In [11]:

```python
import os
import re
import pandas as pd
import numpy as np
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM, Dropout, Embedding
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.preprocessing.text import Tokenizer
import keras
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
```

In [12]:

```python
spam_df = pd.read_csv(filepath_or_buffer='Spam.csv', delimiter=',',encoding='latin-1')
spam_df.head()
```

Out[12]:

| | 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 [14]:

```python
spam_df.drop(columns=['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], axis=1, inplace=True)
spam_df.describe()
```

Out[14]:

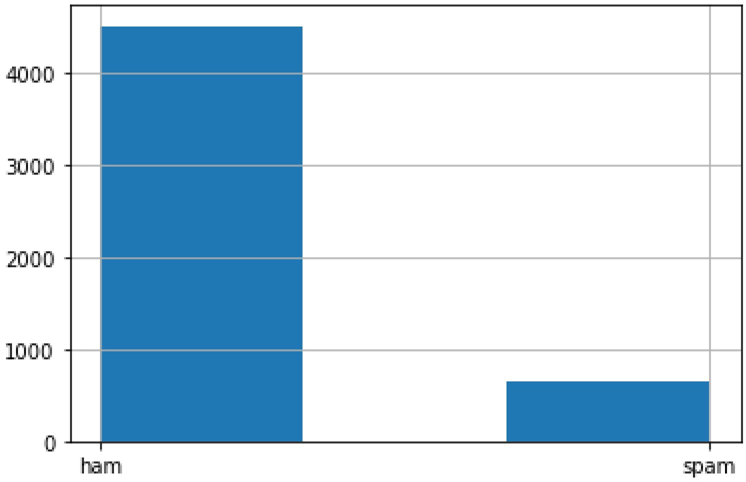
| | v1 | v2 |
|---|---|---|
| **count** | 5572 | 5572 |
| **unique** | 2 | 5169 |
| **top** | ham | Sorry, I'll call later |
| **freq** | 4825 | 30 |

In [15]:

```
spam_df.isna().sum()
spam_df.duplicated().sum()
spam_df = spam_df.drop_duplicates()
spam_df.duplicated().sum()
spam_df['v1'].hist(bins=3)
```

Out[15]:

```
<AxesSubplot:>
```



In [17]:

```
spam_df['alpha_text'] = spam_df['v2'].apply(lambda x: re.sub(r'[^a-zA-Z ]+', '', x.lowe
r()))
spam_df.head()
```

Out[17]:

| | v1 | v2 | alpha_text |
|---|---|---|---|
| **0** | ham | Go until jurong point, crazy.. Available only ... | go until jurong point crazy available only in ... |
| **1** | ham | Ok lar... Joking wif u oni... | ok lar joking wif u oni |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | free entry in a wkly comp to win fa cup final... |
| **3** | ham | U dun say so early hor... U c already then say... | 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... | nah i dont think he goes to usf he lives aroun... |

In [18]:

```
nltk.download('stopwords')
spam_df['imp_text'] = spam_df['alpha_text'].apply(lambda x : ' '.join([word for word in
x.split() if not word in set(stopwords.words('english'))]))
spam_df.head()
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\kris\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

Out[18]:

| | v1 | v2 | alpha_text | imp_text |
|---|---|---|---|---|
| **0** | ham | Go until jurong point, crazy.. Available only ... | go until jurong point crazy available only in ... | go jurong point crazy available bugis n great ... |
| **1** | ham | Ok lar... Joking wif u oni... | ok lar joking wif u oni | ok lar joking wif u oni |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | free entry in a wkly comp to win fa cup final... | free entry wkly comp win fa cup final tkts st ... |
| **3** | ham | U dun say so early hor... U c already then say... | u dun say so early hor u c already then say | u dun say early hor u c already say |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... | nah i dont think he goes to usf he lives aroun... | nah dont think goes usf lives around though |

In [19]:

```
def tokenize(data):
    generated_token = list(data.split())
    return generated_token
spam_df['token_text'] = spam_df['imp_text'].apply(lambda x: tokenize(x))
spam_df.head()
```

Out[19]:

| | v1 | v2 | alpha_text | imp_text | token_text |
|---|---|---|---|---|---|
| **0** | ham | Go until jurong point, crazy.. Available only ... | go until jurong point crazy available only in ... | go jurong point crazy available bugis n great ... | [go, jurong, point, crazy, available, bugis, n... |
| **1** | ham | Ok lar... Joking wif u oni... | ok lar joking wif u oni | ok lar joking wif u oni | [ok, lar, joking, wif, u, oni] |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | free entry in a wkly comp to win fa cup final... | free entry wkly comp win fa cup final tkts st ... | [free, entry, wkly, comp, win, fa, cup, final,... |
| **3** | ham | U dun say so early hor... U c already then say... | u dun say so early hor u c already then say | u dun say early hor u c already say | [u, dun, say, early, hor, u, c, already, say] |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... | nah i dont think he goes to usf he lives aroun... | nah dont think goes usf lives around though | [nah, dont, think, goes, usf, lives, around, t... |

In [20]:

```
nltk.download('wordnet')
nltk.download('omw-1.4')
lemmatizer = WordNetLemmatizer()
def lemmatization(list_of_words):
 lemmatized_list = [lemmatizer.lemmatize(word) for word in list_of_words]
 return lemmatized_list
spam_df['lemmatized_text'] = spam_df['token_text'].apply(lambda x: lemmatization(x))
spam_df.head()
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\kris\AppData\Roaming\nltk_data...
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\kris\AppData\Roaming\nltk_data...
```

Out[20]:

| | v1 | v2 | alpha_text | imp_text | token_text | lemmatized_text |
|---|---|---|---|---|---|---|
| **0** | ham | Go until jurong point, crazy.. Available only ... | go until jurong point crazy available only in ... | go jurong point crazy available bugis n great ... | [go, jurong, point, crazy, available, bugis, n... | [go, jurong, point, crazy, available, bugis, n... |
| **1** | ham | Ok lar... Joking wif u oni... | ok lar joking wif u oni | ok lar joking wif u oni | [ok, lar, joking, wif, u, oni] | [ok, lar, joking, wif, u, oni] |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | free entry in a wkly comp to win fa cup final... | free entry wkly comp win fa cup final tkts st ... | [free, entry, wkly, comp, win, fa, cup, final,... | [free, entry, wkly, comp, win, fa, cup, final,... |
| **3** | ham | U dun say so early hor... U c already then say... | u dun say so early hor u c already then say | u dun say early hor u c already say | [u, dun, say, early, hor, u, c, already, say] | [u, dun, say, early, hor, u, c, already, say] |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... | nah i dont think he goes to usf he lives aroun... | nah dont think goes usf lives around though | [nah, dont, think, goes, usf, lives, around, t... | [nah, dont, think, go, usf, life, around, though] |

In [21]:

```
spam_df['clean'] = spam_df['lemmatized_text'].apply(lambda x: ' '.join(x))
spam_df.head()
```

Out[21]:

| | v1 | v2 | alpha_text | imp_text | token_text | lemmatized_text | clean |
|---|---|---|---|---|---|---|---|
| **0** | ham | Go until jurong point, crazy.. Available only ... | go until jurong point crazy available only in ... | go jurong point crazy available bugis n great ... | [go, jurong, point, crazy, available, bugis, n... | [go, jurong, point, crazy, available, bugis, n... | go jurong point crazy available bugis n great ... |
| **1** | ham | Ok lar... Joking wif u oni... | ok lar joking wif u oni | ok lar joking wif u oni | [ok, lar, joking, wif, u, oni] | [ok, lar, joking, wif, u, oni] | ok lar joking wif u oni |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | free entry in a wkly comp to win fa cup final... | free entry wkly comp win fa cup final tkts st ... | [free, entry, wkly, comp, win, fa, cup, final,... | [free, entry, wkly, comp, win, fa, cup, final,... | free entry wkly comp win fa cup final tkts st ... |
| **3** | ham | U dun say so early hor... U c already then say... | u dun say so early hor u c already then say | u dun say early hor u c already say | [u, dun, say, early, hor, u, c, already, say] | [u, dun, say, early, hor, u, c, already, say] | u dun say early hor u c already say |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... | nah i dont think he goes to usf he lives aroun... | nah dont think goes usf lives around though | [nah, dont, think, goes, usf, lives, around, t... | [nah, dont, think, go, usf, life, around, though] | nah dont think go usf life around though |

In [22]:

```
df1 = spam_df.loc[spam_df['v1'] == 'spam']
df2 = spam_df.loc[spam_df['v1'] == 'ham']
spam = set()
df1['clean'].str.lower().str.split().apply(spam.update)
print("Number of unique words in spam", len(spam))
ham = set()
df2['clean'].str.lower().str.split().apply(ham.update)
print("Number of unique words in ham", len(ham))
```

```
Number of unique words in spam 2037
Number of unique words in ham 6738
```

In [23]:

```
X = spam_df['clean']
y = spam_df['v1']
le = LabelEncoder()
y = le.fit_transform(y)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.15, random_state=
42, stratify=y)
tokenizer = Tokenizer(num_words=1000)
tokenizer.fit_on_texts(X_train)
tokenized_train = tokenizer.texts_to_sequences(X_train)
X_train = tf.keras.utils.pad_sequences(tokenized_train, maxlen=100)
tokenized_test = tokenizer.texts_to_sequences(X_test)
X_test = tf.keras.utils.pad_sequences(tokenized_test, maxlen=100)
```

In [24]:

```
model = Sequential()
```

In [25]:

```
model.add(Embedding(1000, output_dim=50, input_length=100))
model.add(LSTM(units=64 , return_sequences = True, dropout = 0.2))
model.add(LSTM(units=32 , dropout = 0.1))
model.add(Dense(units = 64 , activation = 'relu'))
model.add(Dense(units = 32 , activation = 'relu'))
model.add(Dense(1, activation='sigmoid'))
```

In [26]:

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

In [27]:

```
model.fit(X_train, y_train, batch_size=128,epochs=10,validation_split=0.2,callbacks=[Ea
rlyStopping(monitor='val_loss',patience=2)])
```

```
Epoch 1/10
28/28 [==============================] - 11s 198ms/step - loss: 0.4889 - a
ccuracy: 0.8540 - val_loss: 0.3742 - val_accuracy: 0.8760
Epoch 2/10
28/28 [==============================] - 4s 153ms/step - loss: 0.3679 - ac
curacy: 0.8731 - val_loss: 0.3231 - val_accuracy: 0.8760
Epoch 3/10
28/28 [==============================] - 4s 153ms/step - loss: 0.2131 - ac
curacy: 0.9161 - val_loss: 0.1187 - val_accuracy: 0.9727
Epoch 4/10
28/28 [==============================] - 4s 153ms/step - loss: 0.0765 - ac
curacy: 0.9792 - val_loss: 0.0755 - val_accuracy: 0.9772
Epoch 5/10
28/28 [==============================] - 4s 159ms/step - loss: 0.0474 - ac
curacy: 0.9846 - val_loss: 0.0887 - val_accuracy: 0.9727
Epoch 6/10
28/28 [==============================] - 4s 155ms/step - loss: 0.0333 - ac
curacy: 0.9889 - val_loss: 0.0758 - val_accuracy: 0.9738
```

Out[27]:

```
<keras.callbacks.History at 0x17392470250>
```

In [28]:

```
model.save('spam-classifier.h5')
```

In [30]:

```
print(model.evaluate(X_test,y_test)[1]*100 , "%")
```

```
25/25 [==============================] - 1s 21ms/step - loss: 0.0653 - acc
uracy: 0.9807
98.06700944900513 %
```

In [ ]:

```

```