```
#import movie_reviews
from nltk.corpus import movie_reviews

#define feature extractor for documents

documents = [(list(movie_reviews.words(fileid)), category)
    for category in movie_reviews.categories()
    for fileid in movie_reviews.fileids(category)]
random.shuffle(documents)

#construct list of 2000 most frequent words in corpus
#check whether words is present in given document
all_words = nltk.FreqDist(w.lower() for w in movie_reviews.words())
word_features=list(all_words)[:2000]

def document_features(document):
    document_words=set(document)
    features = {}
    for word in word_features:
        features['contains({})'.format(word)] = (word in document_words)
    return features

#train a classifier to label new movie reviews
featuresets = [(document_features(d), c) for (d,c) in documents]
train_set, test_set = featuresets[100:], featuresets[:100]
classifier = nltk.NaiveBayesClassifier.train(train_set)

print(nltk.classify.accuracy(classifier, test_set))
```

#check accuracty of classifer

print(nltk.classify.accuracy(classifier, test_set))

```
>>> print(nltk.classify.accuracy(classifier, test_set))
0.69
```

#find the most informative classifiers

```
>>> classifier.show_most_informative_features(5)
Most Informative Features
    contains(mediocrity) = True              neg : pos    =      7.6 : 1.0
         contains(jules) = True              pos : neg    =      7.0 : 1.0
       contains(coyote) = True               neg : pos    =      7.0 : 1.0
      contains(tribute) = True               pos : neg    =      6.6 : 1.0
     contains(obstacle) = True               pos : neg    =      6.4 : 1.0
```