Distinguishing News From Fake News

Chongyi Xu
CSE 415, Fall 2017
Assignment 7 Report
Univerisity of Washington

1 Introduction

1.1 Definition

Fake news problem refers to fabricated news or plain government propaganda with huge ramifications in the current society. In general, fake news is explained in 7 types of fake content according to First Draft News, the organization dedicated to improving skills and standards in the reporting and sharing of online information. False connection, false context, manipulated content, satire or parody, misleading content, imposter content, and fabricated content.

1.2 History

In 19th century, one example of the fake news the *Greate Moon Hoax* of 1835. The New York Sun published articles about a real-life astronomer and a made-up colleague who, according to the hoax, had observed bizarre life on the moon. The fictionalized articles successfully attracted new subscribers, and the penny paper suffered very little backlash after it admitted the next month that the series had been a hoax. Such stories were intended to entertain readers, and not to mislead them.

20th century during WWI, one of the most notorious fake news was that of an alleged German Corpse Factory in which the German battlefield dead were rendered down for fats used to make nitroglycerine, candles, lubricants, human soap, and boot dubbing.Unfounded rumors regarding such a factory circulated in the Allied press starting in 1915, and by 1917 the English-language publication North China Daily News presented these allegations as true at a time when Britain was trying to convince China to join the Allied war effort; this was based on new, allegedly true stories from The Times and The Daily Mail that turned out to be forgeries. These false allegations became known as such after the war, and in the Second World War Joseph Goebbels used the story in order to deny the ongoing massacre of Jews as British propaganda. According to Joachim Neander and Randal Marlin, the story also ëncouraged later disbeliefẅhen reports about the Holocaust surfaced after the liberation of Auschwitz and Dachau concentration camps.

2 Formulation

My problem formulation is identical to George McIntire's work based on opendatascience. The data gathering is in two parts. The first part is getting fake news part, which generally focusing on importing the fake news dataset from Kaggle with 13,000 articles published during the 2016 election cycle. The second part is to get "real" news from AllSlides, according to George McIntire's blog, he ended up scraping a total of 5279 articles came from media organizations that were published in 2015 or 2016.

3 Techniques Used

I basically chosed 4 classifiers to implement and compare. My purpose is the find the "better" classifier for solving Fake News Problems.

3.1 Naive Bayes

Navice Bayes Classifier is a probabilistic classifier based on applying Bayes' theorm with naive independece assumptions. It combines the Bayes' probability model with a decision rule. One common rules is to pick the hypothesis that is most probable; that is known as the maximum a posteriori or MAP decision. In this case, I am going to develop a multinomial naive Bayes, where vectors represent the frequencies with which certain events have been generated by a multinomial $(p_1, \ldots, p_n)$ where p_i is the probability that event i occurs

3.2 Support Vector Machine

Support Vector Machines (SVMs) are classifiers with associated learning algorithm that analyze data. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

3.3 Differnce & Similarity

Naive Bayes Classifier (NBC) and Support Vector Machine (SVM) have different options including the choice of kernel function for each. They are both sensitive to parameter optimization (i.e. different parameter selection can significantly change their output) . So, if you have a result showing that NBC is performing better than SVM. This is only true for the selected parameters. However, for another parameter selection, you might find SVM is performing better.
In general, if the assumption of independence in NBC is satisfied by the variables of your dataset and the degree of class overlapping is small (i.e. potential linear decision boundary), NBC would be expected to achieve good. For some datasets, with optimization using wrapper feature selection, for example, NBC may defeat other classifiers. Even if it achieves a comparable performance, NBC will be more desirable because of its high speed.

3.4 Other Classifiers

Besides NBC and SVM, I am also considering other 2 classifiers to compare results.

- Decision Tree
- K Nearest Neighbors

4 Training and Testing Data Used

The data I am going to use is from datacamp. The table has the following columns

- *Unnamed: 0:* unique id
- *title:* title of an article
- *text:* content of an article
- *label:* the label telling if the data is fake or real

And sample table is like

	Unnamed: 0	title	text	label
0	8476	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	10294	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE
2	3608	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	10142	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
4	875	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL
5	6903	Tehran, USA	\nI'm not an immigrant, but my grandparents ...	FAKE
6	7341	Girl Horrified At What She Watches Boyfriend D...	Share This Baylee Luciani (left), Screenshot o...	FAKE
7	95	'Britain's Schindler' Dies at 106	A Czech stockbroker who saved more than 650 Je...	REAL
8	4869	Fact check: Trump and Clinton at the 'commande...	Hillary Clinton and Donald Trump made some ina...	REAL
9	2909	Iran reportedly makes new push for uranium con...	Iranian negotiators reportedly have made a las...	REAL

And the dataset has 6335 rows with 4 colunmns.

5 Experiments

5.1 Seprate Training Data & Testing Data

Since the dataset itself has more than 5000 observations, I decided to seperate the set into training set and testing set. The seperation method I am using is randomly pick 1/3 of the data as testing set. In order to keep my training set and testing set consistent every time, I picked a seed number.
With random seperation, I got two set of tuple data, (x_train, y_train) and (x_test, y_test) where training set contains 4244 rows and testing set contains 2091 rows. Some sample training data is like

Unnamed: 0	text
4856	Donald Trump threatened to sue the New York Ti...
5323	Planned Parenthood: Abortion pill usage now ri...
4265	In a last dash, final "hail mary" attempt to e...
1697	Washington (CNN) Donald Trump and Ben Carson n...
3809	The Obama administration announced Friday it w...
8717	Three local military veterans to receive rec...
10396	Home This Month Popular What The Trump Skeptic...
269	But then the sobering realization sets in: the...
1387	Killing Obama administration rules, dismantlin...
2139	The party looks to Kamala Harris, Catherine Co...

And in each dataset, 50% of the news are real and about the other half are fake. And the csv file is 29.2MB which is an ideal input size to run on personal laptop.

5.2 Vectorize Data

In order to apply classifier, it is neccessary to change the data table into vectors.

5.2.1 tf-idf

TFIDF or tf-idf, short of term frequency-inverse document frequency, is a numerical statistic that reflects how important a word is to a document (article) in a collection or corpus.

5.2.2 CountVectorizer

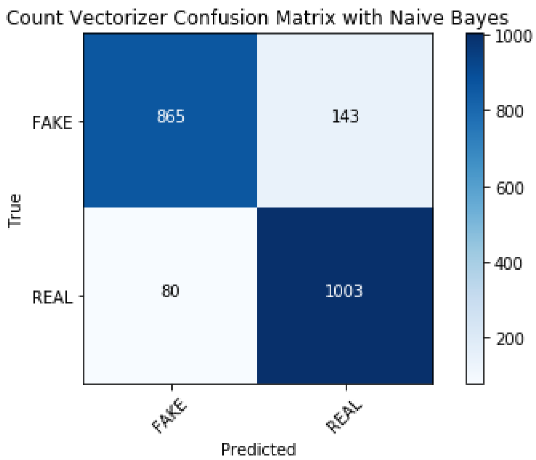
CountVectorizer counts the word frequencies in the article.

6 Results

6.1 Naive Bayes Classifier

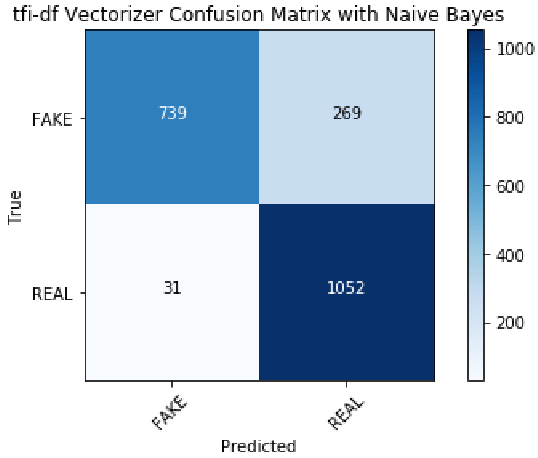
6.1.1 CountVectorizer

f1 score: 0.89314



6.1.2 TFIDF

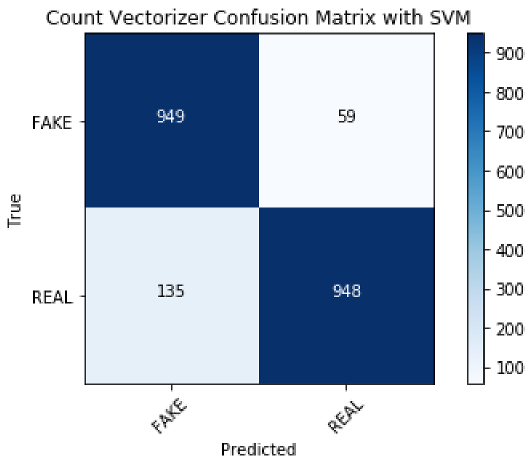
f1 score: 0.85403



6.2 Support Vector Machine(SVM)

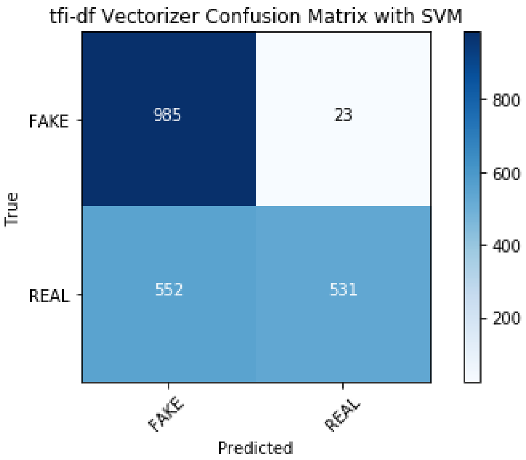
6.2.1 CountVectorizer

f1 score: 0.90722



6.2.2 TFIDF

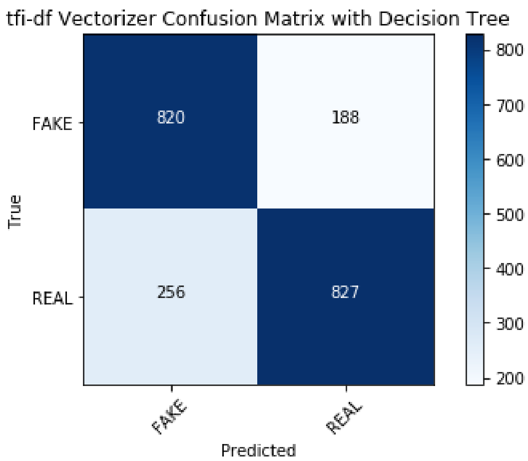
f1 score: 0.70916



6.3 Decision Tree

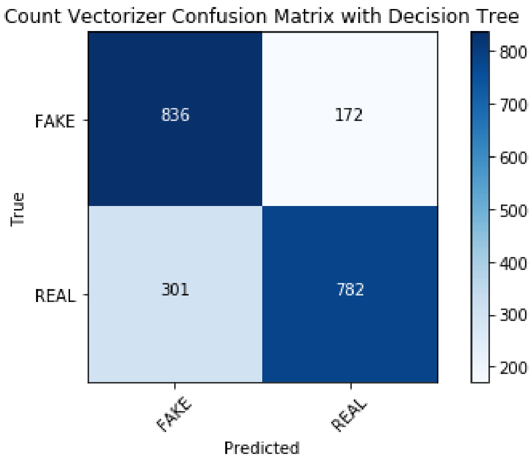
6.3.1 CountVectorizer

f1 score: 0.77441



6.3.2 TFIDF

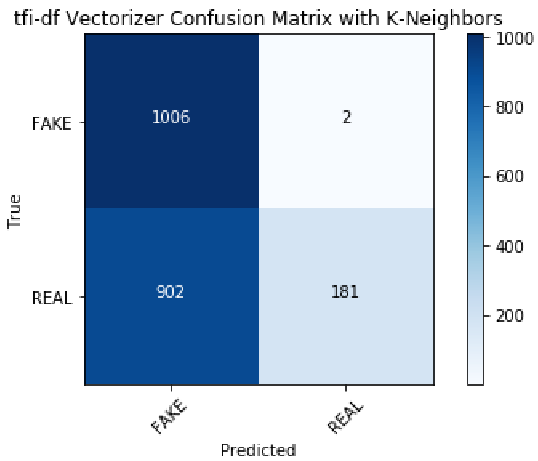
f1 score: 0.78912



6.4 KNeighbors Classifier

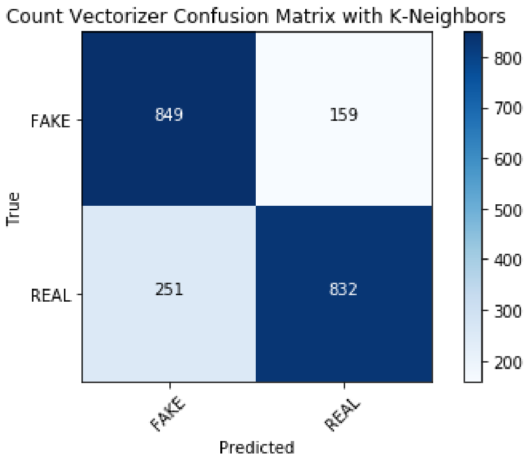
6.4.1 CountVectorizer

f1 score: 0.80385



6.4.2 TFIDF

f1 score: 0.48072



6.5 Comparison of F1 Scores

f1_score table	Count Vecotizer	tfi-df Vectorizer
Naive Bayes	0.89314	0.85403
Support Vector Machine	0.90722	0.70916
Classification Trees	0.77441	0.78912
KNeighbors	0.80385	0.48072

7 Discussion

Currently, society is suffering "Fake News" among various social media. It is our duty to distinguish those fakes from truth. In the process of formulating the problem, it took me a long time to figure out the real news for the import of my training data.
As a result, I found that generlly, SVM could give a better result on distinguishing fake and real news when we vectorize our data considering the appearence of every words. Meanwhile, if we vectorize the text considering the importance of every words in the text content, Naive Bayes Classifier might perform better.

8 Personal Retrospective

In this assignment, I have learned more about the Fake News Problem, such as how important it is to the society and the difficulty to solve for it. Fake News could affect a lot to politics, natural science, economics and all the way to every single field. I also learned the pros and cons of data analyzing classifiers, for instance, naive Bayes, K Nearest Neighborsr, Decision Tree, and Support Vector Machines. It is always a good idea to test all classifiers since we could not be familier to the processing dataset everytime to know which classifier should be used.

References

[1] A new database of fake news sites details how much fakery has spread from Trump v. Clinton to local news by LAURA HAZARD OWEN @laurahazardowen April 21, 2017, 10:17 a.m. http://www.niemanlab.org/2017/04/a-new-database-of-fake-news-sites-details-how-much-fakery-has-spread-from-trump-v-clinton-to-local-news/

[2] Social Media and Fake News in the 2016 Election Journal of Economic Perspectives—Volume 31, Number 2—Spring 2017—Pages 211–236 https://web.stanford.edu/~gentzkow/research/fakenews.pdf

[3] A comparison of a several classifiers in scikit-learn on synthetic datasets. Illustrating the nature of decision boundaries of different classifiers. http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

[4] When does Naive Bayes perform better than SVM? https://stats.stackexchange.com/questions/58214/when-does-naive-bayes-perform-better-than-svm

Report

December 6, 2017

1 Distinguishing News From Fake News

Due December 9

Chongyi Xu, 1531273
CSE 415 Fall 2017, Assignment 7

```
In [49]: #load any required packages here
         %matplotlib inline
         import time
         import itertools
         import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         from sklearn.metrics import confusion_matrix
         from sklearn.metrics import f1_score
         from sklearn.model_selection import train_test_split
         from sklearn.feature_extraction.text import CountVectorizer
         from sklearn.feature_extraction.text import TfidfVectorizer
```

1.1 1. Importing Data

First, we want to import the fake news table from the source I found. The data I found is from https://github.com/GeorgeMcIntire/fake_real_news_dataset

According to the dataset documentation, text and metadata from fake news sites:

- **Unnamed: 0:** unique id
- **title:** title of an article
- **text:** content of an article
- **label:** label telling if the news is fake or real

```
In [50]: data = pd.read_csv("fake_or_real_news.csv")
         fake_news = data.set_index("Unnamed: 0")
         y = fake_news.label
         fake_news = fake_news.drop("label", axis=1)
         data.head(10)
```

```
Out[50]:    Unnamed: 0                                              title  \
         0        8476                       You Can Smell Hillarys Fear
         1       10294  Watch The Exact Moment Paul Ryan Committed Pol...
         2        3608        Kerry to go to Paris in gesture of sympathy
         3       10142  Bernie supporters on Twitter erupt in anger ag...
         4         875   The Battle of New York: Why This Primary Matters
         5        6903                                        Tehran, USA
         6        7341  Girl Horrified At What She Watches Boyfriend D...
         7          95                Britains Schindler Dies at 106
         8        4869  Fact check: Trump and Clinton at the 'commande...
         9        2909  Iran reportedly makes new push for uranium con...

                                                        text label
         0  Daniel Greenfield, a Shillman Journalism Fello...  FAKE
         1  Google Pinterest Digg Linkedin Reddit Stumbleu...  FAKE
         2  U.S. Secretary of State John F. Kerry said Mon...  REAL
         3   Kaydee King (@KaydeeKing) November 9, 2016 T...  FAKE
         4  It's primary day in New York and front-runners...  REAL
         5    \nIm not an immigrant, but my grandparents ...  FAKE
         6  Share This Baylee Luciani (left), Screenshot o...  FAKE
         7  A Czech stockbroker who saved more than 650 Je...  REAL
         8  Hillary Clinton and Donald Trump made some ina...  REAL
         9  Iranian negotiators reportedly have made a las...  REAL
```

```
In [51]: print(data.shape)
```

```
(6335, 4)
```

1.2 2. Seperating Training Data & Testing Data

Using random to seperate training data and testing data

```
In [52]: x_train, x_test, y_train, y_test = train_test_split(fake_news['text'], y, test_size=0
```

```
In [76]: pd.DataFrame(x_test).head(10)
```

```
Out[76]:                                                          text
         Unnamed: 0
         4856        Donald Trump threatened to sue the New York Ti...
         5323        Planned Parenthood: Abortion pill usage now ri...
         4265        In a last dash, final "hail mary" attempt to e...
         1697        Washington (CNN) Donald Trump and Ben Carson n...
         3809        The Obama administration announced Friday it w...
         8717          Three local military veterans to receive rec...
         10396       Home This Month Popular What The Trump Skeptic...
         269         But then the sobering realization sets in: the...
         1387        Killing Obama administration rules, dismantlin...
         2139        The party looks to Kamala Harris, Catherine Co...
```

1.3 3. Vectorizing Data

- **TfidfVectorizer()**

```
In [53]: vectorizer_tfidf = TfidfVectorizer(stop_words='english', max_df=0.8)
         tfidf_train = vectorizer_tfidf.fit_transform(x_train)
         tfidf_test = vectorizer_tfidf.transform(x_test)
```

- **CountVectorizer()**

```
In [54]: vectorizer_count = CountVectorizer(stop_words='english')
         count_train = vectorizer_count.fit_transform(x_train)
         count_test = vectorizer_count.transform(x_test)
```

```
In [55]: # Get the feature names of `tfidf_vectorizer`
         print(vectorizer_tfidf.get_feature_names()[-10:])

         # Get the feature names of `count_vectorizer`
         print(vectorizer_count.get_feature_names()[:10])
```

```
['', '', '', '', '', '', '', '', '', 'ade']
['00', '000', '0000', '00000031', '000035', '00006', '0001', '0001pt', '000ft', '000km']
```

1.4 4. Building Confusion Matrix Plot

```
In [56]: def plot_confusion_matrix(cm, classes,
                                   title='Confusion matrix',
                                   cmap=plt.cm.Blues):
             """
             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)

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

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

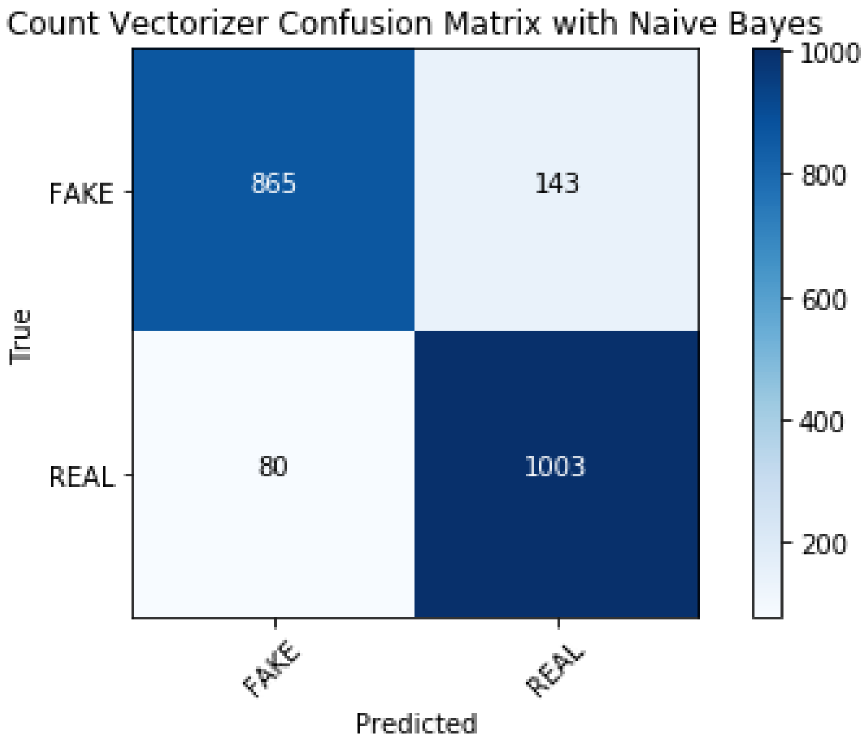
1.5 5. Building Classifier Models

- **Multinomial Naive Bayes**

In [57]:
```
from sklearn.naive_bayes import MultinomialNB
```

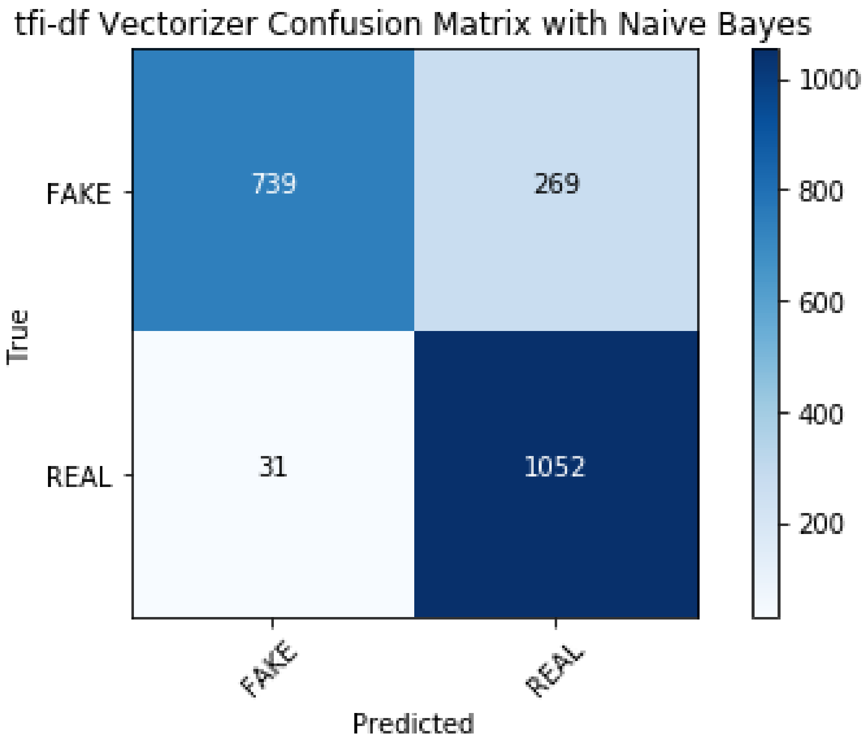
In [85]:
```
clf = MultinomialNB()
clf.fit(count_train, y_train)
pred = clf.predict(count_test)
f1_nb_count = f1_score(y_test, pred, average='weighted')
print("f1_score: %0.5f" % f1_nb_count)
cm_nb_count = confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm_nb_count, classes=['FAKE', 'REAL'], title='Count Vectorizer C
```

```
f1_score: 0.89314
```



In [86]:
```
clf = MultinomialNB()
clf.fit(tfidf_train, y_train)
pred = clf.predict(tfidf_test)
f1_nb_tfidf = f1_score(y_test, pred, average='weighted')
print("f1_score: %0.5f" % f1_nb_tfidf)
cm_nb_tfidf = confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm_nb_tfidf, classes=['FAKE', 'REAL'], title='tfi-df Vectorizer
```

```
f1_score: 0.85403
```

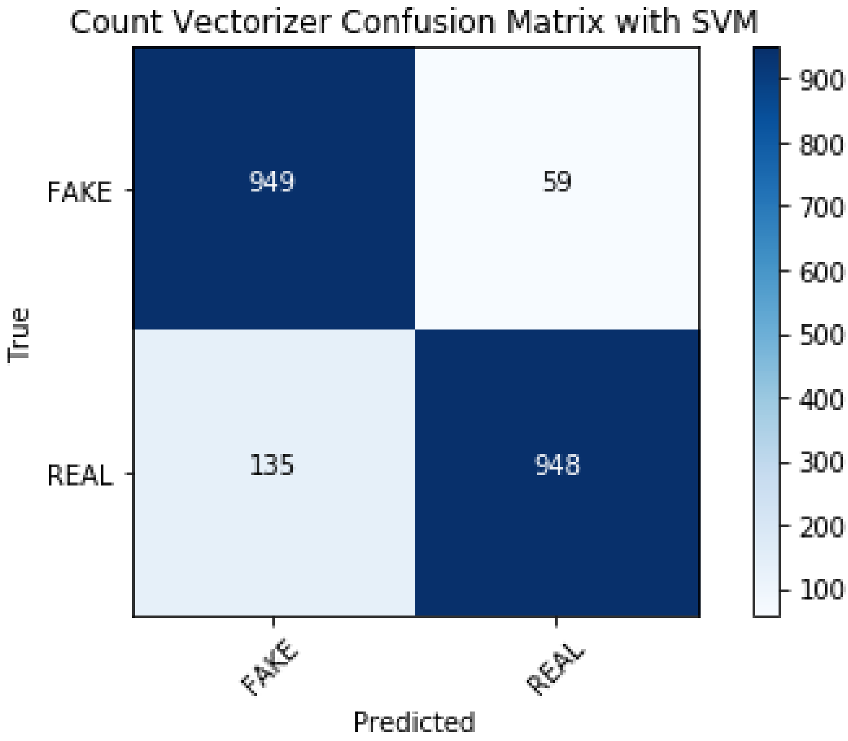


- **Support Vector Machine**

```
In [60]: from sklearn.svm import SVC
```

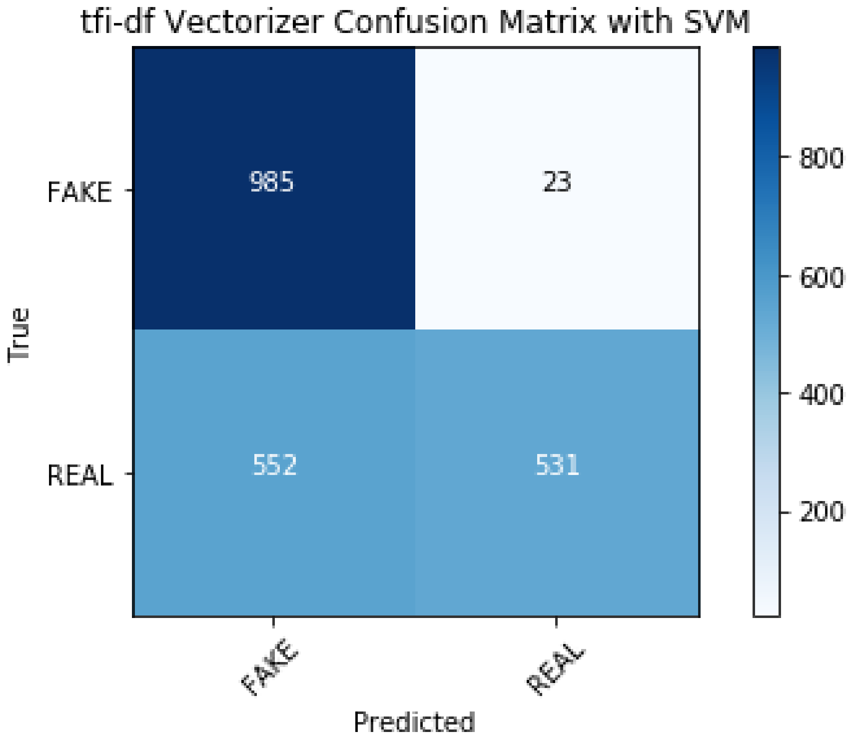
```
In [87]: clf = SVC(kernel="linear", C=0.025)
         clf.fit(count_train, y_train)
         pred = clf.predict(count_test)
         f1_svm_count = f1_score(y_test, pred, average='weighted')
         print("f1_score: %0.5f" % f1_svm_count)
         cm_svm_count = confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
         plot_confusion_matrix(cm_svm_count, classes=['FAKE', 'REAL'], title='Count Vectorizer
```

```
f1_score: 0.90722
```




```
In [83]: clf = SVC(kernel="linear", C=0.025)
         clf.fit(tfidf_train, y_train)
         pred = clf.predict(tfidf_test)
         f1_svm_tfidf = f1_score(y_test, pred, average='weighted')
         print("f1_score: %0.5f" % f1_svm_tfidf)
         cm_svm_tfidf = confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
         plot_confusion_matrix(cm_svm_tfidf, classes=['FAKE', 'REAL'], title='tfi-df Vectorize
```

```
f1_score: 0.70916
```

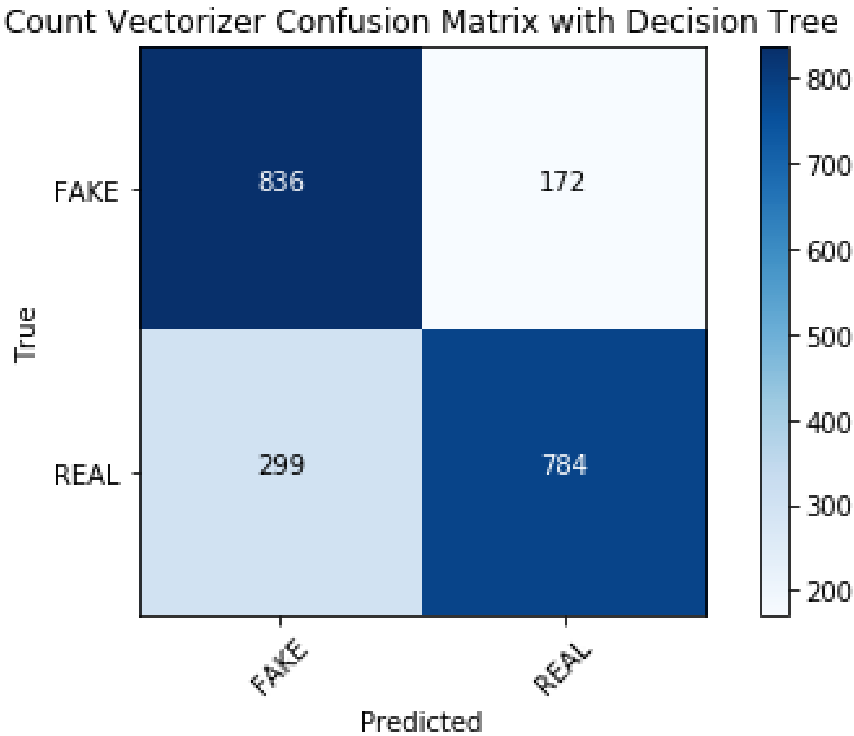


- **Classification Tree**

```
In [63]: from sklearn.tree import  DecisionTreeClassifier
```

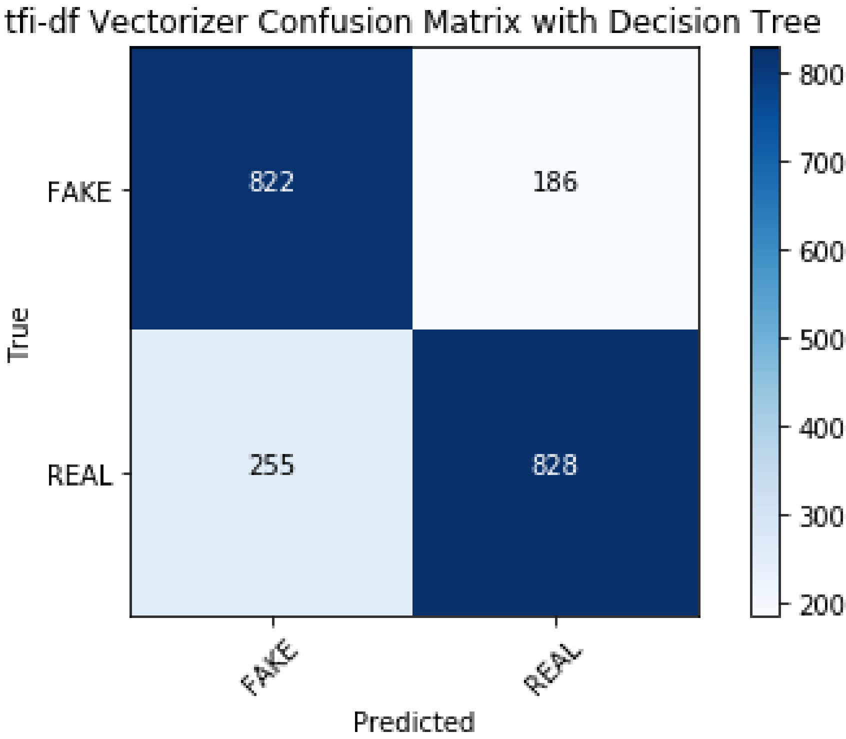
```
In [82]: clf = DecisionTreeClassifier(max_depth=5)
         clf.fit(count_train, y_train)
         pred = clf.predict(count_test)
         f1_tree_count = f1_score(y_test, pred, average='weighted')
         print("f1_score: %0.5f" % f1_tree_count)
         cm_tree_count = confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
         plot_confusion_matrix(cm_tree_count, classes=['FAKE', 'REAL'], title='Count Vectorize
```

```
f1_score: 0.77441
```



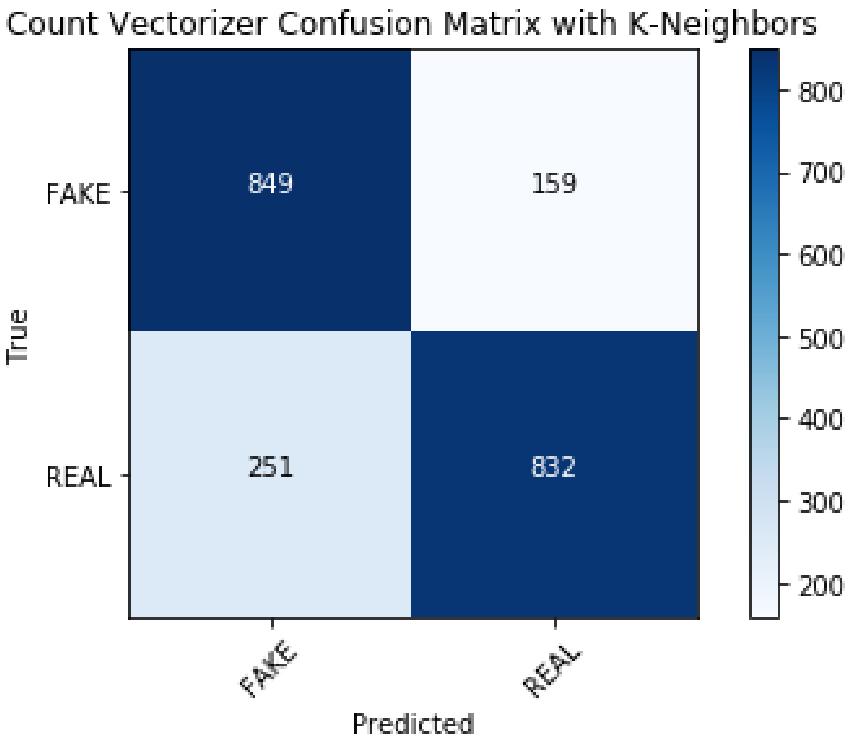

```
In [81]: clf = DecisionTreeClassifier(max_depth=5)
         clf.fit(tfidf_train, y_train)
         pred = clf.predict(tfidf_test)
         f1_tree_tfidf = f1_score(y_test, pred, average='weighted')
         print("f1_score: %0.5f" % f1_tree_tfidf)
         cm_tree_tfidf = confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
         plot_confusion_matrix(cm_tree_tfidf, classes=['FAKE', 'REAL'], title='tfi-df Vectoriz
```

```
f1_score: 0.78912
```



- **KNeighbors Classifier**

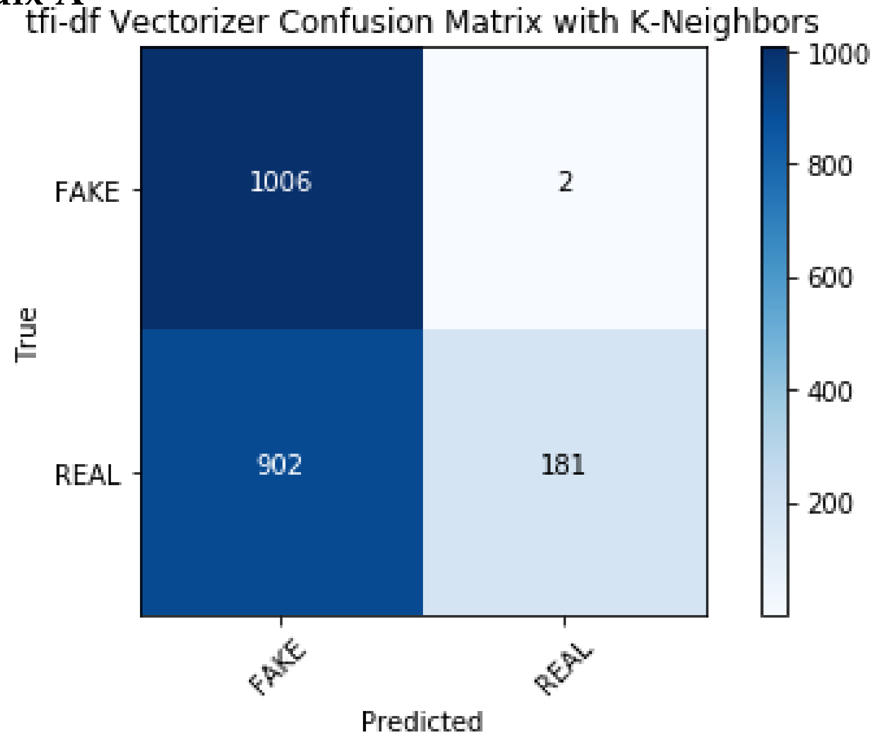
```
In [66]: from sklearn.neighbors import KNeighborsClassifier

In [80]: clf =  KNeighborsClassifier(3)
         clf.fit(count_train, y_train)
         pred = clf.predict(count_test)
         f1_knn_count = f1_score(y_test, pred, average='weighted')
         print("f1_score: %0.5f" % f1_knn_count)
         cm_knn_count = confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
         plot_confusion_matrix(cm_knn_count, classes=['FAKE', 'REAL'], title='Count Vectorizer

f1_score: 0.80385
```



In [79]:
```
clf =  KNeighborsClassifier(3)
clf.fit(tfidf_train, y_train)
pred = clf.predict(tfidf_test)
f1_knn_tfidf = f1_score(y_test, pred, average='weighted')
print("f1_score: %0.5f" % f1_knn_tfidf)
cm_knn_tfidf = confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm_knn_tfidf , classes=['FAKE', 'REAL'], title='tfi-df Vectoriz
```

```
f1_score: 0.48072
```

A Appendix A



1.6 6. Comparing Methods

f1_score table	Count Vecotizer	tfi-df Vectorizer
Naive Bayes	0.89314	0.85403
Support Vector Machine	0.90722	0.70916
Classification Trees	0.77441	0.78912
KNeighbors	0.80385	0.48072