Import required library

In [18]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from keras.models import Model,load_model,Sequential
from keras.layers import LSTM, Activation, Dense, Dropout, Input, Embedding
from keras.optimizers import RMSprop
from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
%matplotlib inline
```

Read dataset and do pre-processing

In [19]:

```
df = pd.read_csv('spam.csv',delimiter=',',encoding='latin-1')
df.head()
```

Out[19]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

In [20]:

```
df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)
df
```

Out[20]:

	v1	v2
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will Ì_ b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...

	v1	v2
5571	ham	Rofl. Its true to its name

5572 rows × 2 columns

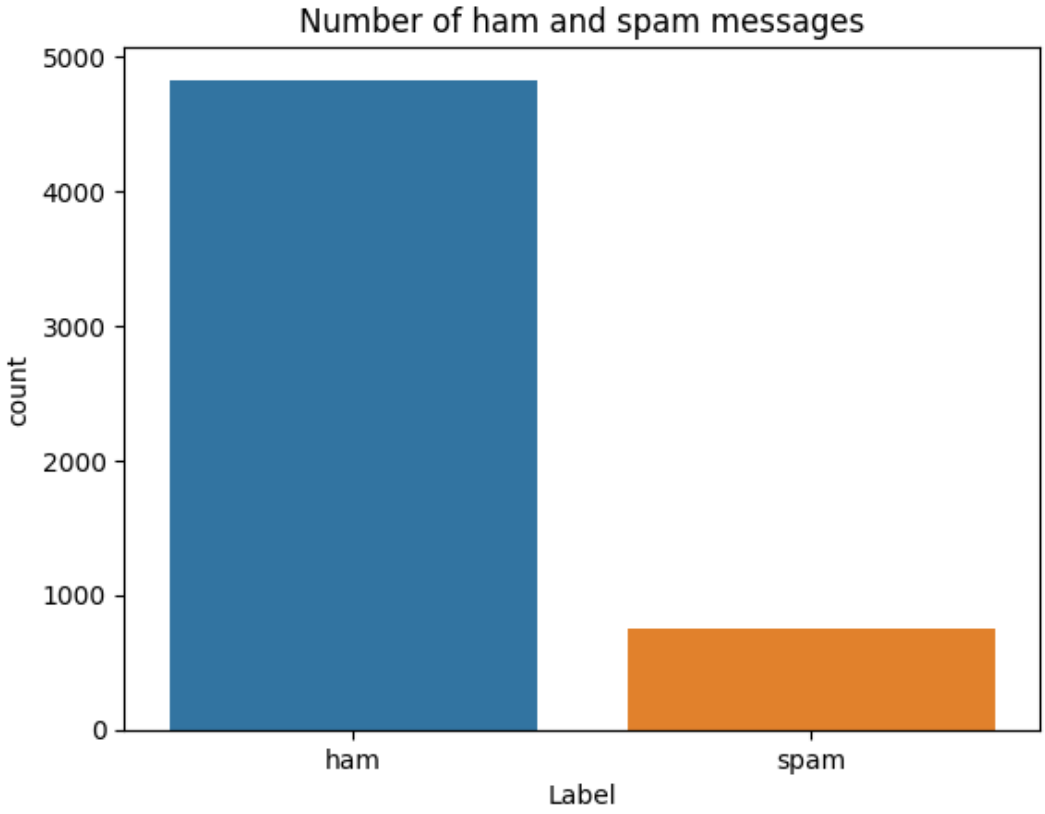
In [21]:

```
sns.countplot(df.v1)
plt.xlabel('Label')
plt.title('Number of ham and spam messages')
```

```
c:\Python3.7\lib\site-packages\seaborn\_decorators.py:43: FutureWarning: Pass the followi
ng variable as a keyword arg: x. From version 0.12, the only valid positional argument wi
ll be `data`, and passing other arguments without an explicit keyword will result in an e
rror or misinterpretation.
  FutureWarning
```

Out[21]:

```
Text(0.5, 1.0, 'Number of ham and spam messages')
```



- **Create input and output vectors.**
- **Process the labels.**

In [22]:

```
X = df.v2
Y = df.v1
le = LabelEncoder()
Y = le.fit_transform(Y)
Y = Y.reshape(-1,1)
```

Split into training and test data.

In [23]:

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)
```

Process the data

- **Tokenize the data and convert the text to sequences.**
- **Add padding to ensure that all the sequences have the same shape.**

- Add padding to ensure that all the sequences have the same shape.
- **There are many ways of taking the *max_len* and here an arbitrary length of 150 is chosen.**

In [24]:

```
max_words = 1000
max_len = 150
tok = Tokenizer(num_words=max_words)
tok.fit_on_texts(X_train)
sequences = tok.texts_to_sequences(X_train)
sequences_matrix = sequence.pad_sequences(sequences,maxlen=max_len)
```

Layers

In [25]:

```
model=Sequential()
model.add(Embedding(max_words,50,input_length=max_len))
model.add(LSTM(64))
model.add(Dense(256,name='FC1'))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(1,name='out_layer'))
model.add(Activation('sigmoid'))
```

Compile the Model

In [26]:

```
model.summary()
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_3 (Embedding)      (None, 150, 50)           50000
_________________________________________________________________
lstm_3 (LSTM)                (None, 64)                29440
_________________________________________________________________
FC1 (Dense)                  (None, 256)               16640
_________________________________________________________________
activation_5 (Activation)    (None, 256)               0
_________________________________________________________________
dropout_3 (Dropout)          (None, 256)               0
_________________________________________________________________
out_layer (Dense)            (None, 1)                 257
_________________________________________________________________
activation_6 (Activation)    (None, 1)                 0
=================================================================
Total params: 96,337
Trainable params: 96,337
Non-trainable params: 0
_________________________________________________________________
```

Fit the Model

In [35]:

```
model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
          validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',min_delta=0.0
001)])
```

```
Train on 3788 samples, validate on 948 samples
Epoch 1/10
3788/3788 [==============================] - 6s 2ms/step - loss: 0.0166 - acc: 0.9963 - v
al_loss: 0.0580 - val_acc: 0.9789
Epoch 2/10
```

```
3788/3788 [==============================] - 5s 1ms/step - loss: 0.0158 - acc: 0.9950 - v
al_loss: 0.0565 - val_acc: 0.9842
Epoch 3/10
3788/3788 [==============================] - 5s 1ms/step - loss: 0.0107 - acc: 0.9974 - v
al_loss: 0.0693 - val_acc: 0.9842
```

Out[35]:

```
<keras.callbacks.History at 0x1453e789a90>
```

Save The Model

In [36]:

```
model.save("assign4.h5")
```

Test The Model

In [37]:

```
model=load_model("assign4.h5")
```

The model performs well on the validation set and this configuration is chosen as the final model.

Process the test set data.

In [38]:

```
test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = sequence.pad_sequences(test_sequences,maxlen=max_len)
```

Evaluate the model on the test set.

In [39]:

```
accr = model.evaluate(test_sequences_matrix,Y_test)
```

```
836/836 [==============================] - 1s 1ms/step
```

In [40]:

```
print('Test set\n  Loss: {:0.3f}\n  Accuracy: {:0.3f}'.format(accr[0],accr[1]))
```

```
Test set
  Loss: 0.071
  Accuracy: 0.986
```