In [2]:
```
import pandas as pd
import numpy as np
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
import re
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

In [6]:
```
df = pd.read_csv('C:\\Users\\Admin\\OneDrive\\Desktop\\news.csv')
```

In [8]:
```
df.head()
```

Out[8]:

	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

In [10]:
```
df.tail()
```

Out[10]:

	Unnamed: 0	title	text	label
6330	4490	State Department says it can't find emails fro...	The State Department told the Republican Natio...	REAL
6331	8062	The 'P' in PBS Should Stand for 'Plutocratic' ...	The 'P' in PBS Should Stand for 'Plutocratic' ...	FAKE
6332	8622	Anti-Trump Protesters Are Tools of the Oligarc...	Anti-Trump Protesters Are Tools of the Oligar...	FAKE
6333	4021	In Ethiopia, Obama seeks progress on peace, se...	ADDIS ABABA, Ethiopia —President Obama convene...	REAL
6334	4330	Jeb Bush Is Suddenly Attacking Trump. Here's W...	Jeb Bush Is Suddenly Attacking Trump. Here's W...	REAL

In [12]:
```
df.shape
```

Out[12]:
```
(6335, 4)
```

In [44]:
```
X = df.drop('label', axis=1)
X['content'] = X['title'] +' '+ X['text']
X.head()
```

Out[44]:

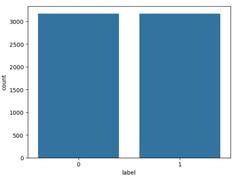
	Unnamed: 0	title	text	content
0	8476	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	You Can Smell Hillary's Fear Daniel Greenfield...
1	10294	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg LinkedIn Reddit Stumbleu...	Watch The Exact Moment Paul Ryan Committed Pol...
2	3608	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	Kerry to go to Paris in gesture of sympathy U....
3	10142	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	Bernie supporters on Twitter erupt in anger ag...
4	875	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	The Battle of New York: Why This Primary Matte...

In [46]:
```
# Converting target class to 0 and 1
df['label'] = df['label'].apply(lambda x: 1 if x=='FAKE' else 0)
y = df['label']
y.head()
```

Out[46]:
```
0    1
1    1
2    0
3    1
4    0
Name: label, dtype: int64
```

In [50]:
```
# Checking true and false values
import seaborn as sns
sns.countplot(x = df['label'])
```

Out[50]:
```
<Axes: xlabel='label', ylabel='count'>
```



In [52]:
```
# Preparing training and test data
X_train, X_test, y_train, y_test = train_test_split(X['content'], y, test_size=0.2, random_state=101)
```

In [57]:
```
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
vect= CountVectorizer(stop_words='english',max_features=100)
x_train_count = vect.fit_transform(X_train)
x_test_count = vect.fit_transform(X_test)
```

In [61]:
```
tfidf_vect = TfidfVectorizer(stop_words='english',max_features=10000)
x_train_tf = tfidf_vect.fit_transform(X_train)
x_test_tf = tfidf_vect.fit_transform(X_test)
```

In [63]:
```
import scipy.sparse as sp
train_data = sp.hstack((x_train_count,x_train_tf))
test_data = sp.hstack((x_test_count,x_test_tf))
```

In [65]:
```
# Let us apply now different classification algorithms

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.linear_model import SGDClassifier
from sklearn.neighbors import KNeighborsClassifier
# from sklearn.naive_bayes import GaussianNB
from sklearn.pipeline import Pipeline
from datetime import datetime
```

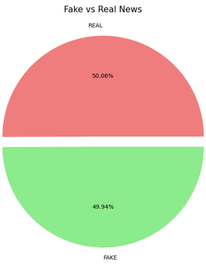
In [69]:
```
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
models = {'Logistic Regression':LogisticRegression(),
          'Support Vector Classifier':SVC(),'Decision Tree':DecisionTreeClassifier(),
          'Random Forest':RandomForestClassifier(),'Neural Network':MLPClassifier(),
          'Stochastic Gradient Descent':SGDClassifier(), 'KNN':KNeighborsClassifier()}

for model, algorithm in models.items():
    start_time = datetime.now()
    pipe = Pipeline([('model', algorithm)])
    pipe.fit(train_data ,y_train)
    end_time = datetime.now()
    prediction = pipe.predict(test_data)
    print("\n %s ======= For {} ==========".format(model))
    print('Accuracy Score : {} '.format(accuracy_score(prediction,y_test)))
    print('Confusion Matrix \n\n ',confusion_matrix(prediction,y_test))
    print('\n Classification Report \n ')
    print(classification_report(prediction,y_test))
    time_difference = (end_time - start_time).total_seconds() * 10**3
    print("Execution time of program is: ", time_difference, "ms")
```

```
E:\Users\Admin\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:469: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

```
 ======= For Logistic Regression ==========
Accuracy Score : 0.5911602209944752
Confusion Matrix

  [[166  31]
 [487 583]]

 Classification Report

              precision    recall  f1-score   support

           0       0.25      0.84      0.39       197
           1       0.95      0.54      0.69      1070

    accuracy                           0.59      1267
   macro avg       0.60      0.69      0.54      1267
weighted avg       0.84      0.59      0.65      1267

Execution time of program is:  550.176 ms


 ======= For Support Vector Classifier ==========
Accuracy Score : 0.6893472770323599
Confusion Matrix

  [[317  69]
 [336 545]]

 Classification Report

              precision    recall  f1-score   support

           0       0.49      0.82      0.61       386
           1       0.89      0.62      0.73       881

    accuracy                           0.68      1267
   macro avg       0.69      0.72      0.67      1267
weighted avg       0.77      0.68      0.69      1267

Execution time of program is:  27922.762 ms


 ======= For Decision Tree ==========
Accuracy Score : 0.5556432517758485
Confusion Matrix

  [[245 155]
 [408 459]]

 Classification Report

              precision    recall  f1-score   support

           0       0.38      0.61      0.47       400
           1       0.75      0.53      0.62       867

    accuracy                           0.56      1267
   macro avg       0.56      0.57      0.54      1267
weighted avg       0.63      0.56      0.57      1267

Execution time of program is:  7027.806 ms


 ======= For Random Forest ==========
Accuracy Score : 0.6685082872928176
Confusion Matrix

  [[314  81]
 [339 533]]

 Classification Report

              precision    recall  f1-score   support

           0       0.48      0.79      0.60       395
           1       0.87      0.61      0.72       872

    accuracy                           0.67      1267
   macro avg       0.67      0.70      0.66      1267
weighted avg       0.75      0.67      0.68      1267

Execution time of program is:  8996.038 ms


 ======= For Noural Network ==========
Accuracy Score : 0.5990709550118839
Confusion Matrix

  [[198  64]
 [455 550]]

 Classification Report

              precision    recall  f1-score   support

           0       0.30      0.76      0.43       262
           1       0.90      0.55      0.68      1005

    accuracy                           0.59      1267
   macro avg       0.60      0.65      0.56      1267
weighted avg       0.77      0.59      0.63      1267

Execution time of program is:  64416.738 ms


 ======= For Stochastic Gradient Descent ==========
Accuracy Score : 0.6116811365439315
Confusion Matrix

  [[216  55]
 [437 559]]

 Classification Report

              precision    recall  f1-score   support

           0       0.33      0.80      0.47       271
           1       0.91      0.56      0.69       996

    accuracy                           0.61      1267
   macro avg       0.62      0.68      0.58      1267
weighted avg       0.79      0.61      0.65      1267

Execution time of program is:  224.102 ms


 ======= For KNN ==========
Accuracy Score : 0.5895816890292028
Confusion Matrix

  [[237  84]
 [436 530]]

 Classification Report

              precision    recall  f1-score   support

           0       0.33      0.72      0.45       301
           1       0.86      0.55      0.67       966

    accuracy                           0.59      1267
   macro avg       0.60      0.63      0.56      1267
weighted avg       0.74      0.59      0.62      1267

Execution time of program is:  5.207 ms
```

In [71]:
```
# remove column 'Unnamed: 0' from the analysis
df.drop(columns=['Unnamed: 0'], axis=1, inplace=True)
```

In [83]:
```
# check the distribution of label (fake vs Real news)
import matplotlib.pyplot as plt
labels = df['label'].value_counts()
plt.figure(figsize=(10,8))
plt.pie(labels.values, labels=labels.index, autopct='%.2f%%', explode=[0.05]*len(labels.index)), colors=['lightcoral','lightgreen'])
plt.title('Fake vs Real News', fontsize=15)
plt.show()
```



In [ ]: