DETECTION OF FAKE NEWS USING MACHINE LEARNING

This is an analysis of various news segments and based on this analysis a particular news is classified as real or fake Link for the dataset https://drive.google.com/drive/folders/1dYsmtW3ZQTKjAmu30uZQs-QLJ6I5OOBZ?usp=sharing (https://drive.google.com/drive/folders/1dYsmtW3ZQTKjAmu30uZQs-QLJ6I5OOBZ?usp=sharing)

Importing the libraries

In [1]:

```
import pandas as pd
import numpy as np
import re
import string
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import VotingClassifier
from sklearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
import matplotlib.pyplot as plt
import itertools
from sklearn.metrics import classification_report
```

In [2]:

```
df_fake=pd.read_csv("Fake.csv")
df_true=pd.read_csv("True.csv")
```

In [3]:

```
df_fake.tail(10)
```

Out[3]:

	title	text	subject	date
23471	Seven Iranians freed in the prisoner swap have...	21st Century Wire says This week, the historic...	Middle-east	January 20, 2016
23472	#Hashtag Hell & The Fake Left	By Dady Chery and Gilbert MercierAll writers ...	Middle-east	January 19, 2016
23473	Astroturfing: Journalist Reveals Brainwashing ...	Vic Bishop Waking TimesOur reality is carefull...	Middle-east	January 19, 2016
23474	The New American Century: An Era of Fraud	Paul Craig RobertsIn the last years of the 20t...	Middle-east	January 19, 2016
23475	Hillary Clinton: ‘Israel First’ (and no peace ...	Robert Fantina CounterpunchAlthough the United...	Middle-east	January 18, 2016
23476	McPain: John McCain Furious That Iran Treated ...	21st Century Wire says As 21WIRE reported earl...	Middle-east	January 16, 2016
23477	JUSTICE? Yahoo Settles E-mail Privacy Class-ac...	21st Century Wire says It s a familiar theme. ...	Middle-east	January 16, 2016
23478	Sunnistan: US and Allied ‘Safe Zone’ Plan to T...	Patrick Henningsen 21st Century WireRemember ...	Middle-east	January 15, 2016
23479	How to Blow $700 Million: Al Jazeera America F...	21st Century Wire says Al Jazeera America will...	Middle-east	January 14, 2016
23480	10 U.S. Navy Sailors Held by Iranian Military ...	21st Century Wire says As 21WIRE predicted in ...	Middle-east	January 12, 2016

In [4]:

```
df_fake.shape
```

Out[4]:

```
(23481, 4)
```

In [5]:

```
df_true.tail(10)
```

Out[5]:

	title	text	subject	date
21407	Mata Pires, owner of embattled Brazil builder ...	SAO PAULO (Reuters) - Cesar Mata Pires, the ow...	worldnews	August 22, 2017
21408	U.S., North Korea clash at U.N. forum over nuc...	GENEVA (Reuters) - North Korea and the United ...	worldnews	August 22, 2017
21409	U.S., North Korea clash at U.N. arms forum on ...	GENEVA (Reuters) - North Korea and the United ...	worldnews	August 22, 2017
21410	Headless torso could belong to submarine journ...	COPENHAGEN (Reuters) - Danish police said on T...	worldnews	August 22, 2017
21411	North Korea shipments to Syria chemical arms a...	UNITED NATIONS (Reuters) - Two North Korean sh...	worldnews	August 21, 2017
21412	'Fully committed' NATO backs new U.S. approach...	BRUSSELS (Reuters) - NATO allies on Tuesday we...	worldnews	August 22, 2017
21413	LexisNexis withdrew two products from Chinese ...	LONDON (Reuters) - LexisNexis, a provider of l...	worldnews	August 22, 2017
21414	Minsk cultural hub becomes haven from authorities	MINSK (Reuters) - In the shadow of disused Sov...	worldnews	August 22, 2017
21415	Vatican upbeat on possibility of Pope Francis ...	MOSCOW (Reuters) - Vatican Secretary of State ...	worldnews	August 22, 2017
21416	Indonesia to buy $1.14 billion worth of Russia...	JAKARTA (Reuters) - Indonesia will buy 11 Sukh...	worldnews	August 22, 2017

In [6]:

```
df_true.shape
```

Out[6]:

```
(21417, 4)
```

Creating label 0 for fake news and 1 for real news

In [7]:

```
df_fake["class"]=0
df_true["class"]=1
```

Creating dataset for manual testing

In [8]:

```
df_fake_manual_testing=df_fake.tail(10)
df_fake.drop([23470,23480],axis=0,inplace=True)
```

In [9]:

```
df_true_manual_testing=df_true.tail(10)
df_true.drop([21406,21416],axis=0,inplace=True)
```

In [10]:

```
df_manual_testing=pd.concat([df_fake_manual_testing,df_true_manual_testing],axis=0)
df_manual_testing.to_csv("manual_testing.csv")
```

Creating merged dataset for fake and real news

In [11]:

```
df_merge=pd.concat([df_fake,df_true],axis=0)
df_merge.tail(10)
```

Out[11]:

	title	text	subject	date	class
21405	Trump talks tough on Pakistan's 'terrorist' ha...	ISLAMABAD (Reuters) - Outlining a new strategy...	worldnews	August 22, 2017	1
21407	Mata Pires, owner of embattled Brazil builder ...	SAO PAULO (Reuters) - Cesar Mata Pires, the ow...	worldnews	August 22, 2017	1
21408	U.S., North Korea clash at U.N. forum over nuc...	GENEVA (Reuters) - North Korea and the United ...	worldnews	August 22, 2017	1
21409	U.S., North Korea clash at U.N. arms forum on ...	GENEVA (Reuters) - North Korea and the United ...	worldnews	August 22, 2017	1
21410	Headless torso could belong to submarine journ...	COPENHAGEN (Reuters) - Danish police said on T...	worldnews	August 22, 2017	1
21411	North Korea shipments to Syria chemical arms a...	UNITED NATIONS (Reuters) - Two North Korean sh...	worldnews	August 21, 2017	1
21412	'Fully committed' NATO backs new U.S. approach...	BRUSSELS (Reuters) - NATO allies on Tuesday we...	worldnews	August 22, 2017	1
21413	LexisNexis withdrew two products from Chinese ...	LONDON (Reuters) - LexisNexis, a provider of l...	worldnews	August 22, 2017	1
21414	Minsk cultural hub becomes haven from authorities	MINSK (Reuters) - In the shadow of disused Sov...	worldnews	August 22, 2017	1
21415	Vatican upbeat on possibility of Pope Francis ...	MOSCOW (Reuters) - Vatican Secretary of State ...	worldnews	August 22, 2017	1

In [12]:

```
df=df_merge.drop(["subject","date"],axis=1)
```

In [13]:

```
df = df.sample(frac = 1)
```

In [14]:

```
df.head()
```

Out[14]:

	title	text	class
13834	Nigerian army repels Boko Haram attack on town...	MAIDUGURI, Nigeria (Reuters) - Nigeria s milit...	1
14608	Zimbabwe's Mugabe, coup chief meet with smiles...	HARARE (Reuters) - A smiling President Robert ...	1
14929	WSJ REPORTER RIPS INTO DEM CANDIDATES For Thei...	THE WSJ S MARY KISSEL NAILS IT ON THE DEM DEBA...	0
14023	Backlash among German MPs against parliamentar...	BERLIN (Reuters) - German lawmakers have prote...	1
10484	‘SEEMS LIKE A THREAT’: ABC’s Raddatz Tries to ...	ABC political hack Martha Raddatz tried to bai...	0

Detecting null values

In [15]:

```
df.isnull().sum()
```

Out[15]:

```
title    0
text     0
class    0
dtype: int64
```

Function to convert the text in lowercase, remove the extra space, special chr., ulr and links.

In [16]:

```
def conversion(title):
    title = title.lower()
    title = re.sub('\[.*?\]', '', title)
    title = re.sub("\\W"," ",title)
    title = re.sub('https?://\S+|www\.\S+', '', title)
    title = re.sub('<.*?>+', '', title)
    title = re.sub('[%s]' % re.escape(string.punctuation), '', title)
    title = re.sub('\n', '', title)
    title = re.sub('\w*\d\w*', '', title)
    return title
```

In [17]:

```
df["title"] = df["title"].apply(conversion)
```

Splitting data into training and testing dataset

In [18]:

```
x = df.iloc[0:5000,0]
y = df.iloc[0:5000,-1]
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
```

Converting text to vector

In [19]:

```
vectorization = TfidfVectorizer()
xv_train = vectorization.fit_transform(x_train)
xv_test = vectorization.transform(x_test)
```

In [20]:

```
xv = vectorization.fit_transform(x)
```

Code to plot confusion matrix

In [21]:

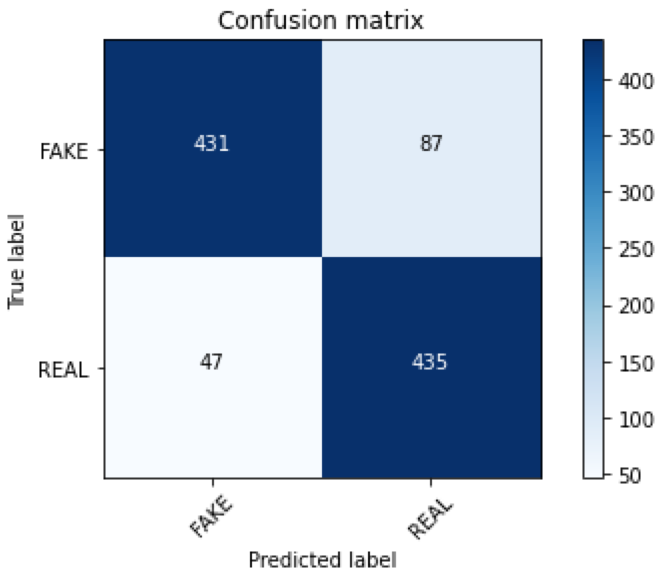
```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    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')
```

K NEAREST NEIGHBORS

In [22]:

```
knn=KNeighborsClassifier(n_neighbors=3)
knn.fit(xv_train, y_train)
pred_train = knn.predict(xv_train)
pred_test = knn.predict(xv_test)
print("Bias is : ",1-accuracy_score(pred_train,y_train))
print("Variance is: ",1-accuracy_score(pred_test,y_test))
print("Accuracy is: ",accuracy_score(pred_test,y_test))
print("Cross Validation result is: ",cross_val_score(knn, xv, y, cv=10, scoring ='accuracy'
cm=confusion_matrix(y_test,pred_test)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(classification_report(y_test, pred_test))
```

```
Bias is :  0.06799999999999995
Variance is:  0.134
Accuracy is:  0.866
Cross Validation result is:  0.8535999999999999
Confusion matrix, without normalization
              precision    recall  f1-score   support

           0       0.90      0.83      0.87       518
           1       0.83      0.90      0.87       482

    accuracy                           0.87      1000
   macro avg       0.87      0.87      0.87      1000
weighted avg       0.87      0.87      0.87      1000
```

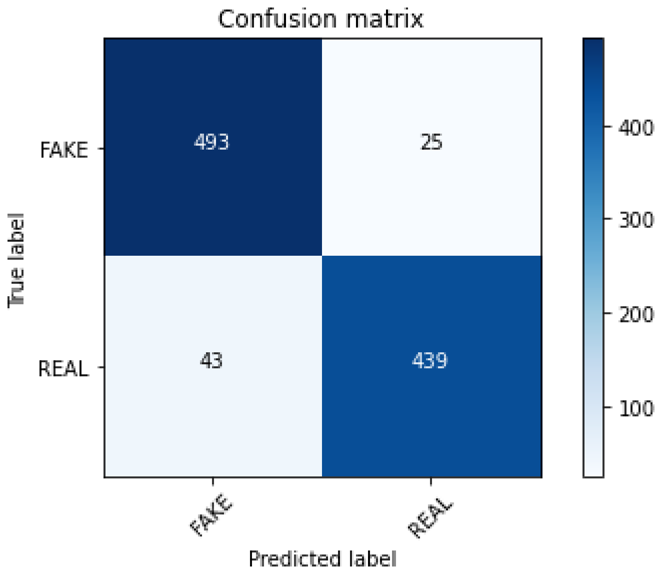


LOGISTIC REGRESSION

In [23]:

```
LR = LogisticRegression()
LR.fit(xv_train,y_train)
pred_train = LR.predict(xv_train)
pred_test = LR.predict(xv_test)
print("Bias is : ",1-accuracy_score(pred_train,y_train))
print("Variance is: ",1-accuracy_score(pred_test,y_test))
print("Accuracy is: ",accuracy_score(pred_test,y_test))
print("Cross Validation result is: ",cross_val_score(LR, xv, y, cv=10, scoring ='accuracy')
cm=confusion_matrix(y_test,pred_test)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(classification_report(y_test, pred_test))
```

```
Bias is :  0.028249999999999997
Variance is:  0.06799999999999995
Accuracy is:  0.932
Cross Validation result is:  0.925
Confusion matrix, without normalization
              precision    recall  f1-score   support

           0       0.92      0.95      0.94       518
           1       0.95      0.91      0.93       482

    accuracy                           0.93      1000
   macro avg       0.93      0.93      0.93      1000
weighted avg       0.93      0.93      0.93      1000
```

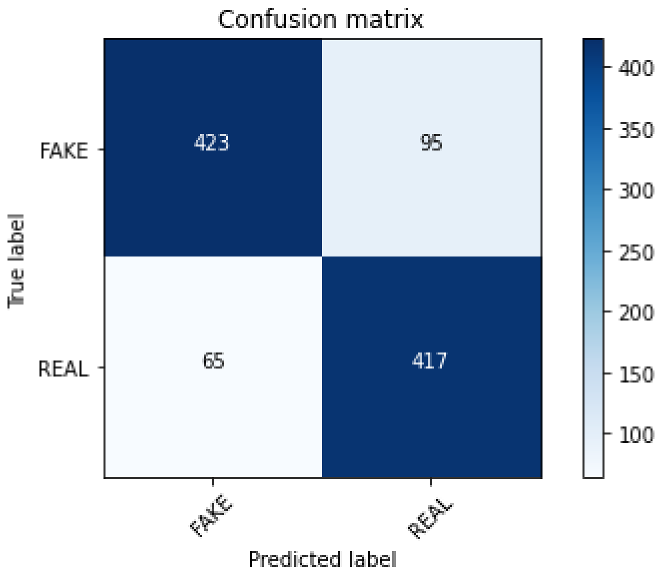


DECISION TREE

In [24]:

```
DT = DecisionTreeClassifier()
DT.fit(xv_train, y_train)
pred_train = DT.predict(xv_train)
pred_test = DT.predict(xv_test)
print("Bias is : ",1-accuracy_score(pred_train,y_train))
print("Variance is: ",1-accuracy_score(pred_test,y_test))
print("Accuracy is: ",accuracy_score(pred_test,y_test))
print("Cross Validation result is: ",cross_val_score(DT, xv, y, cv=10, scoring ='accuracy')
cm=confusion_matrix(y_test,pred_test)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(classification_report(y_test, pred_test))
```

```
Bias is :  0.0
Variance is:  0.16000000000000003
Accuracy is:  0.84
Cross Validation result is:  0.8554
Confusion matrix, without normalization
              precision    recall  f1-score   support

           0       0.87      0.82      0.84       518
           1       0.81      0.87      0.84       482

    accuracy                           0.84      1000
   macro avg       0.84      0.84      0.84      1000
weighted avg       0.84      0.84      0.84      1000
```

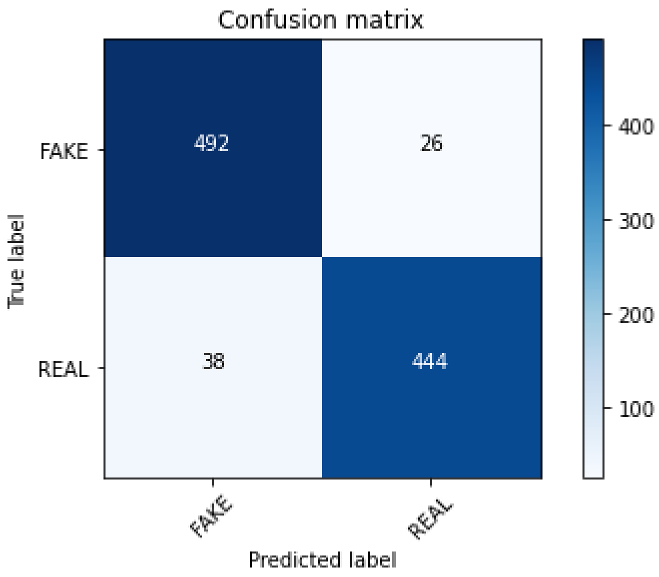


SUPPORT VECTOR CLASSIFIER

In [25]:

```
svc = SVC()
svc.fit(xv_train, y_train)
pred_train = svc.predict(xv_train)
pred_test = svc.predict(xv_test)
print("Bias is : ",1-accuracy_score(pred_train,y_train))
print("Variance is: ",1-accuracy_score(pred_test,y_test))
print("Accuracy is: ",accuracy_score(pred_test,y_test))
print("Cross Validation result is: ",cross_val_score(svc, xv, y, cv=10, scoring ='accuracy'
cm=confusion_matrix(y_test,pred_test)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(classification_report(y_test, pred_test))
```

```
Bias is :  0.0014999999999999458
Variance is:  0.06399999999999995
Accuracy is:  0.936
Cross Validation result is:  0.9296000000000001
Confusion matrix, without normalization
              precision    recall  f1-score   support

           0       0.93      0.95      0.94       518
           1       0.94      0.92      0.93       482

    accuracy                           0.94      1000
   macro avg       0.94      0.94      0.94      1000
weighted avg       0.94      0.94      0.94      1000
```



ENSEMBLING

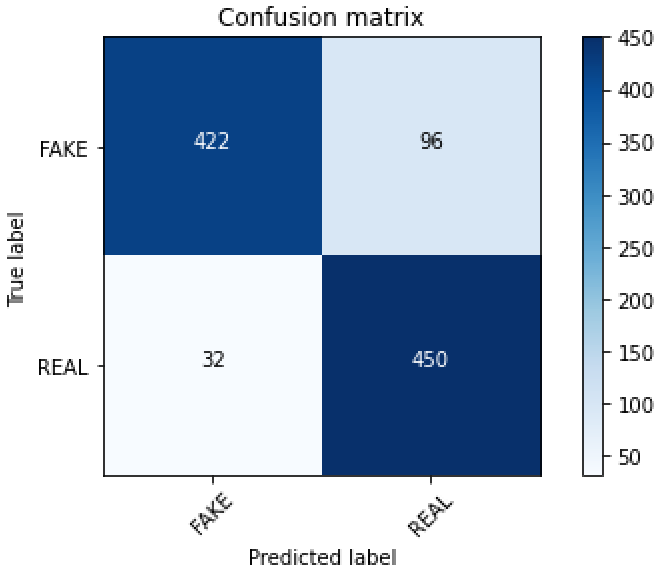
(1) IN BUILT ENSEMBLING

(a) GRADIENT BOOSTING CLASSIFIER (BOOSTING)

```
In [26]:
```

```
GBC = GradientBoostingClassifier(random_state=0)
GBC.fit(xv_train, y_train)
pred_train = GBC.predict(xv_train)
pred_test = GBC.predict(xv_test)
print("Bias is : ",1-accuracy_score(pred_train,y_train))
print("Variance is: ",1-accuracy_score(pred_test,y_test))
print("Accuracy is: ",accuracy_score(pred_test,y_test))
print("Cross Validation result is: ",cross_val_score(GBC, xv, y, cv=10, scoring ='accuracy'
cm=confusion_matrix(y_test,pred_test)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(classification_report(y_test, pred_test))
```

```
Bias is :  0.08574999999999999
Variance is:  0.128
Accuracy is:  0.872
Cross Validation result is:  0.8865999999999999
Confusion matrix, without normalization
              precision    recall  f1-score   support

           0       0.93      0.81      0.87       518
           1       0.82      0.93      0.88       482

    accuracy                           0.87      1000
   macro avg       0.88      0.87      0.87      1000
weighted avg       0.88      0.87      0.87      1000
```

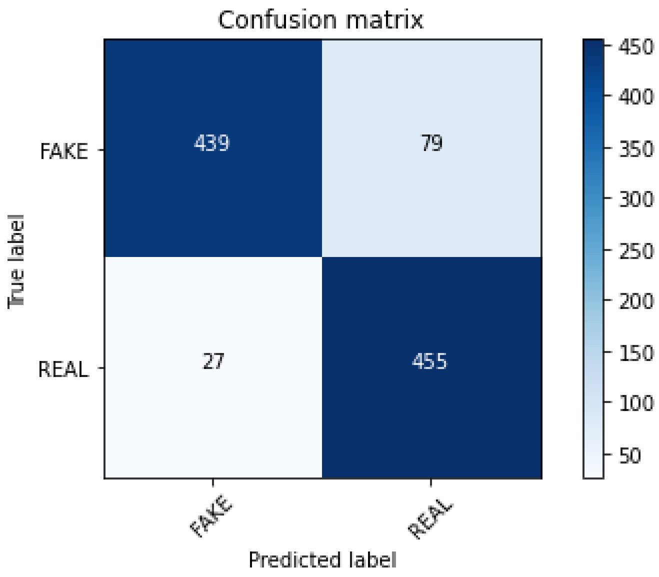


(b) RANDOM FOREST CLASSIFIER (BAGGING)

In [27]:

```
RFC = RandomForestClassifier(random_state=0)
RFC.fit(xv_train, y_train)
pred_train = RFC.predict(xv_train)
pred_test = RFC.predict(xv_test)
print("Bias is : ",1-accuracy_score(pred_train,y_train))
print("Variance is: ",1-accuracy_score(pred_test,y_test))
print("Accuracy is: ",accuracy_score(pred_test,y_test))
print("Cross Validation result is: ",cross_val_score(RFC, xv, y, cv=10, scoring ='accuracy'
cm=confusion_matrix(y_test,pred_test)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(classification_report(y_test, pred_test))
```

```
Bias is :  0.0
Variance is:  0.10599999999999998
Accuracy is:  0.894
Cross Validation result is:  0.9242000000000001
Confusion matrix, without normalization
              precision    recall  f1-score   support

           0       0.94      0.85      0.89       518
           1       0.85      0.94      0.90       482

    accuracy                           0.89      1000
   macro avg       0.90      0.90      0.89      1000
weighted avg       0.90      0.89      0.89      1000
```



2. CUSTOM ENSEMBLING

(a) PIPELINE METHOD

In [28]:

```
knn=KNeighborsClassifier(n_neighbors=3)
knn.fit(xv_train, y_train)
```

Out[28]:

```
▾       KNeighborsClassifier
KNeighborsClassifier(n_neighbors=3)
```

In [29]:

```
LR = LogisticRegression()
LR.fit(xv_train,y_train)
```

Out[29]:

```
▾ LogisticRegression
LogisticRegression()
```

In [30]:

```
svc = SVC()
svc.fit(xv_train, y_train)
```

Out[30]:

```
▾ SVC
SVC()
```

In [31]:

```
models = list()
```

In [32]:

```
logistic_regression = Pipeline([('m', LogisticRegression())])
models.append(('logistic', logistic_regression))
```

In [33]:

```
svc = Pipeline([('m', SVC())])
models.append(('svc', svc))
```

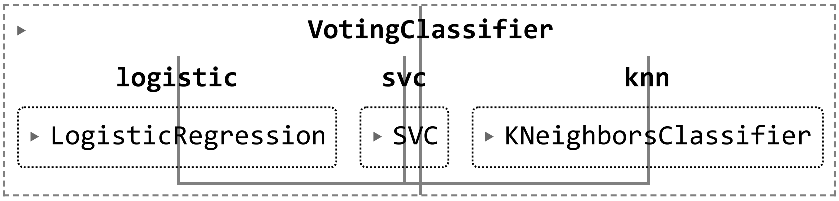
In [34]:

```
k_n_n = Pipeline([('m', KNeighborsClassifier(n_neighbors=3))])
models.append(('knn', k_n_n))
```

In [35]:

```
ensemble = VotingClassifier(estimators=models, voting='hard')
ensemble.fit(xv_train,y_train)
```

Out[35]:



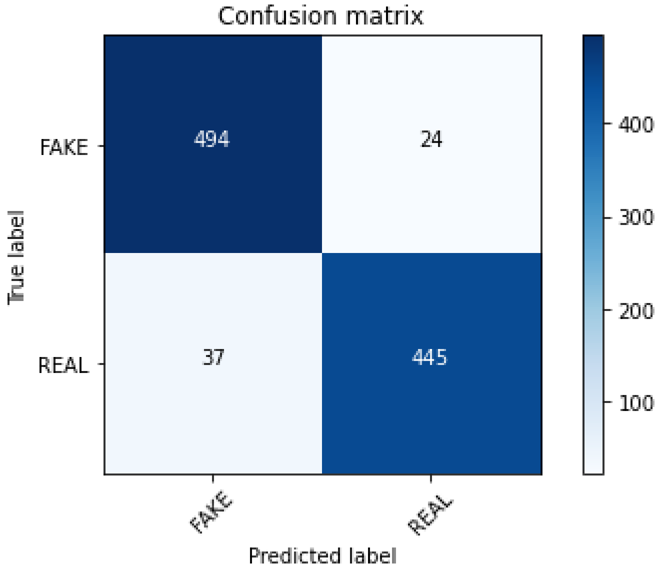
In [36]:

```
pred_train = ensemble.predict(xv_train)
pred_test = ensemble.predict(xv_test)
```

In [37]:

```
print("Bias is : ",1-accuracy_score(pred_train,y_train))
print("Variance is: ",1-accuracy_score(pred_test,y_test))
print("Accuracy is: ",accuracy_score(pred_test,y_test))
print("Cross Validation result is: ",cross_val_score(ensemble, xv, y, cv=10, scoring ='accu
cm=confusion_matrix(y_test,pred_test)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(classification_report(y_test, pred_test))
```

```
Bias is :  0.011750000000000038
Variance is:  0.06100000000000054
Accuracy is:  0.939
Cross Validation result is:  0.9296000000000001
Confusion matrix, without normalization
              precision    recall  f1-score   support

           0       0.93      0.95      0.94       518
           1       0.95      0.92      0.94       482

    accuracy                           0.94      1000
   macro avg       0.94      0.94      0.94      1000
weighted avg       0.94      0.94      0.94      1000
```

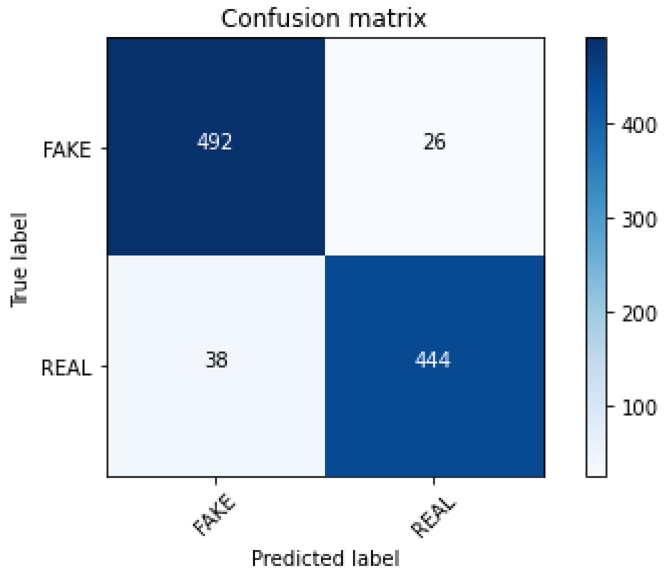


(b) STACKING

In [38]:

```
from mlxtend.classifier import StackingClassifier

base1=SVC()
base2=KNeighborsClassifier(n_neighbors=3)
meta_model=LogisticRegression()

stack=StackingClassifier(classifiers=[base1,base2],meta_classifier=meta_model)
stack.fit(xv_train,y_train)

pred_train = stack.predict(xv_train)
pred_test = stack.predict(xv_test)
print("Bias is : ",1-accuracy_score(pred_train,y_train))
print("Variance is: ",1-accuracy_score(pred_test,y_test))
print("Accuracy is: ",accuracy_score(pred_test,y_test))
print("Cross Validation result is: ",cross_val_score(stack, xv, y, cv=10, scoring ='accurac
cm=confusion_matrix(y_test,pred_test)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(classification_report(y_test, pred_test))
```

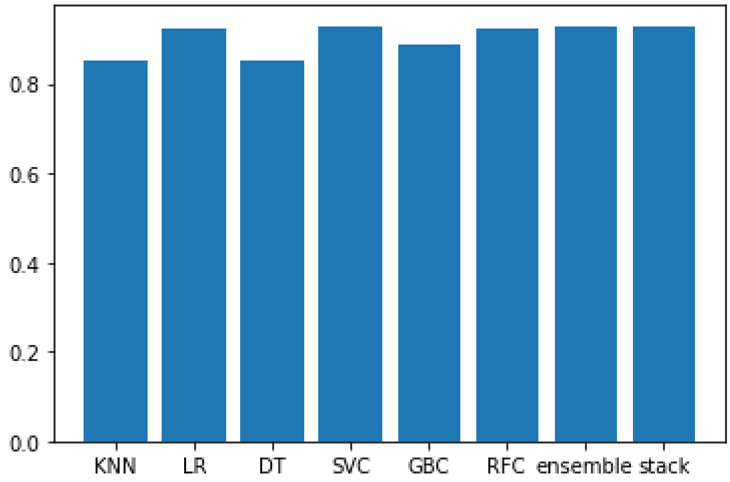
```
Bias is :  0.0014999999999999458
Variance is:  0.06399999999999995
Accuracy is:  0.936
Cross Validation result is:  0.9296000000000001
Confusion matrix, without normalization
              precision    recall  f1-score   support

           0       0.93      0.95      0.94       518
           1       0.94      0.92      0.93       482

    accuracy                           0.94      1000
   macro avg       0.94      0.94      0.94      1000
weighted avg       0.94      0.94      0.94      1000
```



PLOTTING BAR GRAPH OF CROSS VALIDATION SCORES

In [39]:

```
x_coordinates = ['KNN', 'LR', 'DT', 'SVC', 'GBC', 'RFC', 'ensemble', 'stack']
y1=cross_val_score(knn, xv, y, cv=10, scoring ='accuracy').mean()
y2=cross_val_score(LR, xv, y, cv=10, scoring ='accuracy').mean()
y3=cross_val_score(DT, xv, y, cv=10, scoring ='accuracy').mean()
y4=cross_val_score(svc, xv, y, cv=10, scoring ='accuracy').mean()
y5=cross_val_score(GBC, xv, y, cv=10, scoring ='accuracy').mean()
y6=cross_val_score(RFC, xv, y, cv=10, scoring ='accuracy').mean()
y7=cross_val_score(ensemble, xv, y, cv=10, scoring ='accuracy').mean()
y8=cross_val_score(stack, xv, y, cv=10, scoring ='accuracy').mean()
y_coordinates = [y1,y2,y3,y4,y5,y6,y7,y8]
plt.bar(x_coordinates, y_coordinates)
plt.show()
```



MAKING MANUAL PREDICTION

In [45]:

```
def output(n):
    if n == 0:
        return "Fake News"
    elif n == 1:
        return "Not A Fake News"

def manual_testing(news):
    testing_news = {"title":[news]}
    new_def_test = pd.DataFrame(testing_news)
    new_def_test["title"] = new_def_test["title"].apply(conversion)
    new_x_test = new_def_test["title"]
    new_xv_test = vectorization.transform(new_x_test)
    pred_ensemble = ensemble.predict(new_xv_test)

    return print("Prediction: {}",output(pred_ensemble[0]))
```

In [48]:

```
news = str(input())
```

```
McPain: John McCain Furious That Iran Treated US Sailors Well
```

In [49]:

```
manual_testing(news)
```

```
Prediction: {} Fake News
```

In [52]:

```
news = str(input())
```

```
Moose Wala family writes to Amit Shah seeking probe by central agency in kil
ling
```

In [53]:

```
manual_testing(news)
```

```
Prediction: {} Not A Fake News
```