In [1]:

```python
import pandas as pd
df = pd.read_csv('train.csv')
df.head()
```

	id	title	author	
0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucus	House Dem Aide: We Didn't Even See Comey's
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29,
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced t

In [2]:

```python
## getting features.
X = df.drop('label', axis = 1)
X.head()
```

	id	title	author	
0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucus	House Dem Aide: We Didn't Even See Comey's
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29,
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced t

In [3]:

```python
## getting dependent feature such as Label
y = df['label']
y.head()
```

```
0    1
1    0
2    1
3    1
4    1
Name: label, dtype: int64
```

In [6]:

```python
from sklearn.feature_extraction.text import CountVectorizer
```

In [7]:

```python
## removing the null values from the data frame.
df = df.dropna()
df.head(10)
```

	id	title	author	
0	0	House Dem Aide: We Didn’t Even See Comey’s Let...	Darrell Lucus	House Dem Aide: We Didn’t Even See Let...
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired Oc
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been se
5	5	Jackie Mason: Hollywood Would Love Trump if He...	Daniel Nussbaum	In these trying times, Jackie Mason is
7	7	Benoît Hamon Wins French Socialist Party’s Pre...	Alissa J. Rubin	PARIS — France chose an idealistic, t
9	9	A Back-Channel Plan for Ukraine and Russia, Co...	Megan Twohey and Scott Shane	A week before Michael T. Flynn resign
10	10	Obama’s Organizing for Action Partners with So...	Aaron Klein	Organizing for Action, the activist grou
11	11	BBC Comedy Sketch "Real Housewives of ISIS" Ca...	Chris Tomlinson	The BBC produced spoof on the “Real Housewives...

In [9]:

```python
messages = df.copy()
messages.reset_index(inplace = True)
messages.head()
```

	index	id	title	author	
0	0	0	House Dem Aide: We Didn’t Even See Comey’s Let...	Darrell Lucus	House Dem Aide: We Didn’t Even See (Let...
1	1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the
2	2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired Octo
3	3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US /
4	4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sen

In [11]:

```python
messages['title'][6]
```

```
'Benoît Hamon Wins French Socialist Party’s Presidential Nomination - The New York Times'
```

In [13]:

```
## preprocessing and stemming the news.
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
corpus = []

for i in range(len(messages)):
    review = re.sub('[^a-zA-Z]', ' ', messages['title'][i])
    review = review.lower()
    review = review.split()

    review = [ps.stem(word) for word in review if not word in stopwords.words('english')]
    review = " ".join(review)
    corpus.append(review)
```

In [14]:

```
corpus[6]
```

```
'beno hamon win french socialist parti presidenti nomin new york time'
```

In [15]:

```
## apply countvectorizer
## creating bag of words
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features = 5000, ngram_range = (1,3))
X = cv.fit_transform(corpus).toarray()
```

In [18]:

```
X.shape
```

```
(18285, 5000)
```

In [19]:

```
y = messages['label']
```

In [20]:

```
## dividing the data set into testing and training dataset.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y ,test_size = 0.2, random_state = 0)
```

In [23]:

```
cv.get_feature_names()[:20]
```

```
['abandon',
 'abc',
 'abc news',
 'abduct',
 'abe',
 'abedin',
 'abl',
 'abort',
 'abroad',
 'absolut',
 'abstain',
 'absurd',
 'abus',
 'abus new',
 'abus new york',
 'academi',
 'accept',
 'access',
 'access pipelin',
 'access pipelin protest']
```

In [24]:

```
cv.get_params()
```

```
{'analyzer': 'word',
 'binary': False,
 'decode_error': 'strict',
 'dtype': numpy.int64,
 'encoding': 'utf-8',
 'input': 'content',
 'lowercase': True,
 'max_df': 1.0,
 'max_features': 5000,
 'min_df': 1,
 'ngram_range': (1, 3),
 'preprocessor': None,
 'stop_words': None,
 'strip_accents': None,
 'token_pattern': '(?u)\\b\\w\\w+\\b',
 'tokenizer': None,
 'vocabulary': None}
```

In [25]:

```
count_df = pd.DataFrame(X_train, columns = cv.get_feature_names())
```

In [29]:

```
count_df.head()
```

	abandon	abc	abc news	abduct	abe	abedin	abl	abort	abroad	absolut	...	zero	zika	zika viru	zionist	zo
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0

```
5 rows × 5000 columns
```

In [30]:

```
import matplotlib.pyplot as plt
```

In [31]:

```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    See full source and example:
    http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html

    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
```

MultinomialNB Algorithm

In [33]:

```
from sklearn.naive_bayes import MultinomialNB
classifier = MultinomialNB()
```

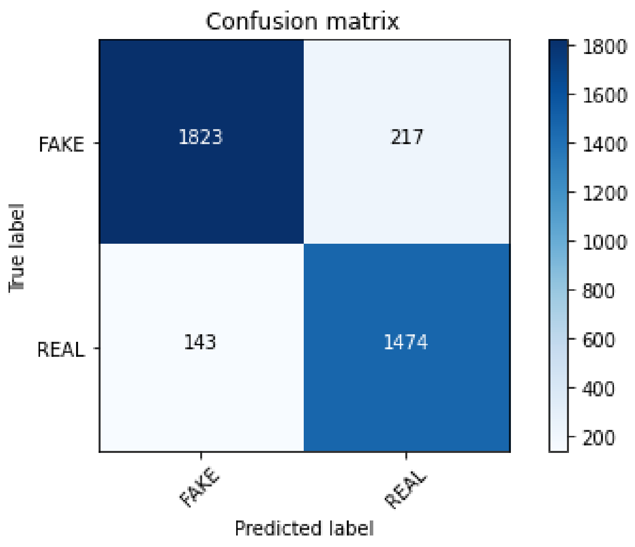
In [34]:

```
from sklearn import metrics
import numpy as np
import itertools
```

In [37]:

```
classifier.fit(X_train, y_train)
pred = classifier.predict(X_test)
score = metrics.accuracy_score(y_test, pred)
print('Accuracy : %0.3f'% score)
cm = metrics.confusion_matrix(y_test, pred)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
```

```
Accuracy : 0.902
Confusion matrix, without normalization
```



Passive Aggresive Classifier Algorithm

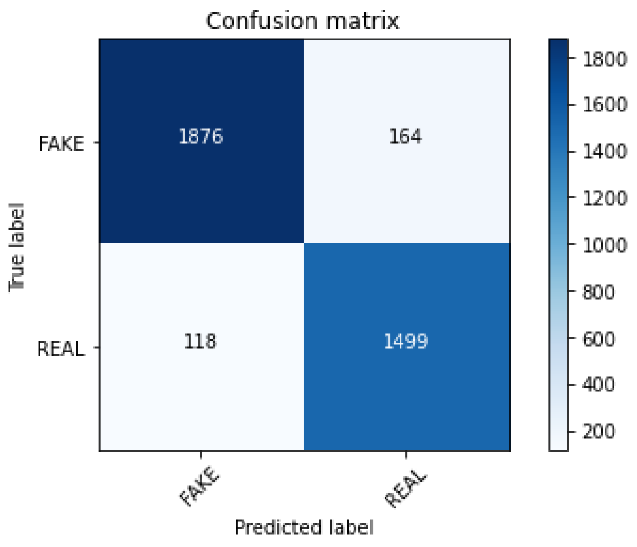
In [40]:

```
from sklearn.linear_model import PassiveAggressiveClassifier
linear_clf = PassiveAggressiveClassifier(n_iter_no_change = 50)
```

In [42]:

```
linear_clf.fit(X_train, y_train)
pred = linear_clf.predict(X_test)
score = metrics.accuracy_score(y_test, pred)
print('Accuracy : %0.3f' %score)
cm = metrics.confusion_matrix(y_test, pred)
plot_confusion_matrix(cm, classes = ['FAKE', 'REAL'])
```

```
Accuracy : 0.923
Confusion matrix, without normalization
```



Multinomial Classifier with Hyperparameter

In [43]:

```
classifier = MultinomialNB( alpha = 0.1)
```

In [47]:

```python
previous_score = 0
for alpha in np.arange(0,1,0.1):
    sub_classifier = MultinomialNB(alpha = alpha)
    sub_classifier.fit(X_train, y_train)
    y_pred = sub_classifier.predict(X_test)
    score = metrics.accuracy_score(y_test, y_pred)
    if score > previous_score:
        classifier = sub_classifier
    print("Alpha : {}, Score : {}".format(alpha, score))
```

```
c:\my softwares\lib\site-packages\sklearn\naive_bayes.py:508: UserWarning: alpha too small will result in numeric erro
1.0e-10
  warnings.warn('alpha too small will result in numeric errors, '


Alpha : 0.0, Score : 0.8955427946404156
Alpha : 0.1, Score : 0.9051134809953514
Alpha : 0.2, Score : 0.9051134809953514
Alpha : 0.30000000000000004, Score : 0.9059338255400602
Alpha : 0.4, Score : 0.9051134809953514
Alpha : 0.5, Score : 0.9042931364506426
Alpha : 0.6000000000000001, Score : 0.9037462400875034
Alpha : 0.7000000000000001, Score : 0.9026524473612251
Alpha : 0.8, Score : 0.9021055509980859
Alpha : 0.9, Score : 0.9015586546349467
```

In [48]:

```python
## get features names
feature_names = cv.get_feature_names()
```

In [49]:

```python
classifier.coef_[0]
```

```
c:\my softwares\lib\site-packages\sklearn\utils\deprecation.py:101: FutureWarning: Attribute coef_ was deprecated in v
be removed in 1.1 (renaming of 0.26).
  warnings.warn(msg, category=FutureWarning)



array([ -9.25630829,  -8.65949222,  -9.25630829, ..., -10.95090401,
        -8.77868073,  -9.48456694])
```

In [50]:

```
### most real values
sorted(zip(classifier.coef_[0], feature_names), reverse = True)[0:20]
```

```
[(-3.959114000028925, 'trump'),
 (-4.270607131437483, 'hillari'),
 (-4.354971714376536, 'clinton'),
 (-4.882221251134608, 'elect'),
 (-5.1420944065413465, 'new'),
 (-5.258669435885832, 'video'),
 (-5.262423194047336, 'comment'),
 (-5.357019074680328, 'us'),
 (-5.373693074987398, 'war'),
 (-5.3821355058826805, 'hillari clinton'),
 (-5.412258265337789, 'fbi'),
 (-5.461507250345735, 'vote'),
 (-5.475370688647845, 'email'),
 (-5.552741306436383, 'world'),
 (-5.5833715723846264, 'obama'),
 (-5.687063070936072, 'donald'),
 (-5.722174928212814, 'donald trump'),
 (-5.740204262730757, 'russia'),
 (-5.822321082694582, 'america'),
 (-5.842268552606346, 'presid')]
```

In [53]:

```
### most false values
sorted(zip(classifier.coef_[0], feature_names))[:5000]
```

```
[(-10.950904007954136, 'abroad'),
 (-10.950904007954136, 'abus new'),
 (-10.950904007954136, 'abus new york'),
 (-10.950904007954136, 'act new'),
 (-10.950904007954136, 'act new york'),
 (-10.950904007954136, 'advic'),
 (-10.950904007954136, 'advis new'),
 (-10.950904007954136, 'advis new york'),
 (-10.950904007954136, 'age new'),
 (-10.950904007954136, 'age new york'),
 (-10.950904007954136, 'agenda breitbart'),
 (-10.950904007954136, 'aleppo new'),
 (-10.950904007954136, 'aleppo new york'),
 (-10.950904007954136, 'ali'),
 (-10.950904007954136, 'america breitbart'),
 (-10.950904007954136, 'america new york'),
 (-10.950904007954136, 'american breitbart'),
 (-10.950904007954136, 'american new'),
 (-10.950904007954136, 'american new york'),
 (-10.950904007954136, 'ami'),
 (-10.950904007954136, 'ami schumer'),
 (-10.950904007954136, 'amp'),
```

In [ ]: