In [30]:
```
import numpy as np
from sklearn.datasets import fetch_20newsgroups
```

In [31]:
```
from sklearn.feature_extraction.text import CountVectorizer
```

In [32]:
```
from sklearn.naive_bayes import BernoulliNB,MultinomialNB
#input features are binary
#input features are multiple in mutlinomial (check frequency)
```

In [34]:
```
newsgroups=fetch_20newsgroups(subset='all')
```

In [35]:
```
vectorizer1=CountVectorizer(binary=True) #words are there are not
vectorizer2=CountVectorizer(binary=False) #words are repeated or not
```

In [36]:
```
X1=vectorizer1.fit_transform(newsgroups.data)
X2=vectorizer2.fit_transform(newsgroups.data)
```

In [37]:
```
y=newsgroups.target
```

In [38]:
```
from sklearn.model_selection import train_test_split
xtrain1,xtest1,ytrain,ytest=train_test_split
(X1,y,test_size=0.25,random_state=1) #X1 binary values
xtrain2,xtest2,ytrain,ytest=train_test_split
(X2,y,test_size=0.25,random_state=1) #X2 multivalues
```

In [39]:
```
bnb=BernoulliNB()
bnb
mnb=MultinomialNB()
mnb
```

Out[39]:
```
MultinomialNB
MultinomialNB()
```

In [40]:
```
bnb.fit(xtrain1,ytrain)
```

Out[40]:
```
BernoulliNB
BernoulliNB()
```

In [41]:
```
1 mnb.fit(xtrain2,ytrain)
```

Out[41]:
```
MultinomialNB
MultinomialNB()
```

In [42]:
```
1 y_pred1=bnb.predict(xtest1)
2 y_pred1
```

Out[42]:
```
array([16,  9, 18, ..., 13,  7, 14])
```

In [43]:
```
1 y_pred2=mnb.predict(xtest2)
2 y_pred2
```

Out[43]:
```
array([16, 19, 18, ..., 13,  7, 14])
```

In [44]:
```
1 from sklearn.metrics import accuracy_score
```

In [45]:
```
1 accuracy_score(ytest,y_pred1)
```

Out[45]:
```
0.681239388794567
```

In [46]:
```
1 accuracy_score(ytest,y_pred2)
```

Out[46]:
```
0.8384974533106961
```

Conclusion: Multinomial is better than bernoulli for this corpus because it checks the frequency.

In [47]:
```
1 from sklearn.feature_extraction.text import TfidfVectorizer
```

In [48]:
```
1 from sklearn.pipeline import make_pipeline
2 model=make_pipeline(TfidfVectorizer(),MultinomialNB())
```

In [49]:
```
1 test_data=fetch_20newsgroups(subset='test')
2 train_data=fetch_20newsgroups(subset='train')
```

In [50]:
```
1 model.fit(train_data.data,train_data.target)
```

Out[50]:
```
Pipeline
TfidfVectorizer
MultinomialNB
```

In [52]:
```
predictions_tf=model.predict(test_data.data)
predictions_tf
```

Out[52]:
```
array([ 7, 11,  0, ...,  9,  3, 15])
```

In [53]:
```
accuracy_score(test_data.target,predictions_tf)
```

Out[53]:
```
0.7738980350504514
```