Comparison of various algorithms for fake review detection on e-commerce websites

```
import pandas as pd

df=pd.read_csv('fake reviews dataset.csv')

df.head()


### dropping the missing values

df=df.dropna()


##Independent features

X=df.drop('label',axis=1)


##Dependent features

y=df['label']


X.shape


y.shape
```

```
import tensorflow as tf
```

```
tf.__version__
```

```
voc_size=5000
```

```
messages=X.copy()
```

```
messages['text_'][1]
```

```
messages.reset_index(inplace=True)
```

```
import nltk

import re

from nltk.corpus import stopwords



nltk.download('stopwords')



### data preprocessing



from nltk.stem.porter import PorterStemmer

ps = PorterStemmer()

corpus = []

for i in range(0, len(messages)):

    print(i)
```

```
    review = re.sub('[^a-zA-Z]', ' ', messages['text_'][i])
    review = review.lower()
    review = review.split()

    review = [ps.stem(word) for word in review if not word in stopwords.words('english')]
    review = ' '.join(review)
    corpus.append(review)

corpus

### One hot representation
onehot_repr=[one_hot(words,voc_size)for words in corpus]
onehot_repr

sent_length=20
embedded_docs=pad_sequences(onehot_repr,padding='pre',maxlen=sent_length)

print(embedded_docs)

embedded_docs[0]
```

```
## Creating model

embedding_vector_features=40

model=Sequential()

model.add(Embedding(voc_size,embedding_vector_features,input_length=sent_length))

model.add(LSTM(100))

model.add(Dense(1,activation='sigmoid'))

model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])

print(model.summary())



import numpy as np

X_final=np.array(embedded_docs)

y_final=np.array(y)




X_final.shape,y_final.shape




from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X_final, y_final, test_size=0.33, random_state=42)
```

```
### Model training
model.fit(
    X_train,
    y_train,
    epochs=12,
    verbose="auto",
)


y_pred = (model.predict(X_test) > 0.5).astype("int32")



from sklearn.metrics import confusion_matrix


confusion_matrix(y_test,y_pred)


from sklearn.metrics import accuracy_score
```

5.1 Output

```
In [194]: from sklearn.metrics import accuracy_score
          accuracy_score(y_test,y_pred)

Out[194]: 0.475605186239976
```

Figure 3: Sample Output