```

CNN & SVM

CNN model:

preprocessing images:

We preprocessing images by calculate average for each pixel in each image (using stat fun) then convert it to numpy array and subtract average then divide it by 255 add image after preprocessing to list with it's class

Splitting data:

Split data to X and Y then using K-fold concept split data to train and test and validation and call the model

Split was training 80, test 20 and validation 20% of training data

Calling CNN model:

from train and test model in data we see that when we add more conv layers

and pooling layers accuracy increased more than we just add more hidden layers

as before increasing conv and pooling layers accuracy was in range between 83% and 88%

this run accuracies were:

recall score = 88.61985472154964

precision score = 88.61985472154964

fscore = 88.61985472154964

accuracy score = 88.61985472154964

```
File Edit View Navigate Code VCS Help mahmoud 2.py - main 2.py
main 2
Run: main 2
11/11 [=====] - 0s 33ms/step - loss: 0.7029 - accuracy: 0.7848
At k_fold 2:
Epoch 1/3
17/17 [=====] - 6s 336ms/step - loss: 0.2917 - accuracy: 0.9128 - val_loss: 0.6081 - val_accuracy: 0.8258
Epoch 2/3
17/17 [=====] - 6s 383ms/step - loss: 0.1471 - accuracy: 0.9630 - val_loss: 0.6370 - val_accuracy: 0.8182
Epoch 3/3
17/17 [=====] - 7s 390ms/step - loss: 0.0971 - accuracy: 0.9754 - val_loss: 0.8542 - val_accuracy: 0.7841
11/11 [=====] - 0s 42ms/step - loss: 0.9018 - accuracy: 0.7788
At k_fold 3:
Epoch 1/3
17/17 [=====] - 6s 369ms/step - loss: 0.0557 - accuracy: 0.9886 - val_loss: 0.6068 - val_accuracy: 0.8523
Epoch 2/3
17/17 [=====] - 6s 347ms/step - loss: 0.0231 - accuracy: 0.9962 - val_loss: 0.8185 - val_accuracy: 0.8106
Epoch 3/3
17/17 [=====] - 6s 335ms/step - loss: 0.0295 - accuracy: 0.9905 - val_loss: 0.7933 - val_accuracy: 0.8523
11/11 [=====] - 0s 36ms/step - loss: 1.0179 - accuracy: 0.7879
At k_fold 4:
Epoch 1/3
17/17 [=====] - 6s 347ms/step - loss: 0.1512 - accuracy: 0.9545 - val_loss: 0.8554 - val_accuracy: 0.7803
Epoch 2/3
17/17 [=====] - 6s 339ms/step - loss: 0.0813 - accuracy: 0.9810 - val_loss: 0.7244 - val_accuracy: 0.8106
Epoch 3/3
17/17 [=====] - 6s 341ms/step - loss: 0.0235 - accuracy: 0.9972 - val_loss: 0.7669 - val_accuracy: 0.8371
11/11 [=====] - 0s 36ms/step - loss: 0.8588 - accuracy: 0.8242
At k_fold 5:
Epoch 1/3
17/17 [=====] - 6s 341ms/step - loss: 0.0082 - accuracy: 1.0000 - val_loss: 0.8406 - val_accuracy: 0.8371
Epoch 2/3
17/17 [=====] - 6s 339ms/step - loss: 0.0025 - accuracy: 1.0000 - val_loss: 0.8201 - val_accuracy: 0.8485
Epoch 3/3
17/17 [=====] - 6s 361ms/step - loss: 0.0014 - accuracy: 1.0000 - val_loss: 0.8466 - val_accuracy: 0.8523
11/11 [=====] - 0s 40ms/step - loss: 0.9724 - accuracy: 0.8242
13/13 [=====] - 1s 42ms/step
E:\Courses Z1\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
E:\Courses Z1\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
E:\Courses Z1\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
recall score
88.61985472154964
precision score
88.61985472154964
fscore
88.61985472154964
accuracy score
88.61985472154964

Process finished with exit code 0
Version Control Run Problems Terminal Python Packages Python Console Services
Packages installed successfully: Installed packages: 'setuptools==65.6.3' (python 3.10 (env))
```

```
File Edit View Navigate Code VCS Help mahmoud 2.py - main 2.py
main 2
Run: main 2
Epoch 3/3
17/17 [=====] - 6s 341ms/step - loss: 0.0235 - accuracy: 0.9972 - val_loss: 0.7669 - val_accuracy: 0.8371
11/11 [=====] - 0s 36ms/step - loss: 0.8588 - accuracy: 0.8242
At k_fold 5:
Epoch 1/3
17/17 [=====] - 6s 341ms/step - loss: 0.0082 - accuracy: 1.0000 - val_loss: 0.8406 - val_accuracy: 0.8371
Epoch 2/3
17/17 [=====] - 6s 339ms/step - loss: 0.0025 - accuracy: 1.0000 - val_loss: 0.8201 - val_accuracy: 0.8485
Epoch 3/3
17/17 [=====] - 6s 361ms/step - loss: 0.0014 - accuracy: 1.0000 - val_loss: 0.8466 - val_accuracy: 0.8523
11/11 [=====] - 0s 40ms/step - loss: 0.9724 - accuracy: 0.8242
13/13 [=====] - 1s 42ms/step
E:\Courses Z1\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
E:\Courses Z1\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
E:\Courses Z1\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
recall score
88.61985472154964
precision score
88.61985472154964
fscore
88.61985472154964
accuracy score
88.61985472154964

Process finished with exit code 0
Version Control Run Problems Terminal Python Packages Python Console Services
Packages installed successfully: Installed packages: 'setuptools==65.6.3' (python 3.10 (env))
```

but after increasing them accuracy became in range between 93% and 97%

SVM model:

Using all steps of preprocessing in CNN then calling SVM model

when we apply SVM model we see that accuracy of SVM is very low according to CNN

as accuracy of SVM in range between 44% and 53% and accuracy of CNN is higher than 90%

this run accuracies were:

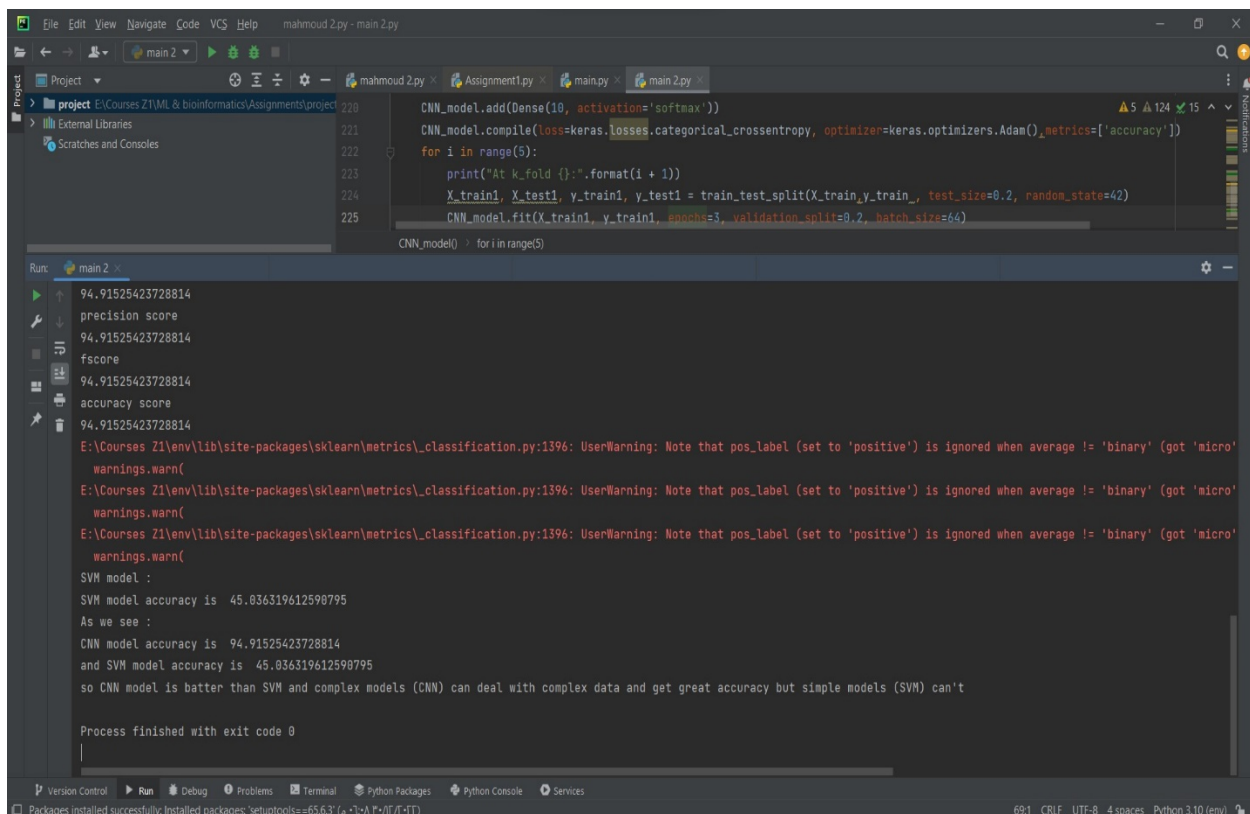
CNN recall score = 94.91525423728814

CNN precision score = 94.91525423728814

CNN fscore = 94.91525423728814

CNN accuracy score = 94.91525423728814

SVM accuracy score = 45.036319612590795



```
File Edit View Navigate Code VCS Help mahmoud 2.py - main 2.py
main 2.py
Project
> project E:\Courses Z1\ML & bioinformatics\Assignments\project
> External Libraries
> Scratches and Consoles
220 CNN_model.add(Dense(10, activation='softmax'))
221 CNN_model.compile(loss=keras.losses.categorical_crossentropy, optimizer=keras.optimizers.Adam(), metrics=['accuracy'])
222 for i in range(5):
223     print("At k-fold {}: ".format(i + 1))
224     X_train1, X_test1, y_train1, y_test1 = train_test_split(X_train, y_train, test_size=0.2, random_state=42)
225     CNN_model.fit(X_train1, y_train1, epochs=3, validation_split=0.2, batch_size=64)
CNN_model() for i in range(5)

Run: main 2
94.91525423728814
precision score
94.91525423728814
fscore
94.91525423728814
accuracy score
94.91525423728814
E:\Courses Z1\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
E:\Courses Z1\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
E:\Courses Z1\env\lib\site-packages\sklearn\metrics\_classification.py:1396: UserWarning: Note that pos_label (set to 'positive') is ignored when average != 'binary' (got 'micro')
warnings.warn(
SVM model :
SVM model accuracy is 45.036319612590795
As we see :
CNN model accuracy is 94.91525423728814
and SVM model accuracy is 45.036319612590795
so CNN model is better than SVM and complex models (CNN) can deal with complex data and get great accuracy but simple models (SVM) can't

Process finished with exit code 0
Version Control Run Debug Problems Terminal Python Packages Python Console Services
Packages installed successfully: Installed packages: 'setuptools==65.6.3' (-1.0.0\IT\J-TT)
691 CRLF UTF-8 4 spaces Python 3.10 (env)
```