```
import pandas as pd
import numpy as np
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense, SpatialDropout1D
from sklearn.model_selection import train_test_split


train_df = pd.read_csv('train.csv')
test_df = pd.read_csv('test.csv')


print(train_df.head())
print(test_df.head())
```

```
   id keyword location                                          text  \
0   1     NaN      NaN  Our Deeds are the Reason of this #earthquake M...
1   4     NaN      NaN             Forest fire near La Ronge Sask. Canada
2   5     NaN      NaN  All residents asked to 'shelter in place' are ...
3   6     NaN      NaN  13,000 people receive #wildfires evacuation or...
4   7     NaN      NaN  Just got sent this photo from Ruby #Alaska as ...

   target
0       1
1       1
2       1
3       1
4       1
   id keyword location                                          text
0   0     NaN      NaN                 Just happened a terrible car crash
1   2     NaN      NaN  Heard about #earthquake is different cities, s...
2   3     NaN      NaN  there is a forest fire at spot pond, geese are...
3   9     NaN      NaN           Apocalypse lighting. #Spokane #wildfires
4  11     NaN      NaN      Typhoon Soudelor kills 28 in China and Taiwan
```

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(train_df['text'])


X = tokenizer.texts_to_sequences(train_df['text'])
X = pad_sequences(X, maxlen=50)


y = train_df['target']


X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)


model = Sequential()
model.add(Embedding(len(tokenizer.word_index) + 1, 32, input_length=50))
model.add(SpatialDropout1D(0.2))
model.add(LSTM(64, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(1, activation='sigmoid'))


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


print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 50, 32)            726432

 spatial_dropout1d (Spatial  (None, 50, 32)            0
 Dropout1D)

 lstm (LSTM)                 (None, 64)                24832

 dense (Dense)               (None, 1)                 65

=================================================================
Total params: 751329 (2.87 MB)
Trainable params: 751329 (2.87 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
```

```
history = model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_val, y_val), verbose=1)
```

```
Epoch 1/10
191/191 [==============================] - 24s 94ms/step - loss: 0.5757 - accuracy: 0.6882 - val_loss: 0.4458 - val_accuracy: 0.812
```

```
Epoch 2/10
191/191 [==============================] - 20s 103ms/step - loss: 0.3346 - accuracy: 0.8670 - val_loss: 0.4504 - val_accuracy: 0.80
Epoch 3/10
191/191 [==============================] - 18s 92ms/step - loss: 0.2027 - accuracy: 0.9284 - val_loss: 0.5307 - val_accuracy: 0.799
Epoch 4/10
191/191 [==============================] - 19s 97ms/step - loss: 0.1187 - accuracy: 0.9588 - val_loss: 0.6628 - val_accuracy: 0.774
Epoch 5/10
191/191 [==============================] - 19s 100ms/step - loss: 0.0840 - accuracy: 0.9731 - val_loss: 0.8857 - val_accuracy: 0.74
Epoch 6/10
191/191 [==============================] - 18s 97ms/step - loss: 0.0449 - accuracy: 0.9857 - val_loss: 0.8703 - val_accuracy: 0.741
Epoch 7/10
191/191 [==============================] - 20s 102ms/step - loss: 0.0335 - accuracy: 0.9910 - val_loss: 0.8966 - val_accuracy: 0.75
Epoch 8/10
191/191 [==============================] - 19s 97ms/step - loss: 0.0296 - accuracy: 0.9906 - val_loss: 0.8908 - val_accuracy: 0.747
Epoch 9/10
191/191 [==============================] - 19s 99ms/step - loss: 0.0220 - accuracy: 0.9938 - val_loss: 0.9122 - val_accuracy: 0.753
Epoch 10/10
191/191 [==============================] - 18s 95ms/step - loss: 0.0217 - accuracy: 0.9931 - val_loss: 0.8650 - val_accuracy: 0.726
```

```
loss, accuracy = model.evaluate(X_val, y_val)
print("Validation Accuracy:", accuracy)
```

```
48/48 [==============================] - 1s 29ms/step - loss: 0.8650 - accuracy: 0.7262
Validation Accuracy: 0.7261983156204224
```

```
X_test = tokenizer.texts_to_sequences(test_df['text'])
X_test = pad_sequences(X_test, maxlen=50)
```

```
predictions_proba = model.predict(X_test).flatten()
```

```
102/102 [==============================] - 1s 11ms/step
```

```
predictions = (predictions_proba > 0.5).astype(int)
```

```
submission_df = pd.DataFrame({'id': test_df['id'], 'target': predictions})
submission_df.to_csv('submission.csv', index=False)
```