

```
In [2]: import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
In [3]: df = pd.read_csv('news.csv')
df
```

Out[3]:

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
...
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

6335 rows × 4 columns

```
In [4]: import nltk
stopwords = nltk.corpus.stopwords.words('english')
```

```
In [5]: tfidf = TfidfVectorizer()
X = tfidf.fit_transform(df['text'])
```

```
In [6]: tfidf.get_feature_names_out()

Out[6]: array(['00', '000', '0000', ..., 'والمرضى', 'هذا', 'fake'], dtype=object)
```

```
In [7]: df2 = pd.DataFrame(X.toarray())
df2.columns = [tfidf.get_feature_names_out()]
df2
```

Out[7]:

	00	000	0000	000000031	00000031	000035	00006	0001	0001pt	0002	...	حلب	عربي	عن	لم	ما	محاولات	من	هذا	مرضى
0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.028139	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...
6330	0.0	0.029299	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6331	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6332	0.0	0.007522	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6333	0.0	0.011331	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6334	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

6335 rows × 67659 columns

```
In [8]: df2.sum()
```

Out[8]:

00	3.560481
000	37.380637
0000	0.097361
000000031	0.026123
00000031	0.053099
...	...
0.041217	محاولات
0.041217	من
0.041217	هذا
0.041217	والمرضى
fake	0.032057

Length: 67659, dtype: float64

```
In [9]: y = df.iloc[:,3]
y
```

Out[9]:

0	FAKE
1	FAKE
2	REAL
3	FAKE
4	REAL
...	...
6330	REAL
6331	FAKE
6332	FAKE
6333	REAL
6334	REAL

Name: label, Length: 6335, dtype: object

```
In [10]: from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
y = le.fit_transform(y)
y

Out[10]: array([0, 0, 1, ..., 0, 1, 1])
```

```
In [11]: x = df2.values
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.20,random_state=0)
```

```
In [12]: from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier( criterion='gini', splitter='best')
classifier.fit(x_train,y_train)

Out[12]: DecisionTreeClassifier()
```

```
In [13]: y_pred = classifier.predict(x_test)
```

```
In [14]: y_pred

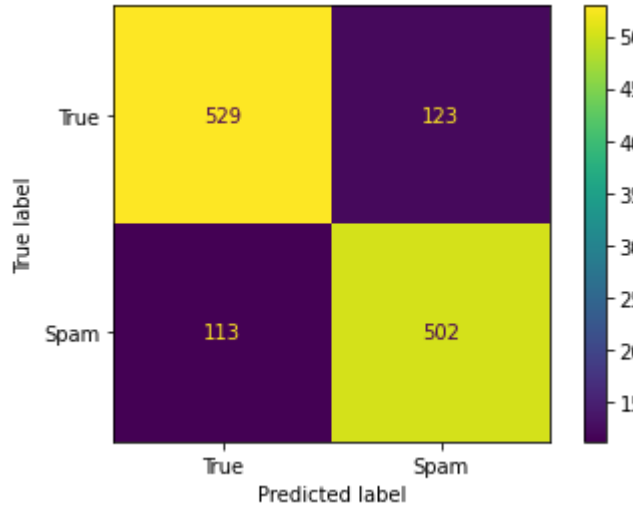
Out[14]: array([1, 0, 0, ..., 0, 1, 1])
```

```
In [15]: from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay
cm = confusion_matrix(y_test,y_pred, labels=[1,0])
disp = ConfusionMatrixDisplay(confusion_matrix=cm , display_labels=['True','Spam'])
```

```
In [16]: import seaborn as sns
```

```
In [17]: disp.plot()
```

Out[17]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x2c9627354e0>



```
In [18]: from sklearn.metrics import classification_report

cr = classification_report(y_test, y_pred)
print(cr)
```

	precision	recall	f1-score	support
0	0.80	0.82	0.81	615
1	0.82	0.81	0.82	652
accuracy			0.81	1267
macro avg	0.81	0.81	0.81	1267
weighted avg	0.81	0.81	0.81	1267

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```