BUILDING A SMARTER AI-POWERED SPAM CLASSIFIER

TEAM MEMBER
311121205048-REFFY PON ESTHER. R

PHASE 5

Project: Building a Smarter AI-Powered Spam Classifier

Problem Statement:
The objective of this project is to develop an AI-powered spam filtering system that effectively identifies and filters out spam content from various communication channels, such as email, messaging apps, and online comments. The system should be capable of adapting to new spamming techniques and minimizing false positives while maximizing the identification of true spam.

Key Requirements:

1. **Multi-Channel Compatibility:** The AI-powered spam filter should be designed to work across multiple communication channels, including email, messaging apps, social media, and online comments.

2. **Adaptability:** The system should be able to adapt to new spamming techniques, patterns, and trends over time. This may involve the use of machine learning and natural language processing techniques.

3. **High Accuracy:** The spam filter should have a high accuracy rate in identifying and filtering out spam, while minimizing false positives to avoid legitimate messages being marked as spam.

4. **Scalability**: The system should be able to handle a large volume of messages and comments in real-time, and it should be scalable to accommodate growing user bases.

5. **User Customization**: Users should have the ability to customize the filter to a certain extent, allowing them to adjust the level of spam filtering to their preferences.

6. **Real-time Analysis:** The system should analyze and filter messages in real-time, providing immediate protection against spam.

7. **Robustness:** It should be robust to attempts to circumvent the filter, such as obfuscation techniques used by spammers.

8. **Data Privacy:** The system should ensure the privacy and security of user data while processing messages and comments for spam identification.

Deliverables:
The project should result in the development and deployment of an AI-powered spam filtering system that meets the above requirements. The deliverables should include:

1. A functional spam filter system.
2. Integration with various communication channels.
3. Documentation for users and administrators.
4. Regular updates and maintenance to adapt to new spamming techniques.

Success Criteria:
The success of this project will be measured based on:

1. High accuracy in spam detection and low false positive rates.
2. A decrease in the volume of spam messages and comments reaching users.
3. Positive user feedback and satisfaction with the spam filter.
4. Adaptability to new spamming techniques and trends.
5. Scalability to handle the expected user load.

Impact:
A successful AI-powered spam filtering system will enhance user experiences across various communication channels by reducing the influx of spam. It will also contribute to the overall security and integrity of online platforms and communication services by protecting users from spam-related threats.

bjective: –

Machine learning algorithms use statistical models to classify data. In the case of spam detection, a trained machine learning model must be able to determine whether the sequence of words found in an email is closer to those found in spam emails or safe ones

.

Proposed model of the system: –

As we look at spam detection systems that use Machine Learning (ML) techniques, it's vital to take a look at the history of ML in the field as well as the many methods that are now used to identify spam. Researchers have discovered that the content of spam emails, as well as their operational procedures, evolve with time. As a result, the tactics that are currently effective may become obsolete in the near future. The conceptual drift [8] is a term used to describe this occurrence. Machine Learning is an engineering approach that allows computational instruments to behave without being explicitly programmed. Because of the ML system's ability to evolve, limiting concept drift, this strategy is a significant help in detecting and combating spam.

In the next section, we'll go through a variety of machine learning techniques, approaches, and algorithms, as well as the benefits of each, using Supervised, Unsupervised, and Semi-Supervised Machine Learning algorithms Approaches.

DATASET: https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset

	A	B
2	ham	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
3	ham	Ok lar... Joking wif u oni...
4	spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 084
5	ham	U dun say so early hor... U c already then say...
6	ham	Nah I don't think he goes to usf, he lives around here though
7	spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, å£1.
8	ham	Even my brother is not like to speak with me. They treat me like aids patent.
9	ham	As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. Press *9 to cop
10	spam	WINNER!! As a valued network customer you have been selected to receivea å£900 prize reward! To claim call 09061701461. Claim code K
11	spam	Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update C
12	ham	I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.
13	spam	SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandCs apply Reply HL 4 in
14	spam	URGENT! You have won a 1 week FREE membership in our å£100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net
15	ham	I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promi
16	ham	I HAVE A DATE ON SUNDAY WITH WILL!!
17	spam	XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>> http://wap. xxxmobilemovieclub.cor
18	ham	Oh k...i'm watching here:)
19	ham	Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.
20	ham	Fine if thatåÕs the way u feel. thatåÕs the way its gota b

Email Spam Classifier

The objective is to develop a machine learning model that can categorize emails into two categories: spam and non-spam (often referred to as "ham").

This model will help us filter out unwanted and potentially harmful emails from our inbox.

We will follow standard data science procedures, including data loading, preprocessing, feature extraction, model training, evaluation, and prediction, to achieve this goal.

Let's begin building our email spam detector!

Importing Necessary Libraries

In [1]:
```
# Import Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

Load and Explore the Dataset

In [2]:
```
# Load the dataset
df = pd.read_csv("/kaggle/input/sms-spam-collection-dataset/spam.csv", encoding='ISO-8859-1')
```

In [3]:
```
# Display the first few rows of the dataset
df.head()
```

Out[3]:

	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

Data Preprocessing

In [4]:
```
# Display the column names of the DataFrame
print(df.columns)
```

```
Index(['v1', 'v2', 'Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], dtype='object')
```

In [5]:
```
# Convert 'spam' and 'ham' to binary labels
df['v1'] = df['v1'].map({'spam': 0, 'ham': 1})
```

In [6]:
```
# Split the data into features (X) and target (Y)
X = df["v2"]
Y = df["v1"]
```

In [7]:
```
# Split the data into training and test sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.35, random_state=3)
```

Feature Extraction - TF-IDF

In [8]:
```
# TF-IDF feature extraction
tfidf_vectorizer = TfidfVectorizer(min_df=1, stop_words='english', lowercase=True)
X_train_features = tfidf_vectorizer.fit_transform(X_train)
X_test_features = tfidf_vectorizer.transform(X_test)
```

Model Training (Random Forest)

In [9]:
```
# Model training
model = RandomForestClassifier(n_estimators=100, random_state=3)
model.fit(X_train_features, Y_train)
```

Out[9]:
```
RandomForestClassifier(random_state=3)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Model Evaluation (Random Forest)

In [10]:
```
prediction_on_training_data = model.predict(X_train_features)
accuracy_on_training_data = accuracy_score(Y_train, prediction_on_training_data)

prediction_on_test_data = model.predict(X_test_features)
accuracy_on_test_data = accuracy_score(Y_test, prediction_on_test_data)
```

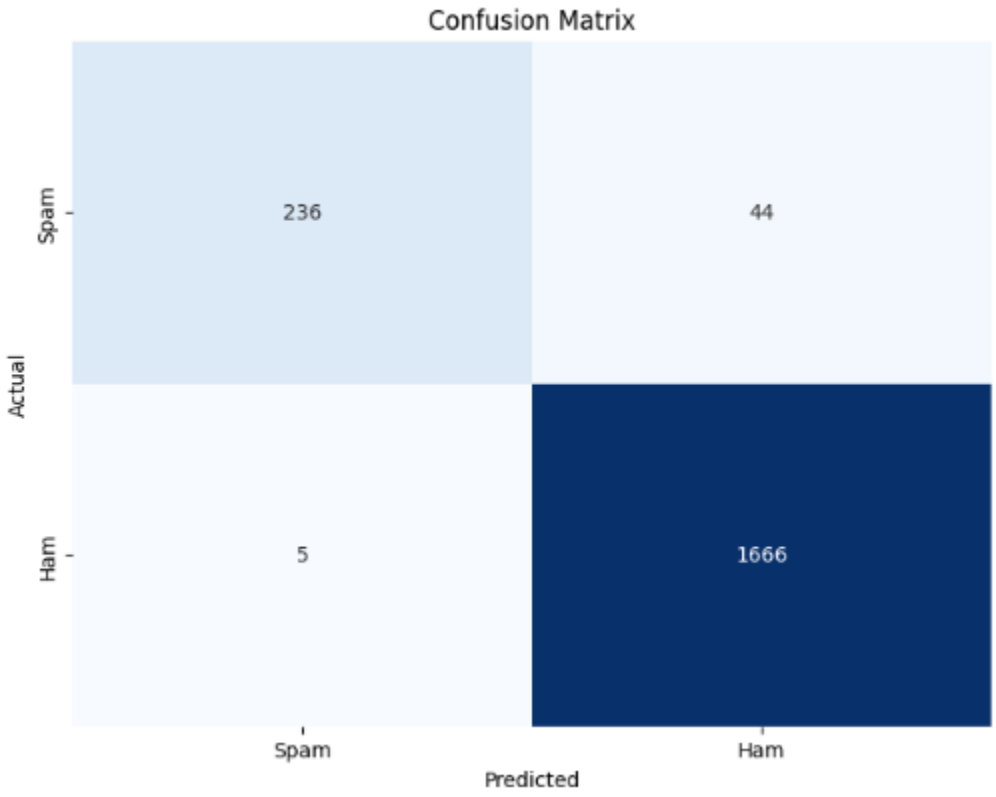
In [11]:
```
#Print accuracy

print('Accuracy on training data: {:.2f} %'.format(accuracy_on_training_data * 100))
print('Accuracy on test data: {:.2f} %'.format(accuracy_on_test_data * 100))
```

```
Accuracy on training data: 100.00 %
Accuracy on test data: 97.49 %
```

Confusion Matrix Visualization(Random Forest Classifier)

In [12]:

```
# Confusion Matrix Visualization
conf_matrix = confusion_matrix(Y_test, prediction_on_test_data)
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", cbar=False,
            xticklabels=['Spam', 'Ham'], yticklabels=['Spam', 'Ham'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```



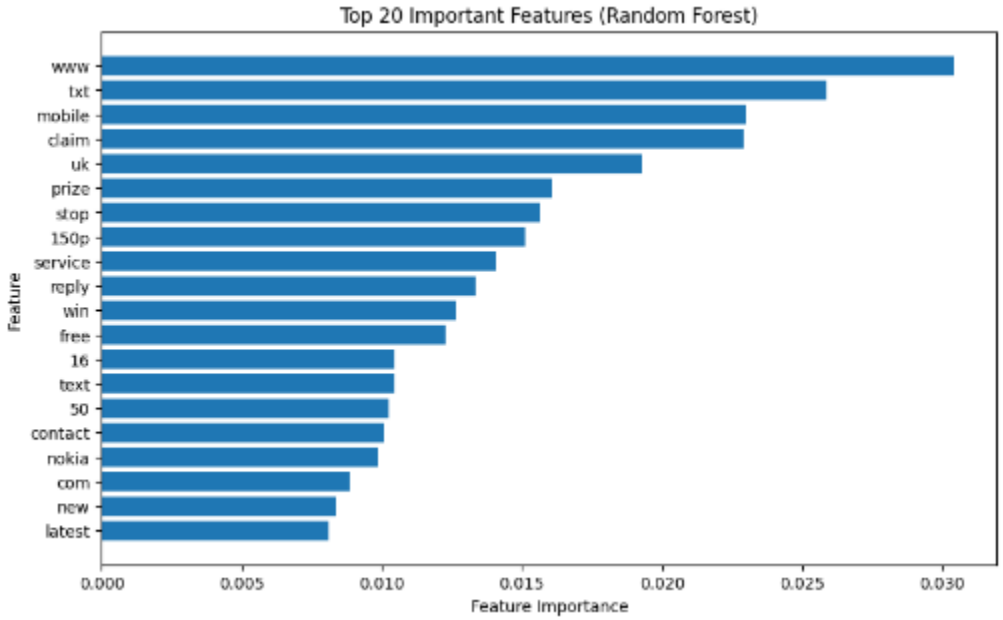
Classification Report (Random Forest Classifier)

In [13]:
```
classification_rep = classification_report(Y_test, prediction_on_test_data, target_names=['Spam', 'Ham'])
print("Classification Report:")
print(classification_rep)
```

```
Classification Report:
              precision    recall  f1-score   support

        Spam       0.98      0.84      0.91       280
         Ham       0.97      1.00      0.99      1671

    accuracy                           0.97      1951
   macro avg       0.98      0.92      0.95      1951
weighted avg       0.97      0.97      0.97      1951
```

Feature Importance Visualization (Random Forest)

In [14]:
```
feature_importance = model.feature_importances_
feature_names = tfidf_vectorizer.get_feature_names_out()
sorted_idx = np.argsort(feature_importance)[-20:]  # Top 20 important features

plt.figure(figsize=(10, 6))
plt.barh(range(len(sorted_idx)), feature_importance[sorted_idx], align="center")
plt.yticks(range(len(sorted_idx)), [feature_names[i] for i in sorted_idx])
plt.xlabel("Feature Importance")
plt.ylabel("Feature")
plt.title("Top 20 Important Features (Random Forest)")
plt.show()
```



Make Predictions on New Input (Random Forest Classifier)

In [15]:
```
input_your_mail = "Keep yourself safe for me because I need you and I miss you already and I envy everyone that see's
input_data_features = tfidf_vectorizer.transform([input_your_mail])
prediction = model.predict(input_data_features)
if prediction[0] == 1:
    print("Ham Mail")
else:
    print("Spam Mail")
```

```
Ham Mail
```

In [50]:
```
# num of words
df['num_words'] = df['text'].apply(lambda x:len(nltk.word_tokenize(x)))
```

In [51]:
```
df.head()
```

Out[51]:

	target	text	num_characters	num_words
0	0	Go until jurong point, crazy.. Available only ...	111	24
1	0	Ok lar... Joking wif u oni...	29	8
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37
3	0	U dun say so early hor... U c already then say...	49	13
4	0	Nah I don't think he goes to usf, he lives aro...	61	15

In [53]:
```
df['num_sentences'] = df['text'].apply(lambda x:len(nltk.sent_tokenize(x)))
```

In [54]:
```
df.head()
```

Out[54]:

	target	text	num_characters	num_words	num_sentences
0	0	Go until jurong point, crazy.. Available only ...	111	24	2
1	0	Ok lar... Joking wif u oni...	29	8	2
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2
3	0	U dun say so early hor... U c already then say...	49	13	1
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1

In [55]:
```
df[['num_characters','num_words','num_sentences']].describe()
```

Out[55]:

	num_characters	num_words	num_sentences
count	5169.000000	5169.000000	5169.000000
mean	78.923776	18.456375	1.962275
std	58.174846	13.323322	1.433892
min	2.000000	1.000000	1.000000
25%	36.000000	9.000000	1.000000
50%	60.000000	15.000000	1.000000
75%	117.000000	26.000000	2.000000
max	910.000000	220.000000	38.000000

In [58]:
```
# ham
df[df['target'] == 0][['num_characters','num_words','num_sentences']].describe()
```

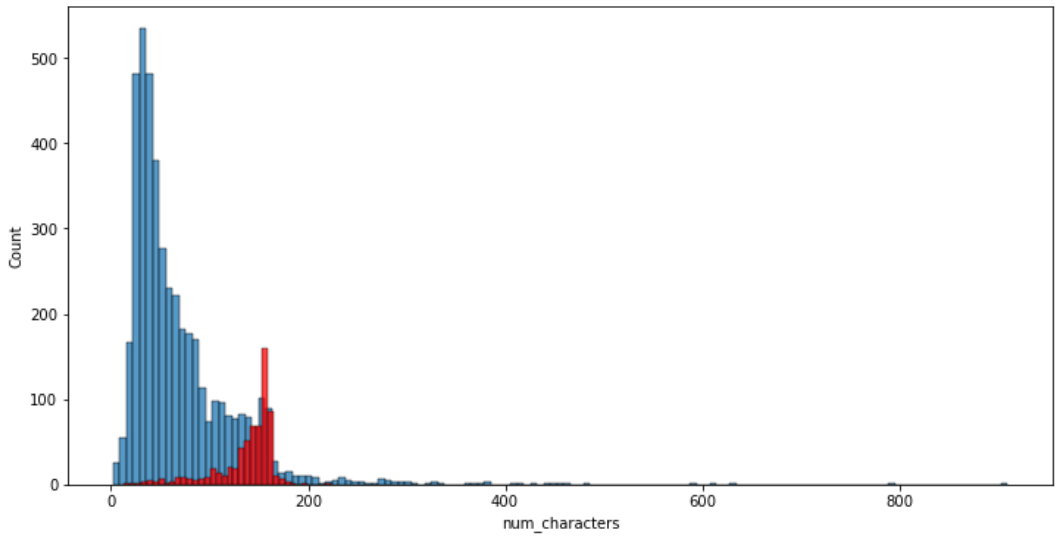
Out[58]:

	num_characters	num_words	num_sentences
count	4516.000000	4516.000000	4516.000000
mean	70.456820	17.123339	1.815545
std	56.356802	13.491315	1.364098
min	2.000000	1.000000	1.000000
25%	34.000000	8.000000	1.000000
50%	52.000000	13.000000	1.000000
75%	90.000000	22.000000	2.000000
max	910.000000	220.000000	38.000000

In [78]:
```
import seaborn as sns
```

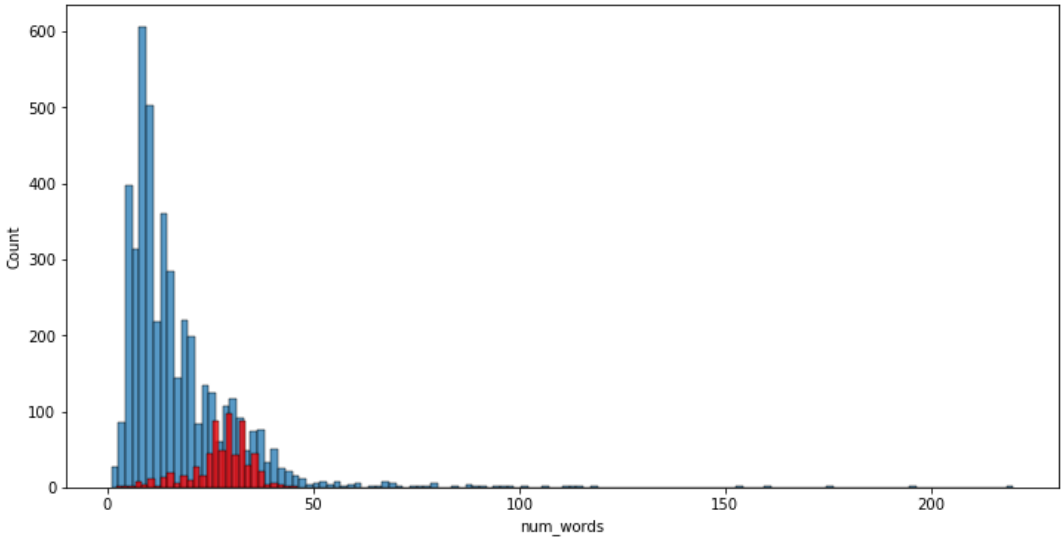
In [84]:
```
plt.figure(figsize=(12,6))
sns.histplot(df[df['target'] == 0]['num_characters'])
sns.histplot(df[df['target'] == 1]['num_characters'],color='red')
```

Out[84]: <AxesSubplot:xlabel='num_characters', ylabel='Count'>



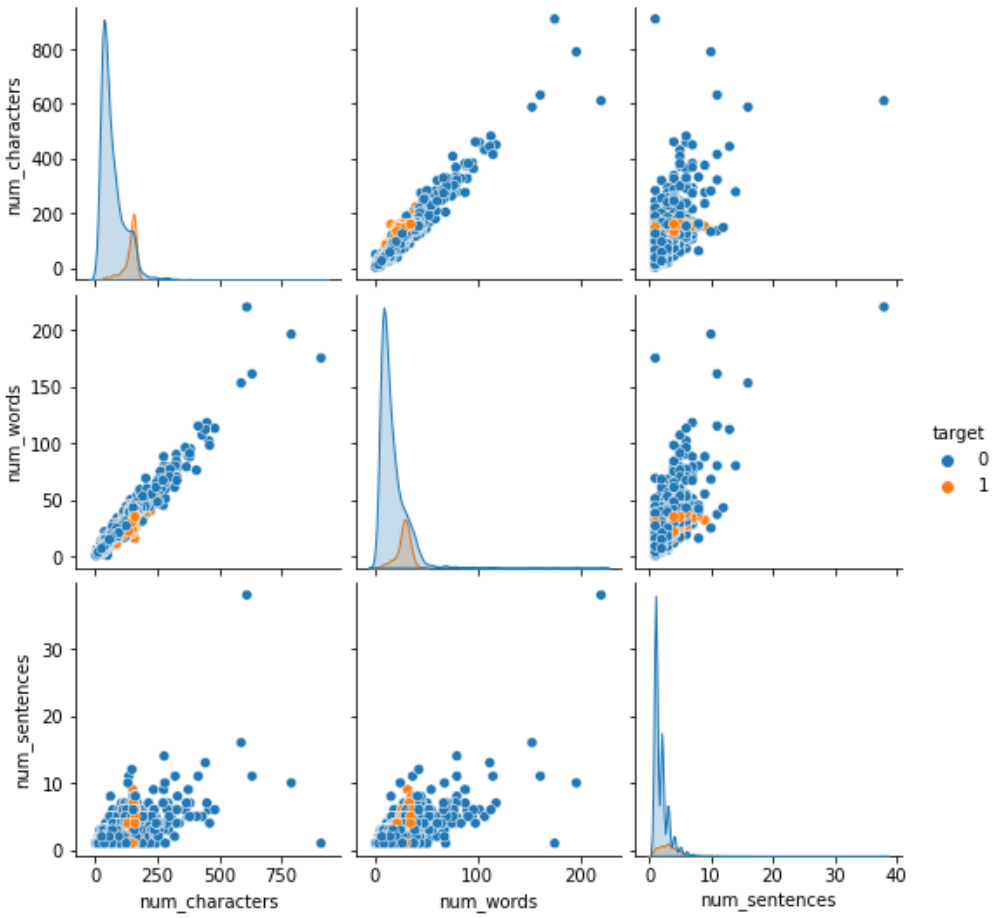
In [85]:
```
plt.figure(figsize=(12,6))
sns.histplot(df[df['target'] == 0]['num_words'])
sns.histplot(df[df['target'] == 1]['num_words'],color='red')
```

Out[85]: <AxesSubplot:xlabel='num_words', ylabel='Count'>



In [86]:
```
sns.pairplot(df,hue='target')
```

Out[86]: <seaborn.axisgrid.PairGrid at 0x16f88c4a4f0>



In [89]:
```
sns.heatmap(df.corr(),annot=True)
```

Out[89]: <AxesSubplot:>