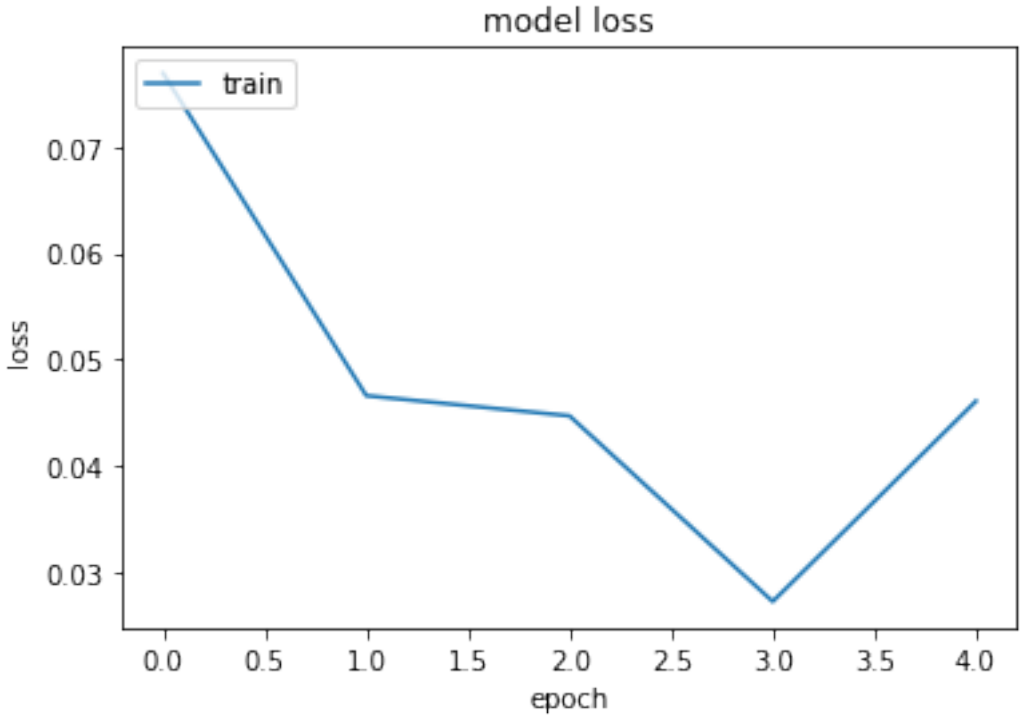
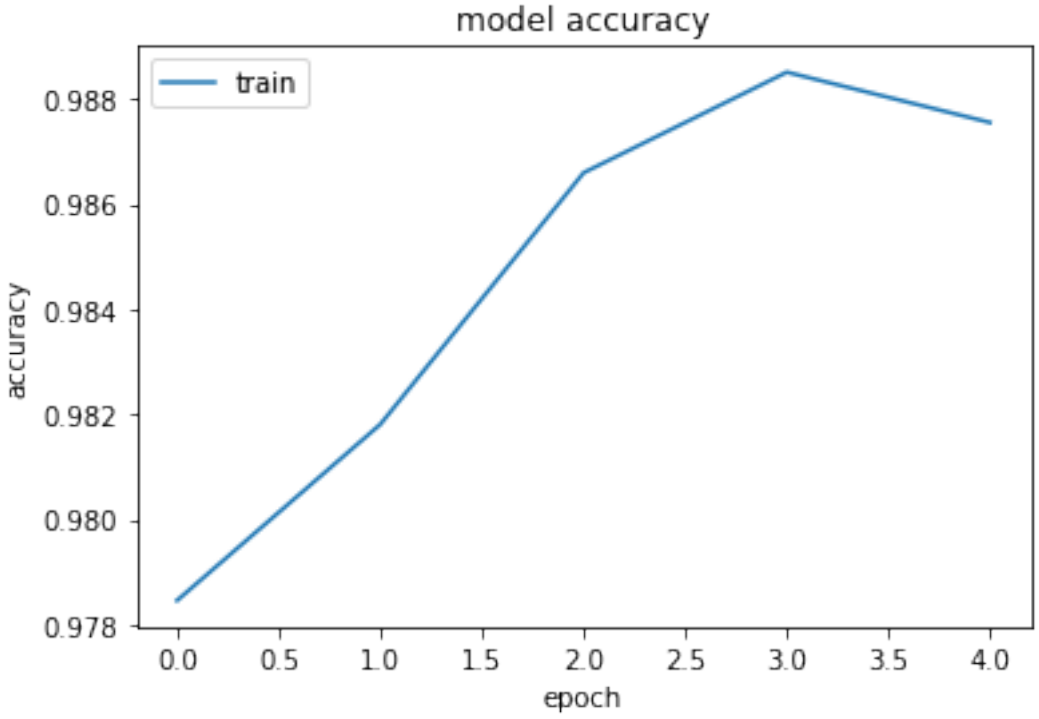
BERT:

```
23/23 [==============================] - 6s 224ms/step - loss:
0.8759 - accuracy: 0.8252
[0.8759374618530273, 0.8252426981925964]
The loss of the model on the test set : 0.8759374618530273
The accuracy of the model on the test set : 82.52426981925964%
```





```
[45] new_input = ["i have a big butt"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[1.8674208e-06]]
```

```
[47] new_input = ["I fantasise about big butts at work"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[0.99968445]]
```

```
     new_input = ["I want to see your big boobies"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[0.9999051]]
```

```
[50] new_input = ["boobies is an offensive word"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[2.1722178e-06]]
```

```
[42] new_input = ["lick my hairy balls"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[0.99990785]]
```

```
[37] new_input = ["i wanna eat your ass"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[0.9999126]]
```

```
     new_input = ["bite me in the ass"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[0.9996673]]
```

```
     new_input = ["i am an ass"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[2.245538e-06]]
```

```
     new_input = ["i am a hairy person and i like playing with tennis balls"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[8.212096e-05]]
```

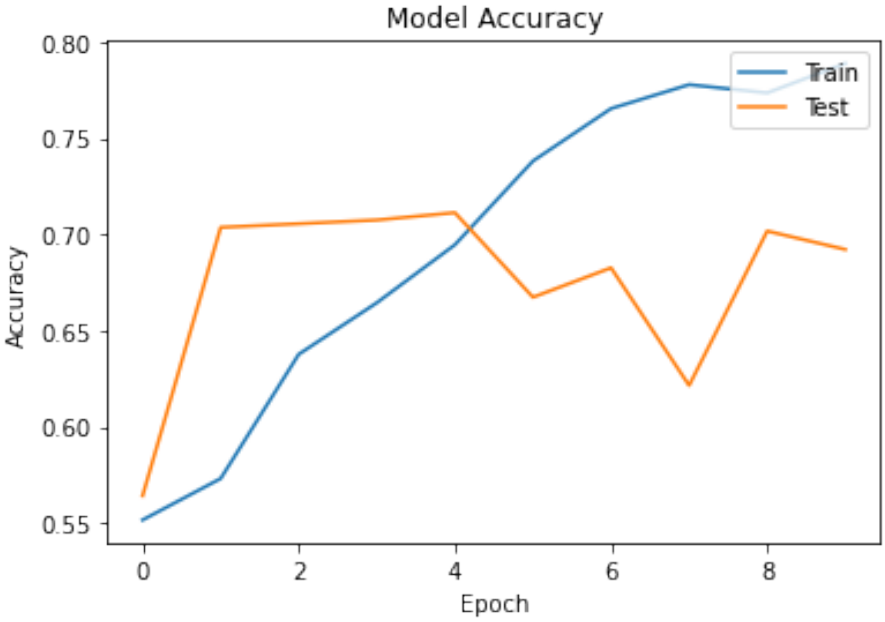
```
[39] new_input = ["a donkey can also be called an ass"]

     new_val = bert_encode(new_input, tokenizer, max_len=max_len)

     print(model.predict(new_val))

     [[0.00032227]]
```

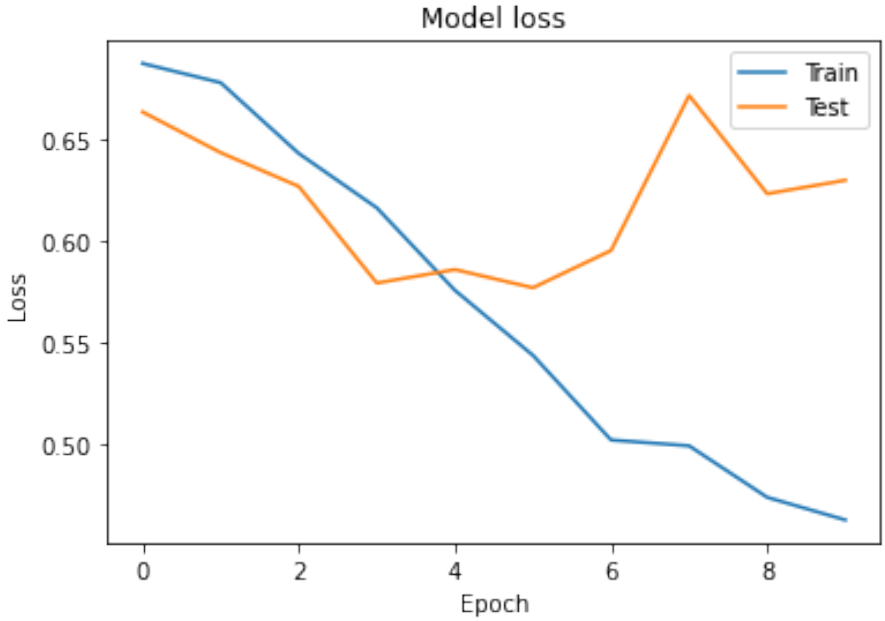
CNN-BiLSTM:

```
[89] test_performance=model.evaluate(x_test,y_test)
     print("The loss of the model on the test set : {}".format(test_performance[0]))
     print("The accuracy of the model on the test set : {}%".format(test_performance[1]*100))
```

```
17/17 [==============================] - 0s 14ms/step - loss: 0.6295 - accuracy: 0.6922
The loss of the model on the test set : 0.6294759511947632
The accuracy of the model on the test set : 69.21606063842773%
```

```
Model: "sequential_11"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_7 (Embedding)      (None, 300, 300)          6000000
_________________________________________________________________
conv1d_11 (Conv1D)           (None, 298, 32)           28832
_________________________________________________________________
max_pooling1d_11 (MaxPooling (None, 149, 32)           0
_________________________________________________________________
dropout_22 (Dropout)         (None, 149, 32)           0
_________________________________________________________________
bidirectional_11 (Bidirectio (None, 149, 256)          164864
_________________________________________________________________
flatten_11 (Flatten)         (None, 38144)             0
_________________________________________________________________
dense_22 (Dense)             (None, 30)                1144350
_________________________________________________________________
dropout_23 (Dropout)         (None, 30)                0
_________________________________________________________________
dense_23 (Dense)             (None, 2)                 62
=================================================================
Total params: 7,338,108
Trainable params: 1,338,108
Non-trainable params: 6,000,000
_________________________________________________________________
```





ML CLASSIFIERS:

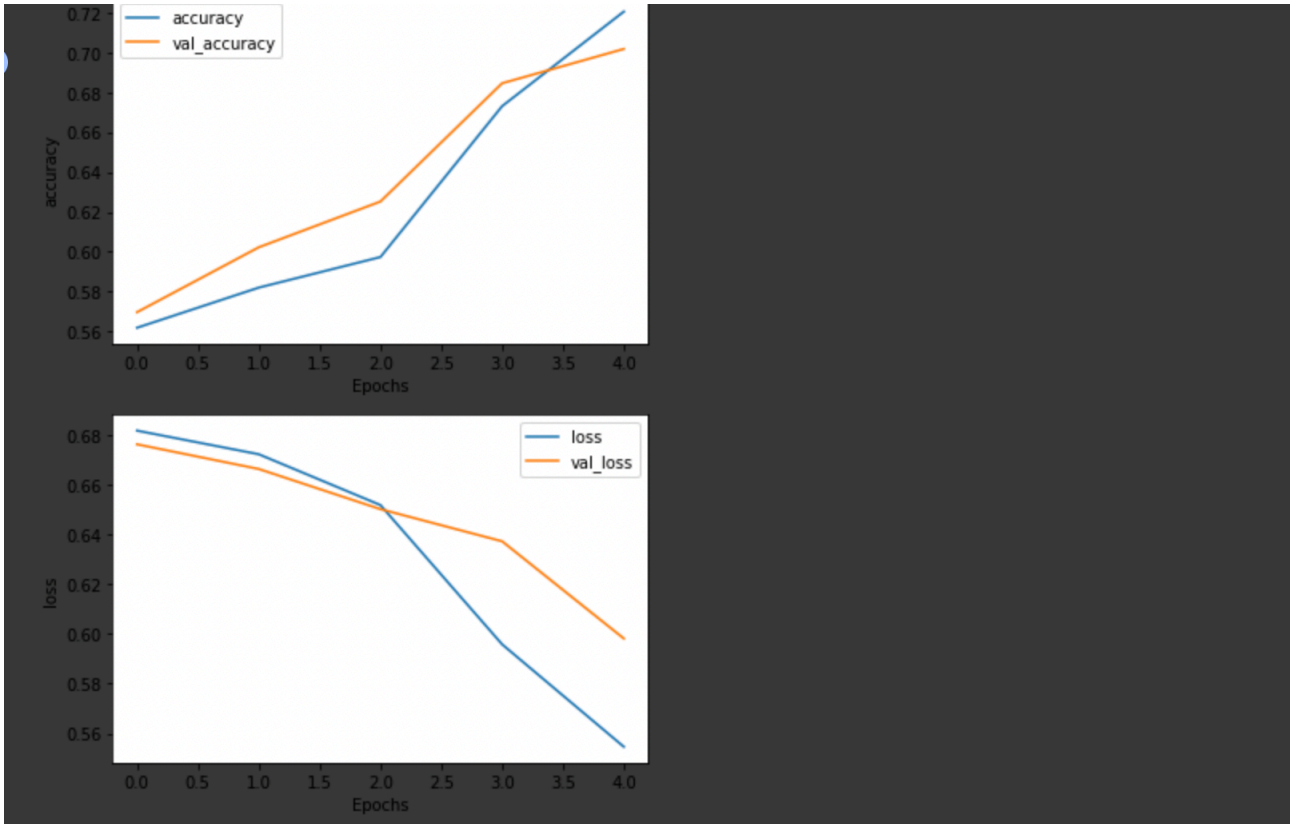
Linear SVC:
0.9784585926280517
0.7571701720841301

Gaussian NB:
0.8879846816658689
0.6347992351816444

Logistic Regression:
0.8736237434179033
0.7571701720841301

Random Forest:
0.9784585926280517
0.7112810707456979

BILSTM:




```
test_performance=model.evaluate(X_test_Glove,y_test)
print((test_performance))
print("The loss of the model on the test set : {}".format(test_performance[0]))
print("The accuracy of the model on the test set : {}%".format(test_performance[1]*100))
```

```
17/17 [==============================] - 7s 391ms/step - loss: 0.5981 - accuracy: 0.7017
[0.5981393456459045, 0.7017208337783813]
The loss of the model on the test set : 0.5981393456459045
The accuracy of the model on the test set : 70.17208337783813%
```

Accuracy decreased from 71% to 67% on removing the stop words from which we can conclude that stop words have valuable info for our task.