# Fake News Detection Project Report

March 2020 – 2021

## Introduction

"Fake news" is "fabricated information that mimics news media content in form but not in organizational process or intent. It has been existing for a long time and becomes a common problem we face every day since the advent of internet. The phenomenon has becomes more evident when social media are increasingly popular.

This case study is to build models to detect fake news from a given dataset, to evaluate the accuracy of different models, and to perform a prediction.

## Methodology

2 datasets are given for this case study: training and testing datasets. Since they are labelled and time independent, I use supervised machine learning to study the problem. I inspect, visualize, and clean the training dataset to see their patterns and characteristics. Then I deploy various models to find out which one is the most suitable for the study case. Finally I test the best model with the testing dataset to see how the prediction is.

## Data Description

The training dataset contains 15000 rows and  6 columns. The 6 variables are:

- *id*: unique id for a news article
- *title*: the title of a news article
- *author*: author of the news article
- *text*: the text of the article; could be incomplete
- *label*: a label that marks the article as potentially unreliable
    - 1: unreliable

- 0: reliable

| | Unnamed: 0 | id | title | author | text | label |
|---|---|---|---|---|---|---|
| 0 | 16285 | 16284 | The Voting Dead: CBS4 Investigation Exposes Vo... | Jenna Leigh Richardson | A CBS Denver voter fraud investigation has exp... | 1 |
| 1 | 16573 | 16572 | How to Suppress the Apology Reflex - The New Y... | Audrey S. Lee | Editors' note: Here's one of our favorite stor... | 0 |
| 2 | 1001 | 1000 | Breitbart News Daily: Gorsuch for SCOTUS - Bre... | Breitbart News | On the Wednesday edition of Breitbart News Dai... | 0 |
| 3 | 13129 | 13128 | Found this nugget in Podesta files Fastwalkers... | seentoomuch | Found this nugget in Podesta files Fastwalkers... | 1 |
| 4 | 6348 | 6347 | NaN | ctwatcher✓ᵛᵉʳⁱᶠⁱᵉᵈ | Beard or no beard I can spot them a mile away.... | 1 |
| ... | ... | ... | ... | ... | ... | ... |
| 14995 | 11065 | 11064 | Stephen Bannon, a Rookie Campaign Chief Who 'L... | Michael Barbaro and Michael M. Grynbaum | As the American financial system collapsed in ... | 0 |
| 14996 | 9474 | 9473 | New York Times: Apologist for Power | Consortiumnews.com | New York Times: Apologist for Power November 7... | 1 |
| 14997 | 1709 | 1708 | Alabama Makes It Illegal to Remove Confederate... | Ryan Saavedra | The new law attempts to preserve history by ma... | 0 |
| 14998 | 1487 | 1486 | SAY WHAT? Chicago Tribune calls on Democrats t... | BareNakedIslam | Notify me of follow-up comments by email. Noti... | 1 |
| 14999 | 11984 | 11983 | Milo: Free Speech Is 'Under Threat' on College... | Trent Baker | Following the violent protest at earlier thi... | 0 |

15000 rows × 6 columns

From the details of the dataset, we can see it contains a number of rows of *nan* values in the columns of title, author, and text. Other columns have integers only. Hence I remove those *nan* rows in order to let models successfully read the data for training and prediction.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Unnamed: 0   15000 non-null  int64 
 1   id           15000 non-null  int64 
 2   title        14609 non-null  object
 3   author       13561 non-null  object
 4   text         14976 non-null  object
 5   label        15000 non-null  int64 
dtypes: int64(3), object(3)
memory usage: 703.2+ KB
```

```
# remove rows with nan values

data_non_null = data.dropna(how = 'any', axis = 0).reset_index()
data_non_null
```

| | index | Unnamed: 0 | id | title | author | text | label |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 16285 | 16284 | The Voting Dead: CBS4 Investigation Exposes Vo... | Jenna Leigh Richardson | A CBS Denver voter fraud investigation has exp... | 1 |
| 1 | 1 | 16573 | 16572 | How to Suppress the Apology Reflex - The New Y... | Audrey S. Lee | Editors' note: Here's one of our favorite stor... | 0 |
| 2 | 2 | 1001 | 1000 | Breitbart News Daily: Gorsuch for SCOTUS - Bre... | Breitbart News | On the Wednesday edition of Breitbart News Dai... | 0 |
| 3 | 3 | 13129 | 13128 | Found this nugget in Podesta files Fastwalkers... | seentoomuch | Found this nugget in Podesta files Fastwalkers... | 1 |
| 4 | 5 | 2187 | 2186 | Despite Democrat Obstruction U.S. Senate Confi... | Warner Todd Huston | On Monday evening, Kansas Republican Mike Pomp... | 0 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 13165 | 14995 | 11065 | 11064 | Stephen Bannon, a Rookie Campaign Chief Who 'L... | Michael Barbaro and Michael M. Grynbaum | As the American financial system collapsed in ... | 0 |
| 13166 | 14996 | 9474 | 9473 | New York Times: Apologist for Power | Consortiumnews.com | New York Times: Apologist for Power November 7... | 1 |
| 13167 | 14997 | 1709 | 1708 | Alabama Makes It Illegal to Remove Confederate... | Ryan Saavedra | The new law attempts to preserve history by ma... | 0 |
| 13168 | 14998 | 1487 | 1486 | SAY WHAT? Chicago Tribune calls on Democrats t... | BareNakedIslam | Notify me of follow-up comments by email. Noti... | 1 |

After removing the *nan* values, we can now see how many rows are left and how many data belong to label 1 and 0 respectively. There are only 13170 rows of data left. The number of label 1 data is 5764 and that of label 0 data is 7406.

```python
# calculate no of news article and labels

label_plot = pd.DataFrame({'label_1': [np.sum(data_non_null['label'] == 1)], 'label_0': [np.sum(data_non_null['label']
data_len = len(data_non_null)

print('Number of news article:', data_len)
print('Number of articles labelled as unreliable (label 1):', label_plot.iloc[0, 0])
print('Number of articles labelled as reliable (label 0):', label_plot.iloc[0, 1])
```

```
Number of news article: 13170
Number of articles labelled as unreliable (label 1): 5764
Number of articles labelled as reliable (label 0): 7406
```

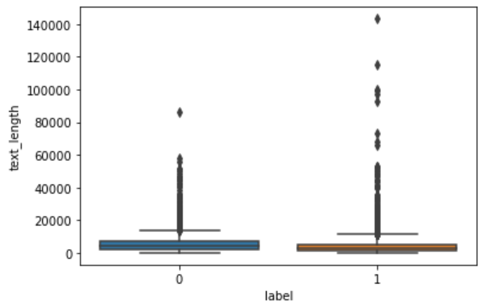
Next we calculate the text length of each text to see how they distribute by using the function *apply* which carries out the function of *len* for every row of the dataframe.

```python
# calculate text length

data_non_null['text_length'] = data_non_null['text'].apply(len)
data_non_null.head()
```

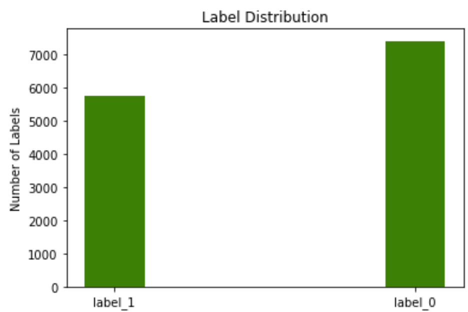
| | index | Unnamed: 0 | id | title | author | text | label | text_length |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 16285 | 16284 | The Voting Dead: CBS4 Investigation Exposes Vo... | Jenna Leigh Richardson | A CBS Denver voter fraud investigation has exp... | 1 | 1219 |
| 1 | 1 | 16573 | 16572 | How to Suppress the Apology Reflex - The New Y... | Audrey S. Lee | Editors' note: Here's one of our favorite stor... | 0 | 4739 |
| 2 | 2 | 1001 | 1000 | Breitbart News Daily: Gorsuch for SCOTUS - Bre... | Breitbart News | On the Wednesday edition of Breitbart News Dai... | 0 | 1211 |
| 3 | 3 | 13129 | 13128 | Found this nugget in Podesta files Fastwalkers... | seentoomuch | Found this nugget in Podesta files Fastwalkers... | 1 | 2758 |
| 4 | 5 | 2187 | 2186 | Despite Democrat Obstruction U.S. Senate Confi... | Warner Todd Huston | On Monday evening, Kansas Republican Mike Pomp... | 0 | 1708 |

Using boxplot we can observe that there are some texts that are relatively long, much higher than the maximum of boxplot. The maximum length of label 0 is more than 80000 and that of label 1 is more than 140000.

```python
# visualize the distribution of text length

sns.boxplot(x = 'label', y = 'text_length', data = data_non_null)
plt.show()
```




```python
# visualize the label distribution

x, y = label_plot.columns, label_plot.iloc[0]
plt.bar(x, y, width = 0.2, color = 'green')
plt.title('Label Distribution')
plt.ylabel('Number of Labels')
plt.show()
```




We can plot another graph to visualize the number of labels to see how they differ by a simple bar plot. It is obvious that there are more label 0 than label 1 in the dataset. The size of label 1 is slightly less than 6000 while that of label 0 is more than 7000, meaning there are more reliable articles than unreliable ones in the dataset.

## Data Preparation and Visualization

Before managing the dataset, we have to first define a function *cleaning* to remove unnecessary elements to lower the workload of the program. The stopword list is from the library Natural Language Toolkit which deals with classification, tokenization, stemming, tagging, parsing, and semantic reasoning for natural language processing.

*cleaning* removes extra space, punctuations, numbers, and capitalization with functions from built-in *str* library. We have to tokenize the text, i.e. breaking down a sentence into words in order to remove stopwords with the help of the stopword list. We combine the words again to make sentences for further actions.

```
# call the stopword list

stop_list = nltk.corpus.stopwords.words("english")

# define a function to clean texts

def cleaning(text):

    # removal of extra spaces
    regex_pat = re.compile(r'\s+')
    text_space = text.str.replace(regex_pat, ' ')

    # removal of punctuations and numbers
    punc_remove = text_space.str.replace("[^a-zA-Z]", " ")

    # remove whitespace with a single space
    newtext = punc_remove.str.replace(r'\s+', ' ')

    # remove leading and trailing whitespace
    newtext = newtext.str.replace(r'^\s+|\s+?$','')

    # replace normal numbers with numbr
    newtext = newtext.str.replace(r'\d+(\.\d+)?','numbr')

    # removal of capitalization
    text_lower = newtext.str.lower()

    # tokenizing
    tokenized_text = text_lower.apply(lambda x: x.split())

    # removal of stopwords
    tokenized_text = tokenized_text.apply(lambda x: [item for item in x if item not in stop_list])

    for i in range(len(tokenized_text)):
        tokenized_text[i] = ' '.join(tokenized_text[i])
        text_final = tokenized_text

    return text_final
```

Then we can compare the results before and after cleaning for the columns text, title, and author. Evidently all the unwanted elements are removed so that we can perform analysis and prediction more easily.

```
print(data_non_null[['text', 'text_clean']].head(5))

                                                text  \
0  A CBS Denver voter fraud investigation has exp...
1  Editors' note: Here's one of our favorite stor...
2  On the Wednesday edition of Breitbart News Dai...
3  Found this nugget in Podesta files Fastwalkers...
4  On Monday evening, Kansas Republican Mike Pomp...

                                          text_clean
0  cbs denver voter fraud investigation exposed d...
1  editors note one favorite stories archives fea...
2  wednesday edition breitbart news daily broadca...
3  found nugget podesta files fastwalkers dsp pro...
4  monday evening kansas republican mike pompeo c...
```

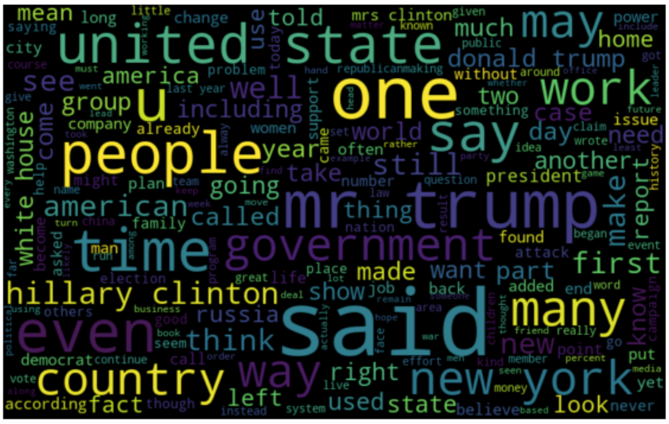
```
print(data_non_null[['title', 'title_clean']].head(5))

                                               title  \
0  The Voting Dead: CBS4 Investigation Exposes Vo...
1  How to Suppress the Apology Reflex - The New Y...
2  Breitbart News Daily: Gorsuch for SCOTUS - Bre...
3  Found this nugget in Podesta files Fastwalkers...
4  Despite Democrat Obstruction U.S. Senate Confi...

                                         title_clean
0  voting dead cbs investigation exposes voter fr...
1             suppress apology reflex new york times
2        breitbart news daily gorsuch scotus breitbart
3  found nugget podesta files fastwalkers dsp pro...
4  despite democrat obstruction u senate confirms...
```

```python
print(data_non_null[['author', 'author_clean']].head(5))
```

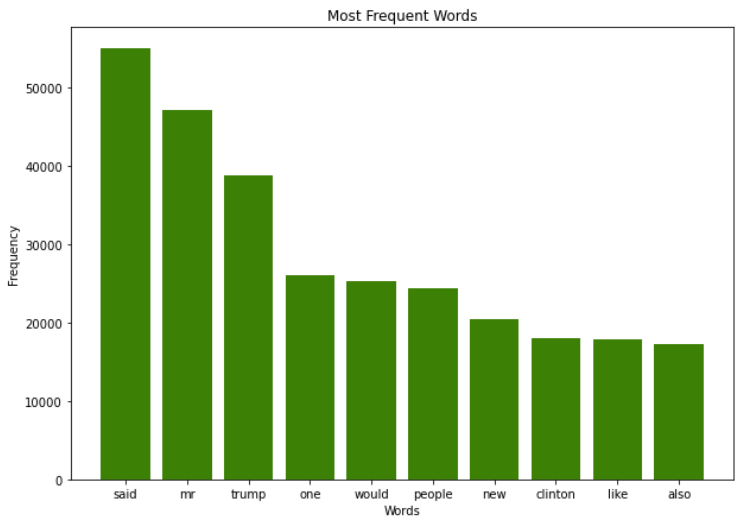
```
                  author            author_clean
0  Jenna Leigh Richardson  jenna leigh richardson
1          Audrey S. Lee              audrey lee
2         Breitbart News          breitbart news
3            seentoomuch             seentoomuch
4      Warner Todd Huston      warner todd huston
```

We would like to see which words are the most frequent in the text. With a simple wordcloud and bar plot, we can visualize them. From the graphs, we can know that "said", "trump", "one", "would", "people" etc. are example of common words. They have each appeared for more than 20000 times.

```python
# visualize the most frequent words in text

all_words = ' '.join([text for text in data_non_null['text_clean']])
wordcloud = WordCloud(width = 800, height = 500, random_state = 21, max_font_size = 110).generate(all_words)
plt.figure(figsize = (10, 7))
plt.imshow(wordcloud, interpolation = "bilinear")
plt.axis('off')
plt.show()
```



```python
# visualize most frequent words

freq = pd.Series(' '.join(data_non_null['text_clean']).split()).value_counts()[:10]
plt.figure(figsize = (10, 7))
plt.bar(freq.index, freq, color = 'green')
plt.xlabel('Words')
plt.ylabel('Frequency')
plt.title('Most Frequent Words')
plt.show()
```




Before heading to modelling, we analyze the sentiment of the text. With the help of the powerful library TextBlob, a convenient tool for processing textual data, we obtain a general sentiment of 0.0109, which is slightly positive. TextBlob defines 1 as the most positive text while -1 as the most negative.

```
# overall tweet sentiment

overall_sentiment = TextBlob(str(data_non_null['text_clean'])).sentiment[0]
print("The overall sentiment of all tweets is: " + str(overall_sentiment), "1: positive", "-1: negative", sep = "\n")
```

```
The overall sentiment of all tweets is: 0.010909090909090901
1: positive
-1: negative
```

## Modelling

Before splitting the data, we need to drop redundant columns such as ‘index’, ‘Unnamed: 0’, ‘title’, ‘author’, and ‘text’ since we already have the cleaned data ‘text_clean’, ‘title_clean’, and ‘author_clean’.

```
# drop unnecessary columns

data_final = data_non_null.drop(['index', 'Unnamed: 0', 'title', 'author', 'text'], axis = 1)
data_final
```

| | id | label | text_length | text_clean | title_clean | author_clean |
|---|---|---|---|---|---|---|
| **0** | 16284 | 1 | 1219 | cbs denver voter fraud investigation exposed d... | voting dead cbs investigation exposes voter fr... | jenna leigh richardson |
| **1** | 16572 | 0 | 4739 | editors note one favorite stories archives fea... | suppress apology reflex new york times | audrey lee |
| **2** | 1000 | 0 | 1211 | wednesday edition breitbart news daily broadca... | breitbart news daily gorsuch scotus breitbart | breitbart news |
| **3** | 13128 | 1 | 2758 | found nugget podesta files fastwalkers dsp pro... | found nugget podesta files fastwalkers dsp pro... | seentoomuch |
| **4** | 2186 | 0 | 1708 | monday evening kansas republican mike pompeo c... | despite democrat obstruction u senate confirms... | warner todd huston |
| **...** | ... | ... | ... | ... | ... | ... |
| **13165** | 11064 | 0 | 8319 | american financial system collapsed fall steph... | stephen bannon rookie campaign chief loves fig... | michael barbaro michael grynbaum |
| **13166** | 9473 | 1 | 26210 | new york times apologist power november specia... | new york times apologist power | consortiumnews com |
| **13167** | 1708 | 0 | 2794 | new law attempts preserve history making illeg... | alabama makes illegal remove confederate monum... | ryan saavedra |
| **13168** | 1486 | 1 | 443 | notify follow comments email notify new posts ... | say chicago tribune calls democrats force clin... | barenakedislam |
| **13169** | 11983 | 0 | 636 | following violent protest earlier week breitba... | milo free speech threat college campuses breit... | trent baker |

13170 rows × 6 columns

From the library sklearn, there is a function called TfidfVectorizer to convert a collection of raw documents to a matrix of TF-IDF features. TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. We apply the vectorizer on the text to transform it into an array of numerical data for model training and testing.

```
# convert the tweets into statistical numbers

tfidf_vectorizer = TfidfVectorizer(ngram_range = (1, 2), max_df = 0.75, min_df = 5, max_features = data_len)

# TF-IDF feature matrix
tfidf = tfidf_vectorizer.fit_transform(data_final['text_clean'] )
tfidf
```

```
<13170x13170 sparse matrix of type '<class 'numpy.float64'>'
	with 3497512 stored elements in Compressed Sparse Row format>
```

### *i. Logistic Regression*

The result of logistic regression is excellent for both labels, with an accuracy of 95.94%.

```
# Logistic Regression

model_lr = LogisticRegression(C = 2, max_iter = 500).fit(X_train_tfidf, y_train)
y_pred_lr = model_lr.predict(X_test_tfidf)
report_lr = classification_report(y_test, y_pred_lr)

print(report_lr)
acc_lr = accuracy_score(y_test, y_pred_lr)
label_lr = "Logistic Regression"
print(label_lr)
print('Accuracy: ', acc_lr)
```

```
              precision    recall  f1-score   support

           0       0.96      0.96      0.96      1457
           1       0.95      0.96      0.95      1177

    accuracy                           0.96      2634
   macro avg       0.96      0.96      0.96      2634
weighted avg       0.96      0.96      0.96      2634

Logistic Regression
Accuracy:  0.9593773728170083
```

### *ii. Decision Tree*

Decision tree is slightly behind logistic regression with an accuracy of 88.19% but still the result is very satisfactory.

```
# Decision Tree

model_dt = tree.DecisionTreeClassifier().fit(X_train_tfidf, y_train)
y_pred_dt = model_dt.predict(X_test_tfidf)
report_dt = classification_report(y_test, y_pred_dt)

print(report_dt)
acc_dt = accuracy_score(y_test, y_pred_dt)
label_dt = 'Decision Tree'
print(label_dt)
print('Accuracy: ', acc_dt)
```

```
              precision    recall  f1-score   support

           0       0.90      0.88      0.89      1457
           1       0.86      0.88      0.87      1177

    accuracy                           0.88      2634
   macro avg       0.88      0.88      0.88      2634
weighted avg       0.88      0.88      0.88      2634

Decision Tree
Accuracy:  0.8819286256643888
```

### *iii. K-nearest Neighbors*

The outcome of kNN is disappointing, with an accuracy of 45.06%. It may be due to the huge size of the dataset. It is also relatively slower than the other algorithms.

```
# kNN

model_knn = KNeighborsClassifier(n_neighbors = 3, weights = 'distance', p = 1).fit(X_train_tfidf, y_train)
y_pred_knn = model_knn.predict(X_test_tfidf)
report_knn = classification_report(y_test, y_pred_knn)

print(report_knn)
acc_knn = accuracy_score(y_test, y_pred_knn)
label_knn = 'K-nearest Neighbors'
print(label_knn)
print('Accuracy: ', acc_knn)
```

```
              precision    recall  f1-score   support

           0       1.00      0.01      0.01      1457
           1       0.45      1.00      0.62      1177

    accuracy                           0.45      2634
   macro avg       0.72      0.50      0.32      2634
weighted avg       0.75      0.45      0.28      2634

K-nearest Neighbors
Accuracy:  0.4506454062262718
```

### *iv. Random Forest*

Random forest also gives an outstanding result, with an accuracy of 94.84%, close to that of logistic regression.

```
# Random Forest

model_rf = RandomForestClassifier(n_estimators = 200).fit(X_train_tfidf, y_train)
y_pred_rf = model_rf.predict(X_test_tfidf)
report_rf = classification_report(y_test, y_pred_rf)

print(report_rf)
acc_rf = accuracy_score(y_test, y_pred_rf)
label_rf = 'Random Forest'
print(label_rf)
print('Accuracy: ', acc_rf)
```

```
              precision    recall  f1-score   support

           0       0.94      0.97      0.95      1457
           1       0.96      0.92      0.94      1177

    accuracy                           0.95      2634
   macro avg       0.95      0.95      0.95      2634
weighted avg       0.95      0.95      0.95      2634

Random Forest
Accuracy:  0.9483675018982536
```

### *v. Support Vector Machine*

SVM is the best algorithm so far, with an accuracy of 96.28%.

```
# SVM

model_svm = LinearSVC().fit(X_train_tfidf, y_train)
y_pred_svm = model_svm.predict(X_test_tfidf)
report_svm = classification_report(y_test, y_pred_svm)

print(report_svm)
acc_svm = accuracy_score(y_test, y_pred_svm)
label_svm = 'SVM'
print(label_svm)
print('Accuracy: ', acc_svm)
```

```
              precision    recall  f1-score   support

           0       0.97      0.96      0.97      1457
           1       0.96      0.96      0.96      1177

    accuracy                           0.96      2634
   macro avg       0.96      0.96      0.96      2634
weighted avg       0.96      0.96      0.96      2634

SVM
Accuracy:  0.9627942293090357
```

### *vi. Ridge Regression*

Ridge regression also yields a stunning accuracy of 95.94%.

```
# Ridge Regression

model_ridge = RidgeClassifier().fit(X_train_tfidf, y_train)
y_pred_ridge = model_ridge.predict(X_test_tfidf)
report_ridge = classification_report(y_test, y_pred_ridge)

print(report_ridge)
acc_ridge = accuracy_score(y_test, y_pred_ridge)
label_ridge = 'Ridge Regression'
print(label_ridge)
print('Accuracy: ', acc_ridge)
```

```
              precision    recall  f1-score   support

           0       0.97      0.96      0.96      1457
           1       0.95      0.96      0.95      1177

    accuracy                           0.96      2634
   macro avg       0.96      0.96      0.96      2634
weighted avg       0.96      0.96      0.96      2634

Ridge Regression
Accuracy:  0.9593773728170083
```

## Result

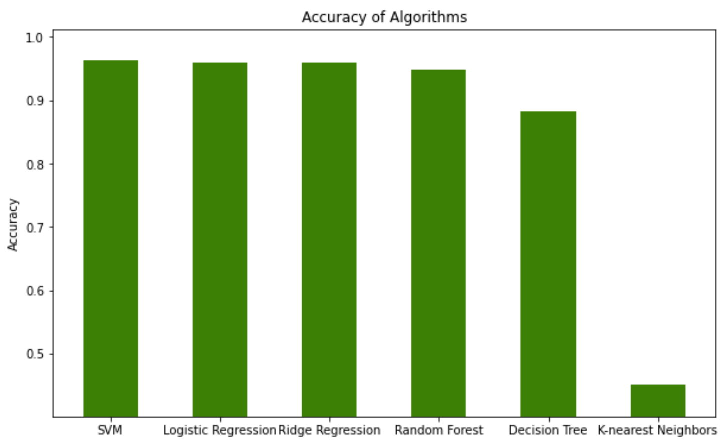
All the results are summarized in the below table and bar plot. The best model is SVM which I am going to use to predict. Other than kNN, all the other algorithms show similar accuracy, which is around 90% or more.

```
# summary of modelling results

d0 = pd.DataFrame([label_lr, label_dt, label_knn, label_rf, label_svm, label_ridge])
d1 = pd.DataFrame([acc_lr, acc_dt, acc_knn, acc_rf, acc_svm, acc_ridge])
d2 = pd.concat([d0, d1], axis = 1)
d2.columns = ['Model', 'Accuracy']
d2 = d2.sort_values(by = ['Accuracy'], ascending = False)

display(d2)
print('The best model is', d2.iloc[0, 0])
```

| | Model | Accuracy |
|---|---|---|
| 4 | SVM | 0.962794 |
| 0 | Logistic Regression | 0.959377 |
| 5 | Ridge Regression | 0.959377 |
| 3 | Random Forest | 0.948368 |
| 1 | Decision Tree | 0.881929 |
| 2 | K-nearest Neighbors | 0.450645 |

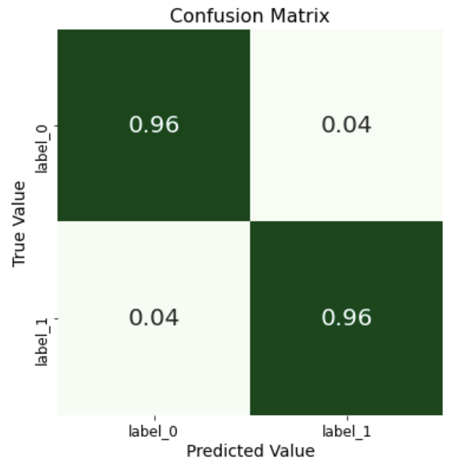
```
The best model is SVM
```

```
# plot the graph of accuracy of algorithms

plt.figure(figsize = (10, 6))
plt.bar(d2['Model'], d2['Accuracy'], width = 0.5, color = 'green')
plt.title('Accuracy of Algorithms')
plt.ylabel('Accuracy')

lower_bound = d2.iloc[-1, -1] - 0.05
upper_bound = d2.iloc[0, 1] + 0.05
plt.ylim(lower_bound, upper_bound)
plt.show()
```




I have also generated a confusion matrix of SVM to take a look of how the algorithm performs. You can see SVM is a successful model to obtain a good result.

```python
# visualize the confusion matrix of the most accurate algorithm

confusion_matrix_final = confusion_matrix(y_test, y_pred_lr)
matrix_proportions = np.zeros((2, 2))
for i in range(0,2):
    matrix_proportions[i, :] = confusion_matrix_final[i, :] / float(confusion_matrix_final[i, :].sum())
names = ['label_0', 'label_1']
confusion_df = pd.DataFrame(matrix_proportions, index = names, columns = names)
plt.figure(figsize = (6, 6))
sns.heatmap(confusion_df, annot = True, annot_kws = {"size": 20}, cmap = 'Greens', cbar = False, square = True,fmt = '.
plt.title('Confusion Matrix', fontsize = 16)
plt.ylabel(r'True Value', fontsize = 14)
plt.xlabel(r'Predicted Value', fontsize = 14)
plt.tick_params(labelsize = 12)
```




## Prediction

We read the testing dataset to predict. It has 4000 rows and 4 variables, namely ‘id’, ‘title’, ‘author’, and ‘text’, similar to the training dataset.

```python
# read testing dataset

test_data = pd.read_csv("/Users/ching/Downloads/testing_set30 .csv", sep = ',')
test_data
```

| | Unnamed: 0 | id | title | author | text |
|---|---|---|---|---|---|
| 0 | 4890 | 25689 | Splash! Jerry Brown Says California's Drought ... | Joel B. Pollak | California Gov. Jerry Brown declared Friday th... |
| 1 | 3599 | 24398 | GOP Sen Lee: 'Would Not Say No' to SCOTUS Offe... | Jeff Poor | In an interview on Salt Lake City's KSL's “The... |
| 2 | 2061 | 22860 | In a Late-Night Move, Russia Arrests a Top Eco... | Neil MacFarquhar | MOSCOW — Russians awoke to an extraordinary... |
| 3 | 1314 | 22113 | Hillary Clinton Planning Trump Russian Agent L... | The European Union Times | \r\nHillary Clinton has been caught in yet ano... |
| 4 | 2658 | 23457 | Obamacare architect: 'The law is working as de... | Howard Portnoy | Print \r\nOn Wednesday's broadcast of “CNN New... |
| ... | ... | ... | ... | ... | ... |
| 3995 | 1848 | 22647 | Anonymous – World War 3 is imminent 2016 | Pakalert | |
| 3996 | 3970 | 24769 | Rick Perry Accuses Texas A&ampM of Disqualifyi... | Tom Ciccotta | Rick Perry has accused administrators at Texas... |
| 3997 | 3408 | 24207 | Cuba Gooding Jr. Sparks Outrage After Lifting ... | Jerome Hudson | Actor Cuba Gooding Jr. came under fire on soci... |
| 3998 | 3424 | 24223 | Diplomats Confront New Threat to Paris Climate... | Coral Davenport | MARRAKESH, Morocco — Diplomats from around ... |
| 3999 | 1182 | 21981 | Clinton's most emailed aid in the US State Dep... | Pamela Geller | Home Search Posted by Pamela Geller \r\nClinto... |

4000 rows × 5 columns

Again with a simple function we can immediately discover that some rows have *nan* values. We remove them and perform data cleaning just like before as the model can only take the same data structure.

```
RangeIndex: 4000 entries, 0 to 3999
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Unnamed: 0  4000 non-null   int64 
 1   id          4000 non-null   int64 
 2   title       3911 non-null   object
 3   author      3607 non-null   object
 4   text        3995 non-null   object
dtypes: int64(2), object(3)
memory usage: 156.4+ KB
```

```python
# remove rows with nan values

test_data_non_null = test_data.dropna(how = 'any', axis = 0).reset_index()
test_data_non_null
```

| | index | Unnamed: 0 | id | title | author | text |
|---|---|---|---|---|---|---|
| **0** | 0 | 4890 | 25689 | Splash! Jerry Brown Says California's Drought ... | Joel B. Pollak | California Gov. Jerry Brown declared Friday th... |
| **1** | 1 | 3599 | 24398 | GOP Sen Lee: 'Would Not Say No' to SCOTUS Offe... | Jeff Poor | In an interview on Salt Lake City's KSL's "The... |
| **2** | 2 | 2061 | 22860 | In a Late-Night Move, Russia Arrests a Top Eco... | Neil MacFarquhar | MOSCOW — Russians awoke to an extraordinary... |
| **3** | 3 | 1314 | 22113 | Hillary Clinton Planning Trump Russian Agent L... | The European Union Times | \r\nHillary Clinton has been caught in yet ano... |
| **4** | 4 | 2658 | 23457 | Obamacare architect: 'The law is working as de... | Howard Portnoy | Print \r\nOn Wednesday's broadcast of "CNN New... |
| **...** | ... | ... | ... | ... | ... | ... |
| **3513** | 3995 | 1848 | 22647 | Anonymous – World War 3 is imminent 2016 | Pakalert | |
| **3514** | 3996 | 3970 | 24769 | Rick Perry Accuses Texas A&ampM of Disqualifyi... | Tom Ciccotta | Rick Perry has accused administrators at Texas... |
| **3515** | 3997 | 3408 | 24207 | Cuba Gooding Jr. Sparks Outrage After Lifting ... | Jerome Hudson | Actor Cuba Gooding Jr. came under fire on soci... |
| **3516** | 3998 | 3424 | 24223 | Diplomats Confront New Threat to Paris Climate... | Coral Davenport | MARRAKESH, Morocco — Diplomats from around ... |
| **3517** | 3999 | 1182 | 21981 | Clinton's most emailed aid in the US State Dep... | Pamela Geller | Home Search Posted by Pamela Geller \r\nClinto... |

3518 rows × 6 columns

```python
# clean the data and combine them with the original dataset

text_clean_test = cleaning(test_data_non_null['text'])
title_clean_test = cleaning(test_data_non_null['title'])
author_clean_test = cleaning(test_data_non_null['author'])

test_data_non_null['text_clean'] = text_clean_test
test_data_non_null['title_clean'] = title_clean_test
test_data_non_null['author_clean'] = author_clean_test
```

```python
# drop unnecessary columns

test_data_final = test_data_non_null.drop(['index', 'Unnamed: 0', 'title', 'author', 'text'], axis = 1)
test_data_final
```

| | id | text_clean | title_clean | author_clean |
|---|---|---|---|---|
| **0** | 25689 | california gov jerry brown declared friday dro... | splash jerry brown says california drought bre... | joel b pollak |
| **1** | 24398 | interview salt lake city ksl doug wright show ... | gop sen lee would say scotus offer breitbart | jeff poor |
| **2** | 22860 | moscow russians awoke extraordinary scandal tu... | late night move russia arrests top economic of... | neil macfarquhar |
| **3** | 22113 | hillary clinton caught yet another dirty trick... | hillary clinton planning trump russian agent l... | european union times |
| **4** | 23457 | print wednesday broadcast cnn newsroom mit eco... | obamacare architect law working designed needs... | howard portnoy |

Fit the testing dataset into the SVM model and we can immediately obtain the prediction:

```python
# prediction

y_pred_svm_test = model_svm.predict(tfidf_test)
test_data_final['prediction'] = y_pred_svm_test
test_data_final
```

| | id | text_clean | title_clean | author_clean | prediction |
|---|---|---|---|---|---|
| **0** | 25689 | california gov jerry brown declared friday dro... | splash jerry brown says california drought bre... | joel b pollak | 1 |
| **1** | 24398 | interview salt lake city ksl doug wright show ... | gop sen lee would say scotus offer breitbart | jeff poor | 1 |
| **2** | 22860 | moscow russians awoke extraordinary scandal tu... | late night move russia arrests top economic of... | neil macfarquhar | 1 |
| **3** | 22113 | hillary clinton caught yet another dirty trick... | hillary clinton planning trump russian agent l... | european union times | 1 |
| **4** | 23457 | print wednesday broadcast cnn newsroom mit eco... | obamacare architect law working designed needs... | howard portnoy | 1 |
| **...** | ... | ... | ... | ... | ... |
| **3513** | 22647 | | anonymous world war imminent | pakalert | 1 |
| **3514** | 24769 | rick perry accused administrators texas disqua... | rick perry accuses texas ampm disqualifying st... | tom ciccotta | 1 |
| **3515** | 24207 | actor cuba gooding jr came fire social media s... | cuba gooding jr sparks outrage lifting sarah p... | jerome hudson | 0 |
| **3516** | 24223 | marrakesh morocco diplomats around world conve... | diplomats confront new threat paris climate pa... | coral davenport | 1 |
| **3517** | 21981 | home search posted pamela geller clinton email... | clinton emailed aid us state department jake s... | pamela geller | 1 |

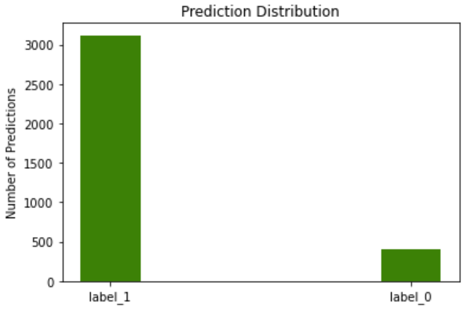
3518 rows × 5 columns

The number of predicted unreliable article is 3117 and that of reliable article is 401. In contrast to the training dataset, the testing dataset has a much higher proportion of fake news as you can see from the bar plot. Since the model is highly accurate, we can conclude that the testing dataset has more unreliable news than reliable news, and higher proportion of unreliable news than that of training dataset.

```
# calculate no of labels

pred = pd.DataFrame({'label_1': [np.sum(test_data_final['prediction'] == 1)], 'label_0': [np.sum(test_data_final['predi

print('Number of articles predicted as unreliable (label 1):', pred.iloc[0,0])
print('Number of articles predicted as reliable (label 0):', pred.iloc[0,1])
```

```
Number of articles predicted as unreliable (label 1): 3117
Number of articles predicted as reliable (label 0): 401
```

```
# visualize the prediction result

x, y = pred.columns, pred.iloc[0]
plt.bar(x, y, width = 0.2, color = 'green')
plt.title('Prediction Distribution')
plt.ylabel('Number of Predictions')
plt.show()
```




## Recommendation

We all have to be careful when we read the news online. Since there is an explosive number of news media and social media, we are exposed to a sea of overwhelming sources of news. We have to always pay attention to the news source we read. Pick the credible ones such as AFP, AP, and Reuters. These are wire service companies which provide the most objective news you can find.

Other than that, we can also read news from renowned public agencies such as BBC, NPR, France 24, and DW which also offer news of quality. They are regulated by the government and that’s why they have more limitations on the news and messages they can deliver. Advertisement and propaganda are strictly banned among these outlets.

Avoid unknown online news media. You will never know how they come up with the information or whether the news is correct or not. On the other hand, we do have some impressive and well-known online news media such as Vice News and Vox but they tend to be more subjective than the aforementioned ones. Be more careful when you read news from these outlets.

## Final Thoughts

It is hard to completely avoid fake news in these days and age as we are already part of the internet. We have to always understand the news source first before absorbing the information. Fact checking can also help but can we really check every piece of news? It seems that more regulations may be a solution to this problem.